[10] M. Unser and A. Aldroubi, “Polynomial Splines and Wavelets- A Signal Processing Perspective”, Wavelets- A Tutorial in Theory and Applications, C. K. Chui (ed.) pp. 91-112 (1992), Academic Press, Inc.

References

[1] P. Auscher, “Wavelets with boundary conditions on the interval,” pp. 217-236, Wavelets: A tutorial in theory and applications, ed. C. K. Chui, Academic Press, 1992.

[2] G. Beylkin, “On the Representation of Operators in Bases of Compactly Supported Wavelets”, SIAM J. Num. Anal., 29(6): 1716-1740, 1992.

[3] A. Cohen, I. Daubechies, and P. Vial, “Wavelets on the interval and fast wavelet transforms”, Submitted to Applied and Computational Harmonic Analysis.

[4] I. Daubechies, “Orthonormal Basis of Compactly Supported Wavelets”, Comm. Pure Appl. Math., **41** (1988), pp. 909-996.

[5] D. Gottlieb, B. Gustafsson, P. Olsson, and B. Strand, “On the Superconvergence of Galerkin Methods for Hyperbolic IBVP”, July 29, 1993, submitted to SIAM J. Sci. Comp.

[6] L. Jameson, “On The Wavelet Based Differentiation Matrix”, Journal of Scientific Computing, September 1993, and ICASE Report No. 93-95, NASA CR-191583.

[7] L. Jameson, “On the Spline-based Wavelet Differentiation Matrix”, Submitted to SIAM J. Num. Anal. and ICASE Report No. 93-80, NASA CR-191557.

[8] Y. Meyer, “Ondelettes sur I’ntervalle”, Revista Matematica Iberoamericana 7, 1992, pp 115-133.

[9] G. Strang, “Wavelets and Dilation Equations: A Brief Introduction”, SIAM Review, Vol. 31, No.4, pp. 614-627, Dec. 1989.

superconvergence encountered with periodic boundary conditions if there are characteristics leaving the domain. Note that the finite element subspace is the same as the first order spline wavelet subspace, see [7].

In conclusion, under the currently available boundary constructions the superconvergence, which is encountered under periodic boundary conditions, is not maintained at the boundaries. In the case of the D_4 wavelet under the boundary construction examined here one gets 4th-order differentiation away from the boundaries and only 1st-order differentiation at the boundaries producing a scheme with only 2nd-order accurate spatial differentiation. It remains to be seen if it is possible to construct boundary functions for a wavelet subspace which can maintain superconvergence at the boundaries.

8 Other Constructions and Conclusion

In this paper I have explored the differentiation matrix for Daubechies-type wavelets defined on an interval using the boundary construction defined in [3]. The exploration has been limited to the wavelet with 2 vanishing moments usually referred to as the Daubechies 4 wavelet. It was seen that the superconvergence encountered with periodic boundary conditions which was proven in [6] is lost with this boundary construction.

There are other boundary constructions available for Daubechies wavelets, see [1], [8], but none of the constructions can maintain the superconvergence. There is a very straightforward way to find the maximum accuracy at the boundary for a given boundary construction. That is, when the basis functions are orthogonal under translation, as they are for Daubechies wavelets, one can simply count the number of basis functions which have support overlapping the support of the outermost boundary functions. This defines the number of nonzero entries in the first and last rows of the middle matrix D referred to here as the derivative projection matrix. All differencing in wavelet differentiation is done by D, and, therefore, the differentiation accuracy at the boundaries cannot exceed one less than the number of nonzero entries in the first and last rows of D. Of course, this argument holds for every row of D, but I am particularly concerned with the boundaries here.

Perhaps it is not possible to construct boundary functions for wavelets which give everything. That is, boundary functions which maintain the wavelet structure, which maintain the approximation properties across the entire interval and which maintain superconvergence. This conjecture is based first on the fact that to date there is not a boundary construction which satisfies these criteria for either Daubechies wavelets or spline-based wavelets, see [7]. In further support of this conjecture, in a paper by Gottlieb, Gustafsson, Olsson, and Strand, [5], it is proven that it is not possible to construct boundary functions for a finite element subspace which maintain the

Differentiation	Grid	Boundary	Error	Inner	Error	Total	Error
Matrix	Size	l_1 Error	Ratio	l_1 Error	Ratio	l_1 Error	Ratio
$(C_1^{2pt})^{-1}DC_1^{2pt}$	16	.0181		$8.70e^{-4}$		.00598	
Grid $\vec{y}$	32	.00933	1.94	$2.31e^{-4}$	3.77	.00162	3.69
	64	.00474	1.96	$6.04e^{-5}$	3.82	$4.26e^{-4}$	3.80
	128	.00238	1.99	$1.49e^{-5}$	4.05	$1.089e^{-4}$	3.91
	256	.00117	2.03	$3.73e^{-6}$	3.99	$2.72e^{-5}$	4.00
NPD of	16	.0211		$6.82e^{-4}$		.00692	
$(C_1^{2pt})^{-1}DC_1^{2pt}$	32	.0116	1.82	$1.42e^{-5}$	48.0	.00191	3.62
Grid $\vec{y}$	64	.00606	1.91	$5.88e^{-6}$	2.41	$5.05e^{-4}$	3.78
	128	.00310	1.95	$1.43e^{-6}$	4.11	$1.30e^{-4}$	3.88
	256	.00156	1.99	$3.50e^{-7}$	4.09	$3.29e^{-5}$	3.95

Table 2: Differentiation accuracy for Daubechies wavelets defined on an interval using grid points at the boundaries and all other grid points evenly-spaced across the interval.

coming from the left. This problem can be 'fixed' if one simply turns the basis around and builds another differentiation matrix. That is, turn all the inner scaling functions around and interchange the role of the RHS and LHS boundary functions. The new differentiation matrix $\tilde{\mathcal{D}}$ can be found from the unreversed differentiation matrix $\mathcal{D}$ by first flipping the matrix by a middle horizontal axis and then by a middle vertical axis and then negating. That is,

$$\tilde{\mathcal{D}} = -flipud(fliplr(\mathcal{D})), \tag{55}$$

and we build a new NPD differentiation matrix by averaging these two matrices:

$$\mathcal{D}_{NPD} = \frac{1}{2}(\mathcal{D} + \tilde{\mathcal{D}}). \tag{56}$$

The following NPD matrix treats data coming from the left in the same manner as data coming from the right. Note that this matrix is full.

$$\mathcal{D}_{NPD} =$$

$$\begin{bmatrix}
-12.00 & 36.96 & -51.58 & 29.30 & -2.67 & -7e^{-4} & -3e^{-4} & -2e^{-4} & 8e^{-4} & -1e^{-3} & 4e^{-5} & 1e^{-4} \\
-4.47 & -13.12 & 29.85 & -13.47 & 1.22 & -4e^{-3} & -2e^{-3} & -1e^{-3} & 4e^{-3} & -7e^{-3} & 2e^{-3} & 6e^{-4} \\
-.39 & -9.77 & 9.11 & 1.62 & -.54 & -.01 & -6e^{-3} & -4e^{-3} & .01 & -.02 & 7e^{-3} & 2e^{-3} \\
-.43 & -.43 & -3.01 & -3.24 & 8.30 & -1.16 & -.02 & -.01 & .04 & -.07 & .02 & 6e^{-3} \\
-.20 & -.70 & 3.44 & -9.44 & .12 & 7.96 & -1.15 & -.03 & .11 & -.18 & .06 & .02 \\
-.09 & -.32 & 1.09 & .42 & -7.86 & -9e^{-3} & 7.94 & -1.16 & .28 & -.47 & .14 & .04 \\
-.04 & -.14 & .47 & -.28 & 1.16 & -7.94 & 9e^{-3} & 7.86 & -.42 & -1.09 & .32 & .09 \\
-.02 & -.06 & .19 & -.11 & .03 & 1.15 & -7.96 & -.12 & 9.44 & -3.44 & .70 & .20 \\
-6e^{-3} & -.02 & .07 & -.04 & .01 & .02 & 1.16 & -8.30 & 3.24 & 3.01 & .43 & .43 \\
-2e^{-3} & -7e^{-3} & .02 & -.01 & 4e^{-3} & 6e^{-3} & .01 & .54 & -1.62 & -9.11 & 9.77 & .39 \\
-6e^{-4} & -2e^{-3} & 7e^{-3} & -4e^{-3} & 1e^{-3} & 2e^{-3} & 4e^{-3} & -1.22 & 13.47 & -29.85 & 13.11 & 4.47 \\
-1e^{-4} & -4e^{-4} & 1e^{-3} & -8e^{-4} & 2e^{-4} & 3e^{-4} & 7e^{-4} & 2.67 & -29.29 & 51.58 & -36.96 & 12.00
\end{bmatrix}$$

Table (7.2.2) gives the accuracy results for two differentiation matrices constructed from the well-conditioned quadrature matrix C_1^{2pt}.

Again, the following table shows that loss of superconvergence at the boundaries regardless of the differentiation matrix. Also, note that for grid $\vec{y}$ that the superconvergence is not even achieved away from the boundary as it is for grid $\vec{x}$. Inner error and boundary error are defined in the same manner as for the previous table.

In conclusion, in this section I have presented a number of differentiation matrices along with the orders of accuracy of the differentiation. In all cases the superconvergence which is encountered under periodic boundary conditions is lost.

7.2 Accuracy Using Grid $\vec{y}$

We now study the differentiation accuracy where we have included a boundary point at each end of the interval in the grid and the remainder of the grid points are required to be evenly-spaced across the interval. This choice of grid is a much more likely choice than grid $\vec{x}$. However, superconvergence is, again, lost at the boundary. This reduces the total accuracy of the differentiation to 2nd order as with grid $\vec{x}$. Again, let us note a few explicit examples of differentiation matrices.

7.2.1 Differentiation Matrix $(C_1^{2pt})^{-1}DC_1^{2pt}$, Grid $\vec{y}$

Again note the unusual, and undesirerable, form of the following matrix. As can be seen from Table (7.2.2), again, the superconvergence is lost at the boundary.

$$\mathcal{D} =$$

$$\begin{bmatrix}
-1.99 & 3.33 & -1.60 & .26 & -2e^{-4} & -1e^{-4} & -6e^{-5} & -4e^{-5} & 1e^{-4} & -2e^{-4} & 6e^{-5} & 2e^{-5} \\
-.46 & -.14 & .72 & -.13 & -1e^{-3} & -6e^{-4} & -3e^{-4} & -2e^{-4} & 7e^{-4} & -1e^{-3} & 3e^{-4} & 1e^{-4} \\
.08 & -.63 & -.02 & .70 & -.12 & -2e^{-3} & -1e^{-3} & -7e^{-4} & 2e^{-3} & -4e^{-3} & 1e^{-3} & 3e^{-4} \\
0 & .07 & -.63 & -.02 & .69 & -.12 & -3e^{-3} & -2e^{-3} & 7e^{-3} & -.01 & 3e^{-3} & 1e^{-3} \\
0 & 0 & .08 & -.63 & -.02 & .69 & -.11 & -5e^{-3} & .02 & -.03 & 9e^{-3} & 3e^{-3} \\
0 & 0 & 0 & .08 & -.63 & -.02 & .69 & -.12 & .05 & -.08 & .02 & 7e^{-3} \\
0 & 0 & 0 & 0 & .08 & -.63 & -.02 & .68 & 5e^{-3} & -.18 & .05 & .02 \\
0 & 0 & 0 & 0 & 0 & .08 & -.63 & -.04 & .94 & -.50 & .12 & .03 \\
0 & 0 & 0 & 0 & 0 & 0 & .08 & -.69 & .52 & -.12 & .15 & .07 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -.03 & .43 & -1.54 & 1.00 & .14 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -.20 & 2.12 & -4.25 & 2.05 & .29 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & .44 & -4.62 & 7.00 & -2.83 & 8e^{-3}
\end{bmatrix}$$

It has been mentioned many times that the structure is undesirable. This is due to the lack of directional independence of the matrix. One 'fix' for this problem is given in the next subsection.

7.2.2 The NPD Differentiation Matrix $(C_1^{2pt})^{-1}DC_1^{2pt}$, Grid $\vec{y}$

NPD denotes No-Preferred-Direction. It is possible to fix one of the problems with the above matrices, the problem of directional dependence. That is, a general differentiation matrix should never treat data coming from the right differently than data

Differentiation Matrix	Grid Size	Boundary l_1 Error	Error Ratio	Inner l_1 Error	Error Ratio	Total l_1 Error	Error Ratio
$(C_1^{2pt})^{-1}DC_1^{2pt}$	16	.0066		$2.86e^{-4}$		.0024	
Grid $\vec{x}$	32	.0032	2.06	$3.20e^{-6}$	89	$5.73e^{-4}$	4.19
	64	.0016	2.00	$2.10e^{-9}$	1524	$1.41e^{-4}$	4.06
	128	$7.85e^{-4}$	2.04	$2.80e^{-11}$	75	$3.50e^{-5}$	4.03
	256	$3.81e^{-4}$	2.06	$1.75e^{-12}$	16.0	$8.53e^{-6}$	4.10
	512	$1.69e^{-4}$	2.25	$1.07e^{-13}$	16.4	$1.94e^{-6}$	4.40
$(C_2^{2pt})^{-1}DC_2^{2pt}$	16	.0091		$2.71e^{-5}$		.0026	
Grid $\vec{x}$	32	.0047	1.94	$3.13e^{-7}$	86.58	$6.78e^{-4}$	3.83
	64	.0024	1.96	$5.92e^{-10}$	529	$1.72e^{-4}$	3.94
	128	.0012	2.00	$2.80e^{-11}$	21.1	$4.33e^{-5}$	3.97
	256	$6.08e^{-4}$	1.97	$1.75e^{-12}$	16.0	$1.10e^{-5}$	3.94
	512	$3.22e^{-4}$	1.89	$1.07e^{-13}$	16.4	$2.89e^{-6}$	3.81
$\hat{C}_{\vec{x}}DC_1^{3pt}$	16	.0038		$6.52e^{-5}$		.0013	
Grid $\vec{x}$	32	.0019	2.00	$1.60e^{-5}$	4.08	$3.15e^{-4}$	4.13
	64	$9.23e^{-4}$	2.06	$3.95e^{-6}$	4.05	$7.87e^{-5}$	3.99
	128	$4.64e^{-4}$	1.99	$9.83e^{-7}$	4.02	$1.97e^{-5}$	3.99
	256	$2.32e^{-4}$	2.00	$2.45e^{-7}$	4.01	$4.92e^{-6}$	4.00
	512	$1.18e^{-4}$	1.97	$6.12e^{-8}$	4.00	$1.22e^{-6}$	4.04

Table 1: Differentiation accuracy for Daubechies wavelets defined on an interval using grid points halfway between the integers.

The noteworthy points are, first of all, that in all three cases the accuracy at the boundary is first order. Recall that when periodic boundary conditions are imposed that the differentiation accuracy is 4th order for the D_4 wavelet. This loss of superconvergence is a serious problem. Furthermore, note from the table that the two well-conditioned quadrature matrices maintain the superconvergence away from the boundary. But, in all three cases the total error of the differentiation matrix across the entire domain is one higher than the boundary accuracy, i.e., we get 2nd order differentiation accuracy.

Note: Inner accuracy is calculated at the middle 50% of the grid points, and boundary accuracy is calculated at the two outermost grid points at each end of the interval.

$$\begin{bmatrix}
-25.33 & 42.38 & -20.77 & 3.73 & -6e^{-4} & 0 & 0 & -7e^{-4} & 7e^{-3} & -9e^{-3} & 2e^{-5} & 3e^{-3} \\
-5.71 & -1.59 & 8.35 & -1.04 & -1e^{-3} & 0 & 0 & -2e^{-3} & .02 & -.02 & 3e^{-5} & 6e^{-3} \\
1.05 & -8.11 & .05 & 8 & -1 & 0 & 0 & -3e^{-3} & .03 & -.04 & 7e^{-5} & .01 \\
0 & 1 & -8 & 0 & 8 & -1 & 0 & -7e^{-3} & .07 & -.09 & 2e^{-4} & .03 \\
0 & 0 & 1 & -8 & 0 & 8 & -1 & -.02 & .16 & -.20 & 3e^{-4} & .06 \\
0 & 0 & 0 & 1 & -8 & 0 & 8 & -1.03 & -.34 & -.43 & 7e^{-4} & .13 \\
0 & 0 & 0 & 0 & 1 & -8 & 0 & 7.93 & -.28 & -.94 & 2e^{-3} & .29 \\
0 & 0 & 0 & 0 & 0 & 1 & -8 & -.16 & 9.56 & -3.02 & 3e^{-3} & .62 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & -8.35 & 3.36 & 3.65 & -.99 & 1.33 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & .25 & -.76 & -8.37 & 6.02 & 2.86 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1.62 & 16.61 & -33.66 & 11.96 & 6.71 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & .53 & -5.47 & 2.78 & -3.27 & 5.43
\end{bmatrix}$$

7.1.2 Differentiation Matrix $(C_2^{2pt})^{-1}DC_2^{2pt}$, Grid $\vec{x}$

Now we use the quadrature matrix C_2^{2pt}. But, the results are similar to the previous matrix. The structure is unacceptable and the superconvergence is lost at the boundaries as can be seen from Table (7.1.3)

$$\mathcal{D} =$$

$$\begin{bmatrix}
-12.21 & 7.44 & 10.48 & -6.44 & .74 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-10.99 & -11.15 & 44.98 & -24.53 & 1.70 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-2.50 & -1.01 & -3.50 & 8 & -1 & 0 & 0 & 0 & 0 & .01 & -.01 & 0 \\
.46 & .07 & -7.54 & 0 & 8 & -1 & 0 & 0 & 0 & .01 & -.02 & .01 \\
0 & 0 & 1 & -8 & 0 & 8 & -1 & 0 & -.03 & .02 & -.04 & .02 \\
0 & 0 & 0 & 1 & -8 & 0 & 8 & -1 & -.01 & .05 & -.09 & .04 \\
0 & 0 & 0 & 0 & 1 & -8 & 0 & 8 & -1.01 & .11 & -.19 & .09 \\
0 & 0 & 0 & 0 & 0 & 1 & -8 & 0 & 7.97 & -.75 & -.41 & .19 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & -8 & -.06 & 8.53 & -1.89 & .04 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -8.13 & 1.14 & 6.09 & -.11 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & .73 & -5.53 & -3.12 & 7.93 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -.24 & 2.74 & -16.75 & 14.26
\end{bmatrix}$$

7.1.3 Conclusion for Grid $\vec{x}$

Recall that in the construction of the differentiation matrix that the middle matrix D is fixed. The Table (7.1.3) contains the accuracy for two well-conditioned quadrature matrices C_1^{2pt} and C_2^{2pt}. In addition, as an experiment I have combined two ill-conditioned matrices: one mapping from $\vec{f}$ to $\vec{s}$ and the other mapping from $\vec{s}$ to $\vec{f}$.

7 Order of Accuracy

In this section the order of accuracy for a few of the differentiation matrices which can be constructed from the derivative projection matrix D and the quadrature matrices constructed in the previous three sections will be found. We begin with the differentiation matrices constructed using grid $\vec{x}$ followed by the differentiation matrices constructed using grid $\vec{y}$.

7.1 Accuracy Using Grid $\vec{x}$

Recall that grid $\vec{x}$ is comprised of the points halfway between the integers and, consequently, the spacing between the grid points is the same as the translation distance for the scaling functions on the finest scale. Furthermore, this grid does not provide a grid point at the boundary itself. This lack of a grid point at the boundary makes this grid an unlikely choice, but I have chosen to study the accuracy as an experiment. Let us begin by giving a few explicit examples of differentiation matrices.

7.1.1 Differentiation Matrix $(C_1^{2pt})^{-1}DC_1^{2pt}$, Grid $\vec{x}$

Note the unusual form of the following matrix. It is essentially upper triangular in form. This form is due to the inversion of C_1^{2pt}. From Table (7.1.3) it can be seen that this differentiation matrix does not maintain the superconvergence at the boundaries. That is, the differentiation is 1st-order accurate at the boundaries, whereas it is 4th-order accurate at the middle 50% of the grid points. The total differentiation accuracy across the entire domain is, therefore, 2nd-order.

$$\mathcal{D} =$$

The numerical values in terms of the scaling function values on grid $\vec{y}$ are,

$$\begin{aligned}
\phi_0^L(8/7) &= \sqrt{2}(h_{0,1}^L\phi_1^L(16/7) + h_{0,2}^L\phi(2/7)) \\
\phi_1^L(8/7) &= \sqrt{2}(h_{1,1}^L\phi_1^L(16/7) + h_{1,2}^L\phi(2/7) + h_{1,3}^L\phi(-5/7)) \qquad (54) \\
\phi_1^L(16/7) &= \sqrt{2}(h_{1,3}^L\phi(11/7) + h_{1,4}^L\phi(4/7)).
\end{aligned}$$

As with grid $\vec{x}$ the quadrature matrix produced using grid $\vec{y}$ is ill-conditioned. Recall from the previous section that all 3-point quadrature matrices were ill-conditioned. I, therefore, conclude this section with the observation, not a proof, that quadrature matrices for wavelet bases on an interval using the construction in [3] should be constructed using moments and that the number of grid points used to approximate each scaling function coefficient should be equal to the number of vanishing moments of the corresponding wavelet.

$$\hat{C}_{\vec{x}} = \begin{bmatrix} .4124 & .8495 & 0 & 0 & 0 & 0 & 0 & 0 \\ .2062 & -.0077 & .9330 & 0 & 0 & 0 & 0 & 0 \\ 0 & .0669 & 0 & .9330 & 0 & 0 & 0 & 0 \\ 0 & 0 & .0670 & 0 & .9330 & 0 & 0 & 0 \\ 0 & 0 & 0 & .0670 & 0 & .9330 & 0 & 0 \\ 0 & 0 & 0 & 0 & .0670 & 0 & .8561 & 0 \\ 0 & 0 & 0 & 0 & 0 & .0670 & .3270 & .4451 \\ 0 & 0 & 0 & 0 & 0 & 0 & -.1406 & .8902 \\ & & & & & & & \end{bmatrix}. \qquad (53)$$

This matrix is ill-conditioned just as the 3-point quadrature formulas were for the quadrature matrix derived using moments.

6.2 A Quadrature Matrix based on Grid $\vec{y}$

Using the definition of the scaling function,

$$\phi(x) = \sqrt{2}\sum_{k=0}^{3} h_k \phi(2x-k),$$

one can set up a matrix such that the eigenvector for the eigenvalue $\lambda = 1$ gives a scalar multiple of the scaling function $\phi(x)$ at $x = i/7$ for $i = 1, ..., 20$. The eigenspace for the eigenvalue $\lambda = 1$ is, however, 2-dimensional. To avoid this complication one can approximate the numerical values of $\phi(x)$ at $x = i/7$ for $i = 1, ..., 20$ by generating $\phi(x)$ at a large number of diadic points, in this case 2^{14} points were chosen. This approximation is sufficient to understand the character of the quadrature matrix, specifically if it is well-conditioned or not. Once the numerical values of the scaling function are known then we can find the numerical values of the boundary functions.

6.2.1 The Numerical Values of the LHS Boundary Functions

As before, the numerical values of the LHS boundary functions on grid $\vec{y}$ can be found from,

$$\phi_k^L(x) = \sqrt{2}\sum_{l=0}^{1} h_{k,l}^L \phi_l^L(2x) + \sqrt{2}\sum_{m=2}^{2+2k} h_{k,m}^L \phi(2x-m).$$

6.1.1 The Numerical Values of the Scaling Function

Using the definition of the scaling function supported on $[0,3]$,

$$\phi(x) = \sqrt{2}\sum_{k=0}^{3} h_k \phi(2x-k),$$

we get

$$\phi(5/2) = \sqrt{2}h_3\phi(2) = .06698729811, \tag{49}$$

$$\phi(3/2) = \sqrt{2}(h_1\phi(2) + h_2\phi(1)), \tag{50}$$

and

$$\phi(1/2) = \sqrt{2}h_0\phi(1) = .9330127019. \tag{51}$$

The numerical values of ϕ at the integers are provided in [9]:

$$\phi(1) = 1/2(1+\sqrt{3}),$$

and

$$\phi(2) = 1/2(1-\sqrt{3}).$$

6.1.2 The Numerical Values of the LHS Boundary Functions

Given the numerical values of the scaling function $\phi(x)$ on grid $\vec{x}$ one can now find the numerical values of the boundary functions on grid $\vec{x}$ from the expression,

$$\phi_k^L(x) = \sqrt{2}\sum_{l=0}^{1} h_{k,l}^L \phi_l^L(2x) + \sqrt{2}\sum_{m=2}^{2+2k} h_{k,m}^L \phi(2x-m). \tag{52}$$

In a similar manner one can find the numerical values for the RHS boundary functions to get the quadrature matrix of the next subsection.

6.1.3 The Complete Quadrature Matrix for Grid $\vec{x}$

An 8 by 8 example of the quadrature matrix for grid $\vec{x}$ is,

6 Quadrature Matrix $\hat{C}$ Using Samples

Let $g(x)$ denote the projection of $f(x) \in L^2(R)$ in V_0:

$$g(x) = P_{V_0} f(x) = \sum_{k=1}^{N} s_k b_k(x). \tag{44}$$

The quadrature formula based on samples of the basis functions can be found from,

$$g(x_i) = P_{V_0} f(x_i) = \sum_{k=1}^{N} s_k b_k(x_i). \tag{45}$$

In matrix form the quadrature formula is,

$$\vec{g} = \hat{C}\vec{s}, \tag{46}$$

or let $\vec{\sigma}$ be the approximation to $\vec{s}$ to get,

$$\vec{f} = \hat{C}\vec{\sigma}, \tag{47}$$

where for an 8 by 8 case the quadrature matrix is,

$$\hat{C} = \begin{bmatrix} \phi_0^L(x_1) & \phi_1^L(x_1) & 0 & 0 & 0 & 0 & 0 & 0 \\ \phi_0^L(x_2) & \phi_1^L(x_2) & \phi_2(x_2) & 0 & 0 & 0 & 0 & 0 \\ 0 & \phi_1^L(x_3) & \phi_2(x_3) & \phi_3(x_3) & 0 & 0 & 0 & 0 \\ 0 & 0 & \phi_2(x_4) & \phi_3(x_4) & \phi_4(x_4) & 0 & 0 & 0 \\ 0 & 0 & 0 & \phi_3(x_5) & \phi_4(x_5) & \phi_5(x_5) & 0 & 0 \\ 0 & 0 & 0 & 0 & \phi_4(x_6) & \phi_5(x_6) & \phi_1^R(x_6) & 0 \\ 0 & 0 & 0 & 0 & 0 & \phi_5(x_7) & \phi_1^R(x_7) & \phi_0^R(x_7) \\ 0 & 0 & 0 & 0 & 0 & 0 & \phi_1^R(x_8) & \phi_0^R(x_8) \\ \end{bmatrix} \tag{48}$$

Now $\hat{C}$ will be found for the two previously defined grids $\vec{x}$ and $\vec{y}$.

6.1 A Quadrature Matrix based on Grid $\vec{x}$

Finding the explicit form of the above defined $\hat{C}$ for the grid $\vec{x}$ is relatively simple since we only need the numerical values of the scaling functions and boundary functions halfway between the integers.

5.2.4 A First Option Using a 2-Point Quadrature

$$C_1^{2pt} = \begin{bmatrix} c_{1,1} & c_{1,2} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & c_{2,2} & c_{2,3} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & c_{3,3} & c_{3,4} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & c_{4,4} & c_{4,5} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & c_{5,5} & c_{5,6} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & c_{6,6} & c_{6,7} & 0 \\ 0 & 0 & 0 & 0 & 0 & c_{7,6} & c_{7,7} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & c_{8,7} & c_{8,8} \end{bmatrix} \qquad (42)$$

This quadrature matrix is well-conditioned for both grids $\vec{x}$ and $\vec{y}$.

5.2.5 Second Option for 2-Point Quadrature

$$C_2^{2pt} = \begin{bmatrix} c_{1,1} & c_{1,2} & 0 & 0 & 0 & 0 & 0 & 0 \\ c_{2,1} & c_{2,2} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & c_{3,3} & c_{3,4} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & c_{4,4} & c_{4,5} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & c_{5,5} & c_{5,6} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & c_{6,6} & c_{6,7} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & c_{7,7} & c_{7,8} \\ 0 & 0 & 0 & 0 & 0 & 0 & c_{8,7} & c_{8,8} \end{bmatrix} \qquad (43)$$

This quadrature matrix is, also, well-conditioned for grids $\vec{x}$ and $\vec{y}$.

5.3 Conclusion

In this section a variety of structures for quadrature matrices have been shown where the matrices are found by using the moments of the scaling function. In general, the pattern is that 3-point quadrature matrices are ill-conditioned and 2-point quadrature matrices are well-conditioned.

The next section will explore quadrature matrices constructed from the samples of the scaling functions.

5.2.2 All Grid Points Contained in Scaling function Support

As a first possible choice of grid points, let us use every grid point that is contained within the support of the respective basis function. Such a quadrature matrix will the following structure:

$$C_1^{3pt} = \begin{bmatrix} c_{1,1} & c_{1,2} & 0 & 0 & 0 & 0 & 0 & 0 \\ c_{2,1} & c_{2,2} & c_{2,3} & 0 & 0 & 0 & 0 & 0 \\ 0 & c_{3,2} & c_{3,3} & c_{3,4} & 0 & 0 & 0 & 0 \\ 0 & 0 & c_{4,3} & c_{4,4} & c_{4,5} & 0 & 0 & 0 \\ 0 & 0 & 0 & c_{5,4} & c_{5,5} & c_{5,6} & 0 & 0 \\ 0 & 0 & 0 & 0 & c_{6,5} & c_{6,6} & c_{6,7} & 0 \\ 0 & 0 & 0 & 0 & 0 & c_{7,6} & c_{7,7} & c_{7,8} \\ 0 & 0 & 0 & 0 & 0 & 0 & c_{8,7} & c_{8,8} \end{bmatrix} \tag{40}$$

For both of the grids $\vec{x}$ and $\vec{y}$ the matrix C_1^{3pt} is ill-conditioned. The condition number is proportional to N^2, where N is the number of grid points, and inversion of C_1^{3pt} unacceptably corrupts the differentiation matrix.

5.2.3 2-Point Quadrature at Boundary and 3-Point Otherwise

The quadrature matrix has the following structure:

$$C_2^{3pt} = \begin{bmatrix} c_{1,1} & c_{1,2} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & c_{2,2} & c_{2,3} & 0 & 0 & 0 & 0 & 0 \\ 0 & c_{3,2} & c_{3,3} & c_{3,4} & 0 & 0 & 0 & 0 \\ 0 & 0 & c_{4,3} & c_{4,4} & c_{4,5} & 0 & 0 & 0 \\ 0 & 0 & 0 & c_{5,4} & c_{5,5} & c_{5,6} & 0 & 0 \\ 0 & 0 & 0 & 0 & c_{6,5} & c_{6,6} & c_{6,7} & 0 \\ 0 & 0 & 0 & 0 & 0 & c_{7,6} & c_{7,7} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & c_{8,7} & c_{8,8} \end{bmatrix} \tag{41}$$

This matrix is, also, ill-conditioned for both grids $\vec{x}$ and $\vec{y}$. Again, the condition number if proportional to N^2.

corresponding to the following two grids:

$$\vec{x} = \begin{bmatrix} .5 \\ 1.5 \\ 2.5 \\ 3.5 \\ 4.5 \\ 5.5 \\ 6.5 \\ 7.5 \end{bmatrix}, \tag{37}$$

and

$$\vec{y} = \begin{bmatrix} 0 \\ 8/7 \\ 2*8/7 \\ 3*8/7 \\ 4*8/7 \\ 5*8/7 \\ 6*8/7 \\ 8 \end{bmatrix}. \tag{38}$$

5.2.1 The Entries of the Quadrature Matrices

Now, one must use the scaling function and boundary function moments and at least two grid points contained within the support of each basis function in order to build the quadrature matrix. The first row of the quadrature matrix will estimate the basis function coefficient for the boundary function with the smallest support, ϕ_1^{bd}. Suppose we have chosen grid $\vec{x}$. We know that for the D_4 wavelet that constants and lines can be represented exactly in the wavelet subspace V_0. There are only two grid points within the support of ϕ_1^{bd}. Therefore, the solution to the following linear system will yield the two coefficients $c_{1,1}$ and $c_{1,2}$ in the first row of the quadrature matrix. The linear system is,

$$\begin{bmatrix} \int_I \phi_1^{bd}(x)dx \\ \int_I \phi_1^{bd}(x)xdx \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ .5 & 1.5 \end{bmatrix} \begin{bmatrix} c_{1,1} \\ c_{1,2} \end{bmatrix}. \tag{39}$$

In a similar manner all the entries of a quadrature matrix can be found once one has calculated the moments of the basis functions and chosen the grid points. The following subsection will illustrate a few of the quadrature matrices associated with various grid choices.

5.2 The Choice of Grid Points

In this subsection the affect of the choice of different grid points on the quadrature matrix will be illustrated. For simplicity, in this subsection we will work with only 8 bases functions and 8 grid points. The interval will be $[0, 8]$. When choosing our grid points to approximate the scaling function coefficient s_i we must use only the grid points which lie within the support of the basis function $b_i(x)$. That is, the inner product that is being approximated is,

$$s_i = \int_0^8 b_i(x) f(x) dx,$$

and the data that is available to approximate s_i must be limited to function values defined within the support of $b_i(x)$. The support of each of the eight bases functions is,

$$supp\{\phi_1^{bd}\} = [0, 2],$$

$$supp\{\phi_2^{bd}\} = [0, 3],$$

$$supp\{\phi_3\} = [1, 4],$$

$$supp\{\phi_4\} = [2, 5],$$

$$supp\{\phi_5\} = [3, 6],$$

$$supp\{\phi_6\} = [4, 7],$$

$$supp\{\phi_7^{bd}\} = [5, 8],$$

$$supp\{\phi_8^{bd}\} = [6, 8],$$

where the superscript '*bd*' denotes a boundary function.

The following subsections will consider a few of the possible quadrature matrices

5 Quadrature Matrix C Using Moments

The quadrature matrix C maps from evenly-spaced point values, $f(x_i)$, of a function to the approximate scaling function coefficients at the finest scale, $\vec{s}$. The quadrature matrix depends on three quantities: the number of vanishing moments of the wavelet, the grid that the function $f(x)$ is defined on, and which grid points are used to approximate each scaling function coefficient.

5.1 Moments of Scaling and Boundary Functions

The moments of the D_4 scaling functions were calculated in appendix A of [6]. The first 3 moments are, $M_0 = 1$, $M_1 = .6339745962$, and $M_2 = .4019237886$.

We will concentrate here on calculating the moments of the boundary functions.

5.1.1 Moment 0 of Left-Hand Side Boundary Functions

The 0-th moment of the boundary function k is,

$$\int_0^\infty \phi_k^L(x)dx = \sqrt{2}\sum_{l=0}^{1} h_{k,l}^L \int_0^\infty \phi_l^L(2x) + \sqrt{2}\sum_{m=2}^{2+2k} h_{k,m}^L \int_0^\infty \phi(2x-m). \tag{34}$$

Now, adopt the notation $m_k^0 = \int \phi_k^L(x)dx$ and let $y = 2x$ where appropriate in the above integrals to get,

$$\sqrt{2}m_k^0 = \sum_{l=0}^{1} h_{k,l}^L m_l^0 + \sum_{m=2}^{2+2k} h_{k,m}^L \int_{-m}^\infty \phi(x)dx. \tag{35}$$

The integral $\int_{-m}^\infty \phi(x)dx$ is 1 since $m \geq 2$, and the scaling function, as defined here, is supported on $[-1, 2]$. Equation (35) defines the following 2 by 2 system:

$$\begin{bmatrix} h_{0,0}^L - \sqrt{2} & h_{0,1}^L \\ h_{1,0}^L & h_{1,1}^L - \sqrt{2} \end{bmatrix} \begin{bmatrix} m_0^0 \\ m_1^0 \end{bmatrix} = \begin{bmatrix} -h_{0,2}^L \\ -h_{1,2}^L - h_{1,3}^L - h_{1,4}^L \end{bmatrix}. \tag{36}$$

The solution to the above system is

$$m_0^0 = .3620520589,$$

$$m_1^0 = 1.001445402.$$

In a similar fashion all moments can be found for the left and right hand side boundary functions.

The second type of quadrature matrix is constructed from samples of the scaling function and boundary functions and is represented by,

$$\vec{f} = \hat{C}\vec{s}. \tag{33}$$

It will be seen that this type of quadrature matrix is always ill-conditioned.

4 Approximating Scaling Function Coefficients

Once again, let us work with only the D_4 wavelet. Recall, as constructed in [3], there are two boundary functions at each end of the interval, as well as, the usual D_4 scaling functions away from the boundaries. Let N denote the number of degrees-of-freedom of the approximation space, V_0. Our function $f(x) \in L^2(R)$ is then discretized as follows (In order to simplify the presentation I will let the 'interval' be $[0, N]$, and there will be no overlap between left-hand boundary functions and right-hand boundary functions.)

First, the left-hand side,

$$s_i^L = \int_0^N \phi_i^L(x) f(x) dx, \tag{29}$$

then the right-hand side,

$$s_i^R = \int_0^N \phi_i^R(x) f(x) dx, \tag{30}$$

where $i = 0, 1$. Next, discretize in the middle with the usual scaling functions,

$$s_k = \int_0^N \phi(x-k) f(x) dx, \tag{31}$$

for $k = 2, ..., N-4$.

Now we will assume that $f(x)$ is defined on a discrete evenly-spaced grid, $f(x_i)$ for $i = 1, ..., N$. We must, therefore, approximate the above three integrals by an appropriate quadrature matrix.

In the following two sections two types of quadrature matrices will be considered. The first type is constructed using the moments of the scaling function and is represented by,

$$\vec{s} = C\vec{f}. \tag{32}$$

It will be seen that two-point quadrature matrices for this method are well-conditioned, whereas three-point quadrature matrices are ill-conditioned.

$$\begin{bmatrix}
-1.9038 & .9444 & -.2565 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1.5163 & -.0430 & .6752 & -.0832 & 0 & 0 & 0 & 0 & 0 & 0 \\
.2565 & -.6752 & 0 & \frac{2}{3} & -\frac{1}{12} & 0 & 0 & 0 & 0 & 0 \\
0 & .0832 & -\frac{2}{3} & 0 & \frac{2}{3} & -\frac{1}{12} & 0 & 0 & 0 & 0 \\
0 & 0 & \frac{1}{12} & -\frac{2}{3} & 0 & \frac{2}{3} & -\frac{1}{12} & 0 & 0 & 0 \\
0 & 0 & 0 & \frac{1}{12} & -\frac{2}{3} & 0 & \frac{2}{3} & -\frac{1}{12} & 0 & 0 \\
0 & 0 & 0 & 0 & \frac{1}{12} & -\frac{2}{3} & 0 & \frac{2}{3} & -.0765 & 0 \\
0 & 0 & 0 & 0 & 0 & \frac{1}{12} & -\frac{2}{3} & 0 & .5825 & -.0397 \\
0 & 0 & 0 & 0 & 0 & 0 & .0765 & -.5825 & .0899 & .3150 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & .0397 & -.7936 & .6369
\end{bmatrix}.$$

Note that, as expected, away from the boundaries D is the same as the derivative projection matrix which was found for periodic boundary conditions, see [6]. Also, note D is not antisymmetric, but is banded. D is not in itself the differentiation matrix, but D is the component of the differentiation matrix, $\mathcal{D} = C^{-1}DC$, which does the differencing. Later in this paper it will become obvious how one can simply 'look' at D in order to tell the maximum differentiation accuracy at the boundaries. The next section begins the discussion of choosing a quadrature matrix, C.

In equation (11) let $k = 1$ and $x = 2$ to get,

$$\phi_1^L(2) = \sqrt{2}(h_{1,3}^L \phi(1) + h_{1,4}^L \phi(0)). \tag{27}$$

The values of the scaling function $\phi(x)$ at the above integers are $\phi(0) = 1/2(1+\sqrt{3})$ and $\phi(1) = 1/2(1-\sqrt{3})$, see [9]. Using these numbers and the coefficients provided in [3] we get $\phi_1^L(2) = -.3654971039$. Similar to above, let $k = 0$ and $x = 1$ in equation (11) to get

$$\phi_0^L(1) = \sqrt{2}h_{0,1}^L \phi_1^L(2) + \sqrt{2}h_{0,2}^L \phi(0), \tag{28}$$

and evaluate to get $\phi_0^L(1) = -1.1265992786$. In this way, one can get the following values for the boundary functions: $\phi_1^L(1) = 1.4058555636$, $\phi_0^L(1/2) = .4123639530$. Now we have two values for $\phi_0^L(x)$ at $x = 1/2$ and $x = 1$. Furthermore, the two boundary functions are lines between $x = 0$ and $x = 1$. We can, therefore, find the value of $\phi_0^L(x)$ at $x = 0$: $\phi_0^L(0) = 1.9513271847$. As outlined in the previous subsection we now have sufficient information to find $\rho_{0,0}^L$ and $\rho_{1,1}^L$:

$$\rho_{0,0}^L = -1.9038388908,$$

and

$$\rho_{1,1}^L = -.04295212263.$$

The calculation of the remaining elements of the derivative projection matrix is straightforward and will not be given. The next subsection provides the complete matrix D.

3.3 The Complete Matrix D

This section of the paper will be concluded by giving a 10 by 10 version of the matrix D. Only four digit accuracy will be given for the corner entries ρ, α, and β. The entries r are known exactly.

$$D =$$

$\phi_k^L(x)$ is 0 at $x = \infty$ leaving only the value at the lower limit,

$$\rho_{k,k}^L = -\frac{(\phi_k^L(0))^2}{2}. \tag{21}$$

That is, $\rho_{k,k}^L$ is determined by the value of the boundary function at the boundary. Before evaluating the boundary functions at the boundary it will be shown that $\rho_{0,0}^L$ and $\rho_{1,1}^L$ are related by a constant multiple.

3.2.2 The Relationship Between $\rho_{0,0}^L$ and $\rho_{1,1}^L$

In equation (11) set $x = 0$ to get,

$$\phi_k^L(0) = \sqrt{2}\sum_{l=0}^{1} h_{k,l}^L \phi_l^L(0) + \sqrt{2}\sum_{m=2}^{2+2k} h_{k,m}^L \phi(-m). \tag{22}$$

For $k = 0$ we get,

$$\phi_0^L(0) = \sqrt{2}(h_{0,0}^L \phi_0^L(0) + h_{0,1}^L \phi_1^L(0)), \tag{23}$$

which yields,

$$\phi_1^L(0) = c^L \phi_0^L(0), \tag{24}$$

where

$$c^L = \frac{1 - \sqrt{2}h_{0,0}^L}{\sqrt{2}h_{0,1}^L}. \tag{25}$$

Recall from above that the $\rho_{k,k}^L$'s can be found from the boundary functions evaluated at the boundary. Combine this information with equation (24) to get,

$$\rho_{1,1}^L = (c^L)^2 \rho_{0,0}^L. \tag{26}$$

That is, we only need to evaluate one of the boundary functions at the boundary in order to know $\rho_{0,0}^L$ and $\rho_{1,1}^L$. The following subsection will evaluate the boundary function $\phi_0^L(x)$ at $x = 0$.

3.2.3 Evaluation of the Boundary Function $\phi_0^L(x)$ at $x = 0$.

In this subsection the straightforward evaluation of the boundary function $\phi_0^L(x)$ at $x = 0$ will be given.

Rename the integrals as,

$$\begin{aligned}
\rho^L_{k,p} &= \int_I \acute{\phi}^L_k(x)\phi^L_p(x)dx, && (14)\\
\alpha^L_{m,i} &= \int_I \acute{\phi}(x-m)\phi^L_i(x)dx, && (15)\\
\beta^L_{l,q} &= \int_I \acute{\phi}^L_l(x)\phi(x-q)dx, && (16)\\
r_{m,q} &= \int_I \acute{\phi}(x-m)\phi(x-q)dx, && (17)
\end{aligned}$$

to get the following system of equations:

$$\begin{aligned}
1/2\rho^L_{k,p} &= \sum_{l}^{N-1}\sum_{i}^{N-1} h^L_{k,l}h^L_{p,i}\rho^L_{l,i} + \sum_{i=0}^{N-1}\sum_{m=N}^{N+2k} h^L_{p,i}h^L_{k,m}\alpha^L_{m,i} + \\
&\sum_{l=0}^{N-1}\sum_{q=N}^{N+2p} h^L_{k,l}h^L_{p,q}\beta^L_{l,q} + \sum_{m=N}^{N+2k}\sum_{q=N}^{N+2p} h^L_{k,m}h^L_{p,q}r_{m,q}. && (18)
\end{aligned}$$

The numerical values of the coefficients $\{r\}$ were found in [2], and the numerical values of the coefficients ρ^L, α^L, and β^L will be found in the following subsections.

3.2 The Coefficients $\rho^L_{i,i}$, for $i = 0, 1$

In this subsection we will find the corner diagonal elements of D. These elements represent the projection of the derivative of each boundary function onto itself.

3.2.1 The Coefficients $\rho^L_{0,0}$ and $\rho^L_{1,1}$

For simplicity, allow the lower limit to be 0 and the upper limit be ∞ in the definition of $\rho^L_{k,k}$:

$$\rho^L_{k,k} = \int_0^\infty \acute{\phi}^L_k(x)\phi^L_k(x)dx.$$

Apply integration by parts to this equation to get,

$$\rho^L_{k,k} = (\phi^L_k(x))^2|^\infty_0 - \int_0^\infty \phi^L_k(x)\acute{\phi}^L_k(x)dx, \qquad (19)$$

or,

$$2\rho^L_{k,k} = (\phi^L_k(x))^2|^\infty_0. \qquad (20)$$

projected onto boundary functions, and the coefficients α and β contain the interactions between derivatives of boundary functions and scaling functions.

In the following subsections the entries of D will be found.

3.1 The Equations for the Derivative Projections

To find the elements of D one needs to find the interaction of the derivative of each basis function with every other basis function. Due to the local support of the bases functions, the left-hand side boundary functions and the right-hand boundary functions due not interact. (Assuming, of course, that one does not choose an unreasonably small domain relative to the support of the scaling functions.)

Each entry in the matrix D must be found separately, but the calculations are straightforward. A sampling of the relevant derivations will be given below.

3.1.1 Derivation of the LHS Equations

Recall that the LHS boundary functions $\phi_k^L(x)$ are generated by,

$$\phi_k^L(x) = \sqrt{2}\sum_{l=0}^{N-1} h_{k,l}^L \phi_l^L(2x) + \sqrt{2}\sum_{m=N}^{N+2k} h_{k,m}^L \phi(2x-m). \tag{11}$$

Differentiate the above equation to get,

$$\frac{d}{dx}\phi_k^L(x) = 2\sqrt{2}\sum_{l=0}^{N-1} h_{k,l}^L \phi_l'^L(2x) + 2\sqrt{2}\sum_{m=N}^{N+2k} h_{k,m}^L \phi'(2x-m). \tag{12}$$

Multiply $\phi_k'^L(x)$ by $\phi_p^L(x)$ and integrate to get,

$$\begin{aligned}\int_I \phi_k'^L(x)\phi_p^L(x)dx &= 4\sum_{l=0}^{N-1}\sum_{i=0}^{N-1} h_{k,l}^L h_{p,i}^L \int_I \phi_l'^L(2x)\phi_i^L(2x)dx \\ &+ 4\sum_{m=N}^{N+2k}\sum_{i=0}^{N-1} h_{p,i}^L h_{k,m}^L \int_I \phi_i^L(2x)\phi'(2x-m)dx \\ &+ 4\sum_{l=0}^{N-1}\sum_{q=N}^{N+2p} h_{k,l}^L h_{p,q}^L \int_I \phi_l'^L(2x)\phi(2x-q)dx \\ &+ 4\sum_{m=N}^{N+2k}\sum_{q=N}^{N+2p} h_{k,m}^L h_{p,q}^L \int_I \phi'(2x-m)\phi(2x-q)dx.\end{aligned} \tag{13}$$

where $\acute{b}(x) = \frac{d}{dx}b(x)$. The derivative takes $P_{V_0}f(x)$ out of V_0. Projecting back into V_0 we get,

$$P_{V_0}\frac{d}{dx}P_{V_0}f(x) = \sum_{l=0}^{d-1} < \frac{d}{dx}P_{V_0}f, b_l > b_l(x), \tag{9}$$

or,

$$P_{V_0}\frac{d}{dx}P_{V_0}f(x) = \sum_{l=0}^{d-1}\sum_{k=0}^{d-1} s_k < \acute{b}_k, b_l > b_l(x). \tag{10}$$

The elements $< \acute{b}_k, b_l >$ comprise the derivative projection matrix D. If we let $\vec{s}$ denote the vector of the scaling function coefficients s_k, for $k = 0, ..., d-1$, then D maps from the scaling function coefficients of a function at the finest scale to the scaling function coefficients of the derivative of the same function:

$$D : \vec{s} \rightarrow \vec{\acute{s}}.$$

D, in effect, contains all the derivative information in a wavelet expansion. The elements of D contain the numerical values of the derivative of each scaling function projected onto all scaling functions in the basis. Due to the compact support of the bases functions, the matrix D has a band diagonal form. The following matrix illustrates the form of D for the D_4 wavelet basis with boundary functions defined as in the previous section:

$$D = \begin{bmatrix}
\rho^L_{0,0} & \rho^L_{1,0} & \alpha^L_{2,0} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\rho^L_{0,1} & \rho^L_{1,1} & \alpha^L_{2,1} & \alpha^L_{3,1} & 0 & 0 & 0 & 0 & 0 & 0 \\
\beta^L_{0,2} & \beta^L_{1,2} & r_0 & r_1 & r_2 & 0 & 0 & 0 & 0 & 0 \\
0 & \beta^L_{1,3} & r_{-1} & r_0 & r_1 & r_2 & 0 & 0 & 0 & 0 \\
0 & 0 & r_{-2} & r_{-1} & r_0 & r_1 & r_2 & 0 & 0 & 0 \\
0 & 0 & 0 & r_{-2} & r_{-1} & r_0 & r_1 & r_2 & 0 & 0 \\
0 & 0 & 0 & 0 & r_{-2} & r_{-1} & r_0 & r_1 & \beta^R_{1,4} & 0 \\
0 & 0 & 0 & 0 & 0 & r_{-2} & r_{-1} & r_0 & \beta^R_{1,5} & \beta^R_{0,5} \\
0 & 0 & 0 & 0 & 0 & 0 & \alpha^R_{4,1} & \alpha^R_{5,1} & \rho^R_{1,1} & \rho^R_{0,1} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \alpha^R_{5,0} & \rho^R_{1,0} & \rho^R_{0,0}
\end{bmatrix},$$

where the coefficients r are the derivative projections due to scaling function interaction, the coefficients ρ are the projections of the derivatives of boundary functions

3 The Derivative Projection Matrix D

Let us recall the origin of the matrix D. First, we begin with a function $f(x) \in L^2(R)$. Next, we want to approximate, or discretize, this function in a scaling function subspace $V_0 \subset L^2(R)$,

$$P_{V_0} f(x) = \sum_{k=0}^{d-1} s_k b_k(x), \tag{5}$$

where $\{b_k(x)\}$ denotes the bases functions of this finest scale discretization and d is the dimension or number of degrees-of-freedom of the subspace. In the case of wavelets with periodic boundary conditions $b_k(x) = \phi_k(x)$, the usual Daubechies scaling functions, for all $k \in \{0, ..., d-1\}$. In the current scenario, however, of wavelets on an interval $b_k(x)$ is different at the boundaries:

$$b_k(x) = \phi_k^L(x)$$

for $k = 0, ..., N-1$,

$$b_k(x) = \phi_k(x)$$

for $N \leq k \leq d - 2N + 1$ and,

$$b_k(x) = \phi_k^R(x)$$

for $k = d - 2N + 2, ...d - 1$, where N is the number of vanishing moments of the wavelet. Under the orthonormality of the current basis set, $< b_k, b_l >= \delta_{k,l}$, where,

$$< g, h >\equiv \int_I g(x)h(x)dx, \tag{6}$$

and $\int_I$ denotes integration over the interval, we get,

$$s_k =< b_k, f > . \tag{7}$$

We now differentiate the approximation $P_{V_0} f(x)$ to get,

$$\frac{d}{dx} P_{V_0} f(x) = \sum_{k=0}^{d-1} s_k \acute{b}_k(x), \tag{8}$$

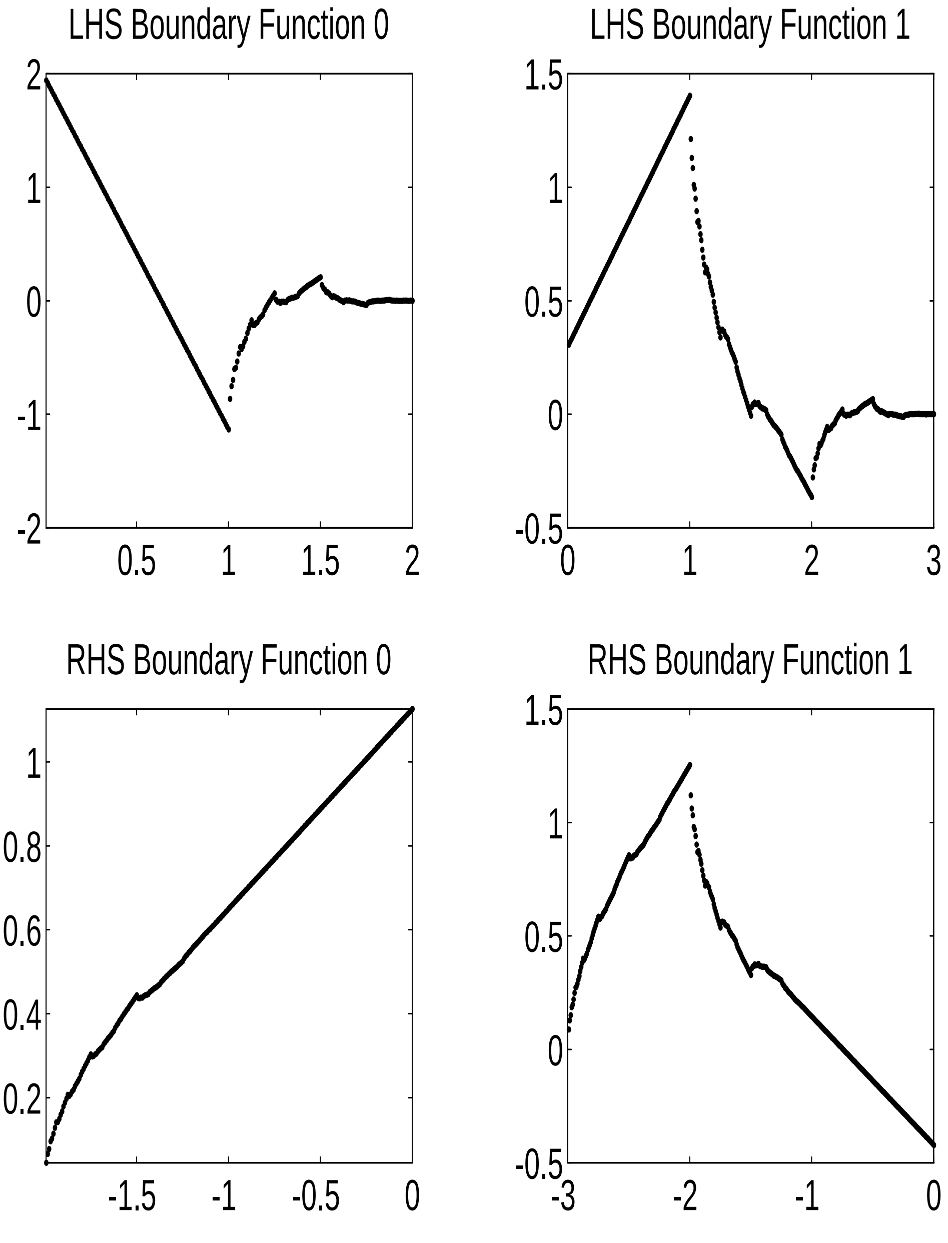

Figure 1: The boundary functions for the Daubechies 4 scaling function.

2 Construction of Boundary Functions

This section is a brief outline of the boundary function construction proposed by Cohen, Daubechies, and Vial, see [4].

The goal is to build a wavelet basis on an interval where the scaling functions and wavelets away from the boundaries are the usual Daubechies scaling functions and wavelets. At the boundaries, boundary scaling functions are constructed such that polynomials of order up to the number of vanishing moments of the wavelet can be reproduced exactly across the entire interval.

The boundary function construction begins by building independent, but not orthogonal, functions as follows:

$$\hat{\phi}^k(x) = \sum_{n=k}^{2N-2} \begin{pmatrix} n \\ k \end{pmatrix} \phi(x+n-N+1), \tag{1}$$

where $\phi(x)$ is the usual Daubechies scaling function and N is the number of vanishing moments of the associated wavelet. Note that these functions are compactly supported, and their supports are staggered, i.e.,

$$supp(\hat{\phi}^k) = [0, 2N-1-k]. \tag{2}$$

The staggered support yields independence, and the boundary functions are then defined by simply orthonormalizing these functions by a Gram-Schmidt method.

Coefficients h^L and h^R can be found, see [4], such that the boundary functions are defined recursively as follows beginning with the left-hand side boundary functions:

$$\phi_k^L(x) = \sqrt{2}\sum_{l=0}^{N-1} h_{k,l}^L \phi_l^L(2x) + \sqrt{2}\sum_{m=N}^{N+2k} h_{k,m}^L \phi(2x-m), \tag{3}$$

and the equation defining the right-hand side boundary functions is,

$$\phi_k^R(x) = \sqrt{2}\sum_{l=0}^{N-1} h_{k,l}^R \phi_l^R(2x) + \sqrt{2}\sum_{m=N}^{N+2k} h_{k,m}^R \phi(2x+m+1). \tag{4}$$

The following figure is a plot of the four boundary scaling functions for the Daubechies 4 wavelet.

periodic boundary conditions are assumed. By simply counting the overlap of boundary functions in other boundary constructions it can be seen that all currently available boundary constructions fail to maintain the superconvergence encountered under periodic boundary conditions.

the number of nonzeros entries in the first few and last few rows of D are related to the overlap of the boundary functions.

4. **Approximating Scaling Function Coefficients:**

   An outline of scaling function coefficient approximation is given.

5. **Quadrature Matrix C Using Moments:**

   C performs the mapping, $C : \vec{f} \rightarrow \vec{s}$. The D_4 wavelet has two vanishing moments which yields well-conditioned 2-point quadrature matrices and ill-conditioned 3-point quadrature matrices.

6. **Quadrature Matrix $\hat{C}$ Using Samples:**

   $\hat{C}$ performs the mapping, $\hat{C} : \vec{s} \rightarrow \vec{f}$. For the D_4 wavelet the matrix $\hat{C}$ has three entries in each row and is banded. $\hat{C}$ is, however, ill-conditioned.

7. **Examples and Order of Accuracy:**

   The matrix D is fixed. There are, however, many choices for the quadrature matrix producing many versions of $\mathcal{D}$. In all cases superconvergence is lost at the boundaries. Furthermore, choosing among the quadrature matrices which not ill-conditioned one gets a differentiation matrix which is essentially full. A banded $\mathcal{D}$ can be constructed from a combination of the two types of quadrature matrices C and $\hat{C}$. The bandwidth from such a construction is large, 9, for a D_4 wavelet and, again, the superconvergence is lost at the boundaries. In addition, all of these matrices display directional dependency. This directional dependency can be 'fixed' by reversing the coordinate system and averaging the appropriate differentiation matrices.

8. **Conclusion:**

   The differentiation matrices for wavelets on an interval using the boundary functions defined in [3] do not maintain the superconvergence encountered when with other boundary constructions.

We begin by defining the differentiation matrix, $\mathcal{D}$, as the product of three matrices: $\mathcal{D} = \hat{C}DC$. The three matrices are defined as follows:

- The first matrix C is a quadrature matrix mapping from samples in the physical space to approximate scaling function coefficients at the finest scale: $\vec{s} = C\vec{f}$. C can be derived from the moments of the scaling function. C, also, can simply be the inverse of the third matrix: $C = (\hat{C})^{-1}$.

- The second matrix D maps from the scaling function coefficients of a function to the scaling function coefficients of the derivative of the same function. For convenience I will, henceforth, refer to this matrix as the 'derivative projection' matrix.

- The third matrix $\hat{C}$ is a quadrature matrix mapping from approximate scaling function coefficients to approximate point values in the physical space: $\vec{f} = \hat{C}\vec{s}$. $\hat{C}$ can be derived from samples of the scaling function or can simply be the inverse of the first matrix: $\hat{C} = C^{-1}$.

This paper contains the following sections.

1. **Introduction**

2. **Boundary Functions:**

   An outline of the boundary functions constructed in [3] is given.

3. **Derivative Projection Matrix D:**

   The entries of D are defined and calculated. D has a banded structure due to the local support of the basis functions, but D is not antisymmetric. The number of nonzero entries in each row of D bounds the maximum differentiation accuracy for the corresponding row in the differentiation matrix. Furthermore,

1 Introduction

Superconvergence is a property of wavelet methods, Daubechies, see [4], and spline, see [10], based, when periodic boundary conditions are assumed. That is, for Daubechies-based methods under the assumption of periodicity it was proven in [6] that the differentiation matrix is accurate of order $2M$, even though the highest degree polynomial which can be reconstructed exactly in the wavelet subspace is of degree $M-1$, where M is the number of vanishing moments of the wavelet. For spline-based methods, again assuming periodicity, it was proven in [7] that the differentiation matrix is accurate of order $2n+2$, even though one can only construct splines of order n exactly in the underlying subspace. For Daubechies and spline-based wavelet systems this, roughly, doubling of the differentiation accuracy is known as superconvergence, a name borrowed from the finite element literature.

When periodicity is no longer assumed and one defines wavelets on an interval one of the goals when building boundary functions should be to maintain the superconvergence of the differentiation matrix across the entire interval. It has been proven by Gottlieb, et. al. [5], that the superconvergence encountered in finite element methods under the assumption of periodicity can not be maintained on the interval if there are characteristics leaving the domain. Furthermore, it was shown in [7] that for spline-based wavelet systems that superconvergence is lost at the boundaries when the boundary functions are truncated B-splines. The goal of this paper is to explore the accuracy of the differentiation matrix for Daubechies-based wavelet systems defined on an interval. In particular, the differentiation accuracy of the differentiation matrices constructed from the currently available boundary constructions for Daubechies wavelets defined on an interval will be explored. The differentiation matrix for the Daubechies 4 wavelet basis using the boundary functions defined in [3] will be calculated explicitly and the accuracy will be found. From this explicit example one can see a necessary condition for superconvergence to be maintained

ON THE DIFFERENTIATION MATRIX FOR DAUBECHIES-BASED WAVELETS ON AN INTERVAL

Leland Jameson[1]

Institute for Computer Applications in Science and Engineering

NASA Langley Research Center

Hampton, VA 23681

ABSTRACT

The differentiation matrix for a Daubechies-based wavelet basis defined on an interval will be constructed. It will be shown that the differentiation matrix based on the currently available boundary constructions does not maintain the superconvergence encountered under periodic boundary conditions.

[1] This research was supported by the National Aeronautics and Space Administration under NASA Contract No. NAS1-19480 while the author was in residence at the Institute for Computer Applications in Science and Engineering (ICASE), NASA Langley Research Center, Hampton, VA 23681. Research was also supported by AFOSR grant 93-1-0090, by DARPA grant N00014-91-4016, and by NSF grant DMS-9211820.