Wavelets and their application for the solution of partial differential equations in physics

Stefan Goedecker,
Max-Planck Institute for Solid State Research,
Stuttgart, Germany
goedeck@prr.mpi-stuttgart.mpg.de

May 20, 2009

Foreword

This book is based on a postgraduate course given by the author at EPFL in January/February 1998. The motivation for teaching this course as well as for writing this book was to make this fascinating and highly useful field of wavelets accessible to non-mathematicians. The overwhelming part of the literature on wavelets is in the classical mathematical style of theorems followed by proofs. The threshold for entering the subject for non-mathematicians is therefore rather high. In addition computational scientists, who just want to apply this theory, are more interested in an intuitive understanding of the important features instead of the formal mathematical framework. This book is intended to make the theory of wavelets understandable to this audience. In addition to a self-contained and intuitive presentation of the theory of wavelets, extensive tables with the basic filter coefficients of differential operators in several wavelet families can be found in this book. After working through this book anyone who wants to numerically solve partial differential equations in physics, chemistry or engineering using wavelets should be able to do so. Wavelets are a basis set with extraordinary properties for the solution of differential equations. Their flexibility and efficiency allows us to attack problems which are very hard or even impossible to tackle with conventional methods. It is to be expected that the theory of wavelets will soon be part of any science and engineering curriculum in the same way as Fourier analysis is nowadays.

I thank F. Lévy and K. Maschke for their encouragement to develop this course and publish it as a book as well as the "Troisième Cycle de la Physique en Suisse Romande" to invite me to deliver this series of lectures. I also want to thank Oleg Ivanov, with whom I started this exciting research on wavelets when he was visiting the Max-Planck Institute in 1996/97. Anna Putrino helped me to eliminate various errors in the manuscript.

Contents

1 Wavelets, an optimal basis set

The prefered way to solve partial differential equations is to express the solution as a linear combination of so-called basis functions. These basis functions can for instance be plane waves, Gaussians or finite elements. Having discretized the differential equation in this way makes it amenable to a numerical solution. In the case of Poisson's equation one obtains for instance a linear system of equation, in the case of Schrödinger's equation one obtains an eigenvalue problem. This procedure is usually more stable than other methods which do not involve basis functions, such as finite difference methods. Wavelets are just another basis set which however offers considerable advantages over alternative basis sets and allows us to attack problems not accessible with conventional numerical methods. Its main advantages are:

- The basis set can be improved in an systematic way:
  If one wants the solution of the differential equation with higher accuracy one can just add more wavelets in the expansion of the solution. This will not lead to any numerical instabilities as one encounters for instance with Gaussians. The accuracy of the solution is determined by one single parameter similar to the minimal wavelength determining the accuracy of a plane wave expansion. In the case of Gaussians there are many parameters which determine the accuracy and it is frequently not obvious which one has the largest leverage to improve upon the accuracy.

- Different resolutions can be used in different regions of space:
  If the solution of the differential equation is varying particularly rapidly in a particular region of space one can increase the resolution in this region by adding more high resolution wavelets centered around this region. This varying resolution is for instance not possible with plane waves, which give the same resolution in the whole computational volume.

- The coupling between different resolution levels is easy:
  Finite elements can also be used with varying resolution levels. The resulting highly structured grids lead however to very complicated matrix structures, requiring indirect indexing of most arrays. In the case of wavelets, in contrast, the coupling between different resolution levels is rather easy.

- There are few topological constraints for increased resolution regions:
  The regions of increased resolution can be chosen in arbitrarily, the only requirement being that a region of higher resolution be contained in a

region of the next lower resolution. If one uses for instance generalized plane waves in connection with curvilinear coordinates [1] to obtain varying resolution one has the requirement that the varying resolution grid can be obtained by a mapping from a equally spaced grid.

- The Laplace operator is diagonally dominant in an appropriate wavelet basis:
  This allows for a simple but efficient preconditioning scheme for equations such as Laplace or Schrödinger's equation which contains the Laplacian as well. As a result the number of iterations needed in the iterative solution of the linear algebra equations corresponding to these differential equations is fairly small and independent of the maximal resolution. No such easy and efficient preconditioning scheme is known for other varying resolution schemes such as finite elements, Gaussians or generalized plane waves with curvilinear coordinates.

- The matrix elements of the Laplace operator are very easy to calculate:
  The requirement that the matrix elements can easily be calculated is essential for any basis set and therefore fulfilled by all standard basis sets. For the case of wavelets it is however particularly easy since they can be calculated on the fly by simple scaling arguments and therefore need not be stored in memory.

- The numerical effort scales linearly with respect to system size:
  Three-dimensional problems of realistic size require usually a very large number of basis functions. It is therefore of utmost importance, that the numerical effort scales only linearly (and not quadratically or cubically) with respect to the number of basis functions. If one uses iterative matrix techniques, this requirement is equivalent to the two requirements, namely that the matrix vector multiplications which are necessary for all iterative methods can be done with linear scaling and that the number of matrix vector multiplications is independent of the problem size. The first requirement is fulfilled if either the matrix representing the differential operator is sparse or can be transformed into sparse form by a transformation which has linear scaling, a requirement fulfilled by wavelets. The second requirement is related to the availability of a good preconditioning scheme. Since such a scheme exists, the conditioning number of the involved matrices do not vary strongly with respect to the problem size and the number of iterations (i.e. matrix vector multiplications) is independent of the problem size.

In summary, we see that wavelets combine many advantages of standard basis sets. It is therefore to be expected that wavelet based techniques will

replace current basis sets in many applications. There will however certainly be special cases where conventional techniques are more powerful. It is for instance at present not quite clear how wavelet based method can be adapted to the highly irregular boundary geometries that are found in many engineering applications, that are usually relying on finite elements. Even though the theory of wavelets [2, 3] is already more than 10 years old, it has only been recently worked out [20] how to efficiently solve differential equations in a wavelet basis. In the mathematical literature, one can find now several papers, where wavelets are used for the solution of differential equations, however only for cases of equal resolution. In these cases as mentioned above there are however other efficient methods available and wavelets can therefore probably not offer substantial advantages over the standard methods. The full power of wavelets manifests itself however in those cases which require highly nonuniform resolution. Problems of this type exhibiting several length scales are abundant in physics. An example is for instance computational quantum chemistry where one has to deal with both valence and core electrons. The first ones are characterized by a length scale of a few atomic units, whereas the core electrons are localized within 2/100 of an atomic unit in the case of uranium. Problems of this type will therefore be the focus of this treatise and it will be shown how wavelets can be used to solve differential equations whose solution exhibits several length scales and which are practically unsolvable with other methods.

2 A first tour of some wavelet families

Many families of wavelets have been proposed in the mathematical literature. If one wants to use wavelets for the solution of differential equations, one therefore has to choose one specific family which is most advantageous for the intended application. Within one family there are also members of different degree. Without going into any detail at this point we will in the following just show some plots of some common wavelet families. Only families with compact support (i.e. they are nonzero only in a finite interval) will be presented. All these wavelet families can be classified as either being an orthogonal or biorthogonal family. The meaning of orthogonality will be explained later. Each orthogonal wavelet family is characterized by two functions, the mother scaling function ϕ and the mother wavelet ψ. In the case of biorthogonal families one has a dual scaling function $\tilde{\phi}$ and a dual wavelet $\tilde{\psi}$ in addition to the non-dual quantities.

Figure 1 shows the orthogonal Haar wavelet family, which is conceptually the simplest wavelet family. It is too crude to be useful for any numerical work, but its simplicity will help us to illustrate some basic wavelet concepts.

The Haar wavelet is identical to the zero-th degree Daubechies [3] wavelet.

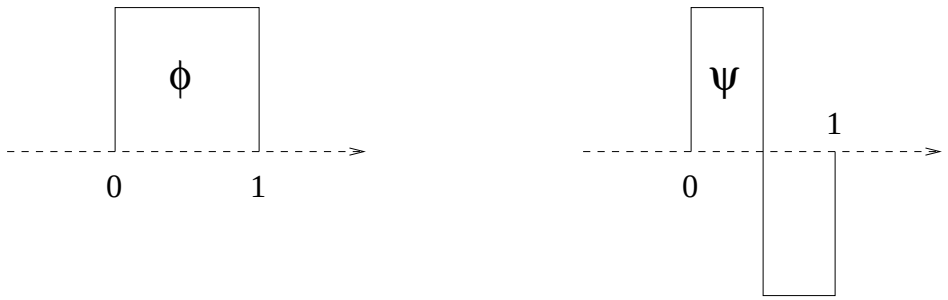


Figure 1: *The Haar scaling function ϕ and wavelet ψ.*

Figure 2 shows the 4 and 8 order Daubechies wavelets. Note that both the regularity and the support length increase with increasing order of the wavelets. The Daubechies family is an orthogonal wavelet family.

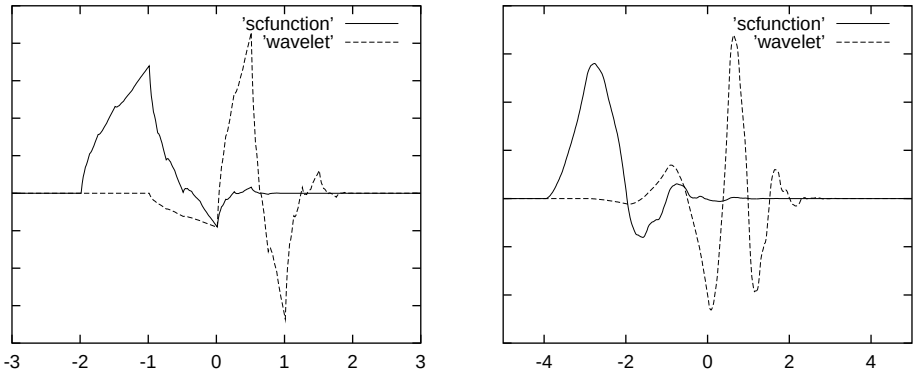


Figure 2: *The orthogonal Daubechies scaling function and wavelet of degree 4 (left panel) and 8 (right panel).*

Figure 3 shows a biorthogonal interpolating wavelet family of degree 4. It is smoother than other families of the same degree. Note that the scaling function vanishes at all integer points except at the origin.

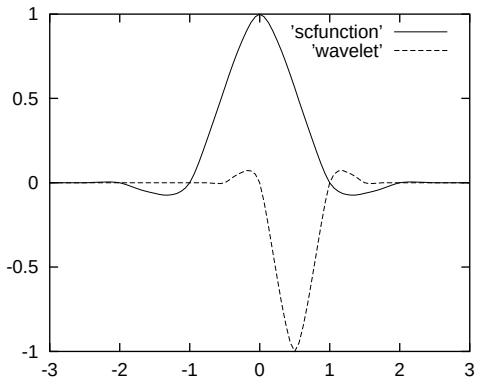


Figure 3: *The interpolating scaling function and wavelet of degree 4.*

3 Forming a basis set

To obtain a basis set at a certain resolution level k one can use all the integer translations of the mother scaling function of some wavelet family,

$$\phi_i^k(x) \propto \phi(2^k x - i) \, . \tag{1}$$

Note that with this convention higher resolution corresponds to larger values of k. Since high resolution scaling functions are skinnier, more translation indices i are allowed for a interval of fixed length. Some examples for an unspecified wavelet family are shown in Figure 4.

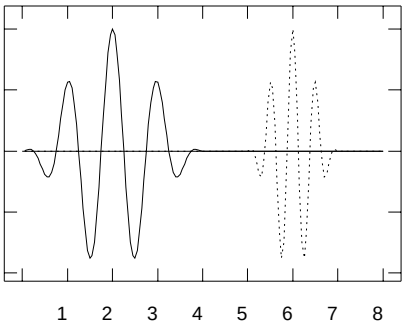


Figure 4: *Two basis functions $\phi_2^0(x)$ (solid line) and $\phi_{12}^1(x)$ (dotted line) for an arbitrary wavelet family.*

Exactly the same scaling and shifting operations can of course also be applied to the wavelets,

$$\psi_i^k(x) \propto \psi(2^k x - i) \, . \tag{2}$$

This set of wavelet basis functions can be added as a basis to the scaling functions as will be explained in the following.

4 The Haar wavelet

In the case of the Haar family, any function which can exactly be represented at any level of resolution is necessarily piecewise constant. One such function is shown in Figure 5.

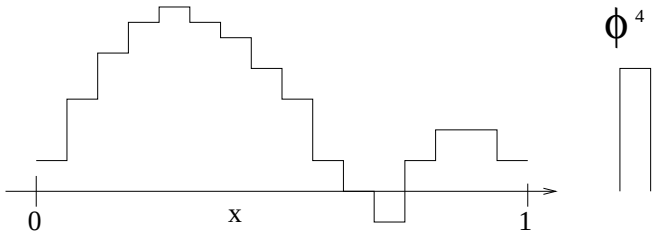


Figure 5: *A function at resolution level 4 together with the scaling function at the same resolution level.*

Evidently this function can be written as a linear combination of the scaling functions $\phi_i^4(x)$

$$f = \sum_{i=0}^{15} s_i^4 \, \phi_i^4(x) \, , \tag{3}$$

where $s_i^4 = f(i/16)$.

Another, more interesting, possibility consists of expanding a function with respect to both scaling functions and wavelets of different resolution. Even though such a expansion contains both scaling functions and wavelets, we will refer to it as a wavelet representation to distinguish it form the our scaling function representation of Equation (3). A wavelet representation is possible because a scaling function at resolution level k is always a linear combination of a scaling function and a wavelet at the next coarser level $k-1$ as shown in Figure 6.

Using this relation depicted in Figure 6, we can write any linear combination of the two scaling functions $\phi_{2i}^k(x)$ and $\phi_{2i+1}^k(x)$ as a linear combination of $\phi_i^{k-1}(x)$ and $\psi_i^{k-1}(x)$. Hence we can write f as

$$f = \sum_{i=0}^{7} s_i^3 \, \phi_i^3(x) + \sum_{i=0}^{7} d_i^3 \, \psi_i^3(x) \, . \tag{4}$$

It is easy to verify that the transformation rule for the coefficients is given

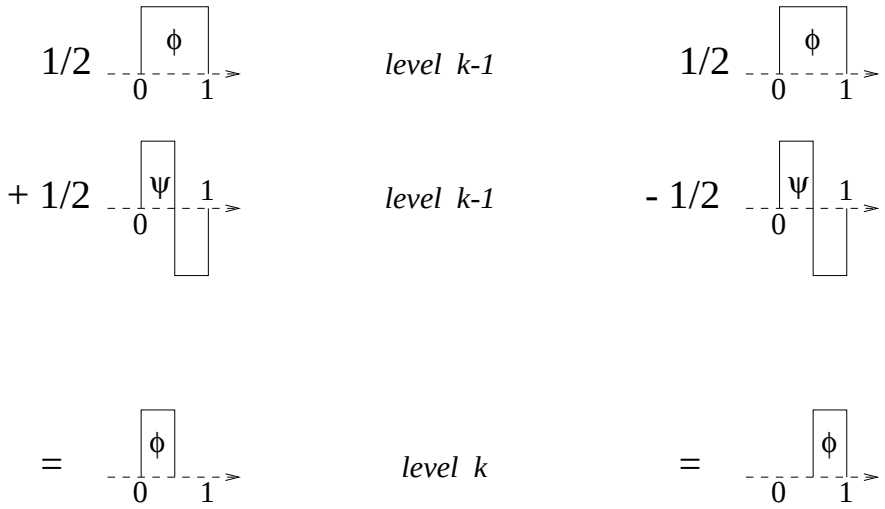


Figure 6: *A skinny (level k) scaling function is a linear combination of a fat (level $k-1$) scaling function and a fat wavelet.*

by

$$s_i^{k-1} = \frac{1}{2}s_{2i}^k + \frac{1}{2}s_{2i+1}^k \qquad ; \qquad d_i^{k-1} = \frac{1}{2}s_{2i}^k - \frac{1}{2}s_{2i+1}^k \,. \tag{5}$$

So to calculate the expansion coefficients with respect to the scaling functions at the next coarser level, we have to take an average over expansion coefficients at the finer resolution level. Because we have to take some weighted sum these coefficients are denoted by s. To get the expansion coefficients with respect to the wavelet, we have to take some weighted difference and the coefficients are accordingly denoted by d. The wavelet part contains mainly high frequency components and by doing this transformation we therefore peel off the highly oscillatory parts of the function. The remaining part represented by the coefficients s_i^{k-1} is therefore smoother. It is admittedly difficult to talk about smoothness for this kind of piecewise constant functions. This effect will be more visible for better wavelet families discussed later. For the case of our example in Figure 5 this remaining part after one transformation step is shown in Figure 7.

For any data set whose size is a power of 2, we can now apply this transformation repeatedly. In each step the number of s coefficients will be cut into half. So we have to stop the procedure as soon as there is only one s coefficient left. Such a series of transformation steps is called a forward Haar wavelet transform. The resulting wavelet representation of the function in

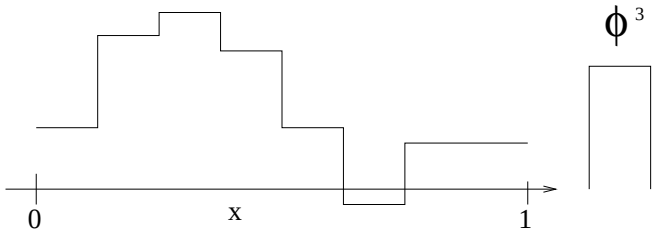


Figure 7: *The function from Figure 5 at resolution level 3.*

Equation (3) is then

$$f = s_0^0\phi_0^0(x) + d_0^0\psi_0^0(x) + \sum_{i=0}^{1} d_i^1\psi_i^1(x) + \sum_{i=0}^{3} d_i^2\psi_i^2(x) + \sum_{i=0}^{7} d_i^3\psi_i^3(x)\,. \qquad (6)$$

Note that in both cases we need exactly 16 coefficients to represent the function. In the coming sections such wavelet representations will be the focus of our interest.

By doing a backward wavelet transform, we can go back to the original scaling function representation of Equation (3). Starting at the coarsest resolution level, we have to express each scaling function and wavelet on the coarse level in terms of scaling functions at the finer level. This can be done exactly because wavelet families satisfy the so-called refinement relations depicted in Figure 8 for the Haar family.

It then follows that we have to back-transform the coefficients in the following way

$$s_{2i}^{k+1} = s_i^k + d_i^k \qquad ; \qquad s_{2i+1}^{k+1} = s_i^k - d_i^k\,. \qquad (7)$$

5 The concept of Multi-Resolution Analysis

In the previous sections a very intuitive introduction to wavelet theory was given. The formal theory behind wavelets is called Multi-Resolution Analysis [3] (MRA). Even though the formal definitions of MRA are usually not required for practical work, we will for completeness briefly present them. The equations which are useful for numerical work will be listed afterwards.

5.1 Formal definition of Multi-Resolution Analysis

- A Multi-Resolution Analysis consists of a sequence of successive approximation spaces V_k and associated dual spaces $\tilde{V}_k$, (which turn out to be the scaling function spaces and their dual counterpart) satisfying

$$V_k \subset V_{k+1} \qquad ; \qquad \tilde{V}_k \subset \tilde{V}_{k+1}\,.$$

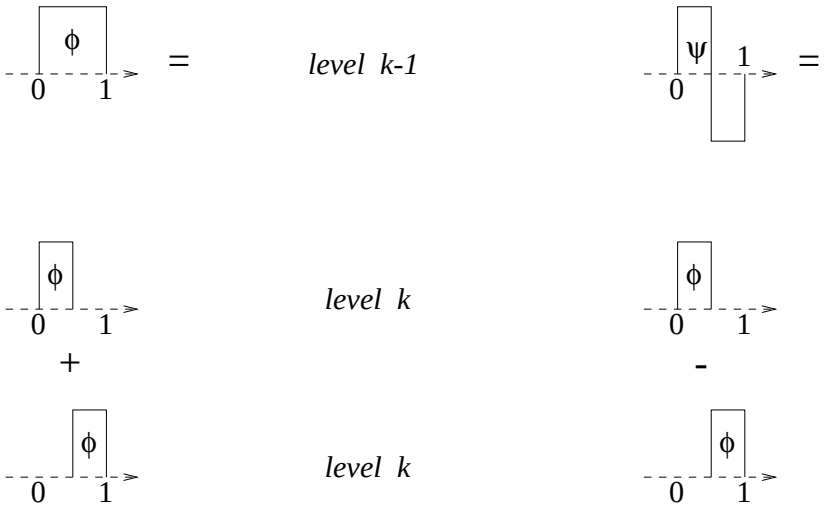


Figure 8: *Fat (level $k-1$) scaling functions and fat wavelets are linear combinations of skinny (level k) scaling functions.*

- If a function $f(x)$ is contained in the space V_k, the compressed function $f(2x)$ has to be contained in the higher resolution space V_{k+1},

$$f(x) \in V_k \Leftrightarrow f(2x) \in V_{k+1} \qquad ; \qquad f(x) \in \tilde{V}_k \Leftrightarrow f(2x) \in \tilde{V}_{k+1} \, .$$

- If a function $f(x)$ is contained in the space V_k, its integer translate has to be contained in the same space,

$$f(x) \in V_0 \Leftrightarrow f(x+1) \in V_0 \qquad ; \qquad f(x) \in \tilde{V}_0 \Leftrightarrow f(x+1) \in \tilde{V}_0 \, .$$

- The union of all these spaces is the $L^2(\Re)$ space,

$$\overline{\bigcup_k V_k} = L^2(\Re) \, .$$

- There exists a biorthogonal pair of functions spanning V_k,

$$\int \tilde{\phi}_i^k(x) \, \phi_j^k(x) \, dx = \delta_{i,j} \, .$$

The wavelet spaces W_k, $\tilde{W}_k$ are then defined as the complement (orthogonal complement in the case of orthogonal families) of V_k in V_{k+1},

$$V_{k+1} = V_k \oplus W_k \qquad ; \qquad \tilde{V}_{k+1} = \tilde{V}_k \oplus W_k \, .$$

5.2 Basic formulas for biorthogonal wavelet families

The formal MRA requirements listed above lead to the following useful basic facts of wavelet analysis. The interested reader can find the nontrivial proofs of these formulas in the book by Daubechies [3]. The list is highly redundant and in the exercises some of the dependencies will be explored.

- A biorthogonal wavelet family of degree m is characterized by 4 finite filters denoted by h_j, $\tilde{h}_j$, g_j, $\tilde{g}_j$. Since we will mainly deal with symmetric wavelet families, whose filters have a natural symmetry center, we will adopt a convention where the nonzero filter elements are in the interval $j = -m, ..., m$, and where m is even. In case the number of nonzero filter elements does not fit into this convention, it is always possible to pad the filters on both sides with zeroes, and to increase m artificially until it is compatible with this convention.

  The filter coefficients satisfy the orthogonality relations

$$\sum_l h_{l-2i}\, \tilde{h}_{l-2j} = \delta_{i,j}\,, \tag{8}$$

$$\sum_l g_{l-2i}\, \tilde{g}_{l-2j} = \delta_{i,j}\,, \tag{9}$$

$$\sum_l h_{l-2i}\, \tilde{g}_{l-2j} = 0\,, \tag{10}$$

$$\sum_l \tilde{h}_{l-2i}\, g_{l-2j} = 0 \tag{11}$$

  and the symmetry relations

$$g_{i+1} = (-1)^{i+1}\tilde{h}_{-i}\,, \tag{12}$$

$$\tilde{g}_{i+1} = (-1)^{i+1}h_{-i}\,. \tag{13}$$

- Scaling functions and wavelets at a coarse level can be written as the following linear combinations of scaling functions at a higher resolution level. These equations are called refinement relations,

$$\phi(x) = \sum_{j=-m}^{m} h_j\, \phi(2x-j)\,, \tag{14}$$

$$\psi(x) = \sum_{j=-m}^{m} g_j\, \phi(2x-j)\,, \tag{15}$$

$$\tilde{\phi}(x) = 2\sum_{j=-m}^{m} \tilde{h}_j\, \tilde{\phi}(2x-j)\,, \tag{16}$$

$$\tilde{\psi}(x) = 2\sum_{j=-m}^{m} \tilde{g}_j\, \tilde{\phi}(2x-j)\,. \tag{17}$$

In terms of the the two index multi level basis functions defined by,

$$\begin{aligned}
\phi_i^k(x) &= \phi(2^k x - i)\,, && (18)\\
\psi_i^k(x) &= \psi(2^k x - i)\,, && (19)\\
\tilde{\phi}_i^k(x) &= 2^k\tilde{\phi}(2^k x - i)\,, && (20)\\
\tilde{\psi}_i^k(x) &= 2^k\tilde{\psi}(2^k x - i)\,, && (21)
\end{aligned}$$

the refinement relations are,

$$\begin{aligned}
\phi_i^k(x) &= \sum_{j=-m}^{m} h_j\, \phi_{2i+j}^{k+1}(x)\,, && (22)\\
\psi_i^k(x) &= \sum_{j=-m}^{m} g_j\, \phi_{2i+j}^{k+1}(x)\,, && (23)\\
\tilde{\phi}_i^k(x) &= \sum_{j=-m}^{m} \tilde{h}_j\, \tilde{\phi}_{2i+j}^{k+1}(x)\,, && (24)\\
\tilde{\psi}_i^k(x) &= \sum_{j=-m}^{m} \tilde{g}_j\, \tilde{\phi}_{2i+j}^{k+1}(x)\,. && (25)
\end{aligned}$$

- A wavelet analysis (forward) transform is given by

$$\begin{aligned}
s_i^{k-1} &= \sum_{j=-m}^{m} \tilde{h}_j s_{j+2i}^k\,, && (26)\\
d_i^{k-1} &= \sum_{j=-m}^{m} \tilde{g}_j s_{j+2i}^k\,.
\end{aligned}$$

A wavelet synthesis (backward) transform is given by

$$\begin{aligned}
s_{2i}^{k+1} &= \sum_{j=-m/2}^{m/2} h_{2j}\, s_{i-j}^k + g_{2j}\, d_{i-j}^k && (27)\\
s_{2i+1}^{k+1} &= \sum_{j=-m/2}^{m/2} h_{2j+1}\, s_{i-j}^k + g_{2j+1}\, d_{i-j}^k\,.
\end{aligned}$$

These two equations are generalizations of equations (5), (7) that we derived in an intuitive way.

The wavelet transform is in principle for periodic data sets. Therefore the subscripts of the s and d coefficients have to be wrapped around once they are out of bounds.

- The fundamental functions satisfy the following orthogonality relations,

$$\int \tilde{\phi}_i^k(x)\phi_j^k(x)dx = \delta_{i,j}\,, \tag{28}$$

$$\int \tilde{\psi}_i^k(x)\phi_j^q(x)dx = 0\,,\ k \geq q\,, \tag{29}$$

$$\int \psi_i^k(x)\tilde{\phi}_j^q(x)dx = 0\,,\ k \geq q\,, \tag{30}$$

$$\int \psi_i^k(x)\tilde{\psi}_j^q(x)dx = \delta_{k,q}\delta_{i,j}\,. \tag{31}$$

Exercises

- 1) Show that the formulas for the backward transform (27) follow from the refinement relations for the scaling function and wavelet. (Equations (22) to (25))
- 2) Show that a forward transform (26) followed by a backward transform (27) gives the identity.
- 3) Prove the orthogonality relations (Equations (28) to (31)) for the fundamental functions using the properties of the associated filters.
- 4) Show that the normalization condition $\int \phi(x)dx = 1$ leads to the condition $\sum_j h_j = 2$.

5.3 Basic formulas for orthogonal wavelet families

- An orthogonal wavelet family of degree m is characterized by 2 finite filters denoted by h_j, g_j, satisfying the orthogonality relations

$$\sum_l h_{l-2i}\, h_{l-2j} = \delta_{i,j}\,, \tag{32}$$

$$\sum_l g_{l-2i}\, g_{l-2j} = \delta_{i,j}\,, \tag{33}$$

$$\sum_l h_{l-2i}\, g_{l-2j} = 0 \tag{34}$$

  and the symmetry relation

$$g_{i+1} = (-1)^{i+1} h_{-i}\,. \tag{35}$$

- The refinement relations are

$$\phi(x) = \sqrt{2}\sum_{j=-m}^{m} h_j\, \phi(2x-j)\,, \tag{36}$$

$$\psi(x) = \sqrt{2}\sum_{j=-m}^{m} g_j\, \phi(2x-j)\,. \tag{37}$$

In terms of the the two index multi level basis functions defined by

$$\phi_i^k(x) = \sqrt{2}^k \phi(2^k x - i) \,, \tag{38}$$

$$\psi_i^k(x) = \sqrt{2}^k \psi(2^k x - i) \,, \tag{39}$$

the refinement relations are

$$\phi_i^k(x) = \sum_{j=-m}^{m} h_j \, \phi_{2i+j}^{k+1}(x) \,, \tag{40}$$

$$\psi_i^k(x) = \sum_{j=-m}^{m} g_j \, \phi_{2i+j}^{k+1}(x) \,. \tag{41}$$

- The formulas for the forward and backward wavelet transforms are identical to the biorthogonal case (Equation (26) and (27)), with the exception that the filters $\tilde{h}$ and $\tilde{g}$ have to be replaced by the filters h and g in the forward transform.

- The fundamental functions satisfy the orthogonality relations,

$$\int \phi_i^k(x)\phi_j^k(x)dx = \delta_{i,j} \,, \tag{42}$$

$$\int \psi_i^k(x)\phi_j^q(x)dx = 0 \,,\ k \geq q \,, \tag{43}$$

$$\int \psi_i^k(x)\psi_j^q(x)dx = \delta_{k,q}\delta_{i,j} \,. \tag{44}$$

6 The fast wavelet transform

Let us first look at the forward transform given by Equation (26). The peeling off of the high frequency components in the forward transform can be illustrated in the following way:

$$\begin{array}{ccccccccc} S^4 & \rightarrow & S^3 & \rightarrow & S^2 & \rightarrow & S^1 & \rightarrow & S^0 \\ & \searrow & & \searrow & & \searrow & & \searrow & \\ & & D^3 & & D^2 & & D^1 & & D^0 \end{array}$$

We note that just two arrays of length n (where n is a power of 2) are necessary to do the transform as shown below:

original data
$s_0^4\ s_1^4\ s_2^4\ s_3^4\ s_4^4\ s_5^4\ s_6^4\ s_7^4\ s_8^4\ s_9^4\ s_{10}^4\ s_{11}^4\ s_{12}^4\ s_{13}^4\ s_{14}^4\ s_{15}^4\ =\ S^4$

after first sweep
$s_0^3\ s_1^3\ s_2^3\ s_3^3\ s_4^3\ s_5^3\ s_6^3\ s_7^3\ d_0^3\ d_1^3\ d_2^3\ d_3^3\ d_4^3\ d_5^3\ d_6^3\ d_7^3\ =\ S^3, D^3$

after second sweep
$s_0^2\ s_1^2\ s_2^2\ s_3^2\ d_0^2\ d_1^2\ d_2^2\ d_3^2\ d_0^3\ d_1^3\ d_2^3\ d_3^3\ d_4^3\ d_5^3\ d_6^3\ d_7^3\ =\ S^2, D^2, D^3$

after third sweep
$s_0^1\ s_1^1\ d_0^1\ d_1^1\ d_0^2\ d_1^2\ d_2^2\ d_3^2\ d_0^3\ d_1^3\ d_2^3\ d_3^3\ d_4^3\ d_5^3\ d_6^3\ d_7^3\ =\ S^1, D^1, D^2, D^3$

final data
$s_0^0\ d_0^0\ d_0^1\ d_1^1\ d_0^2\ d_1^2\ d_2^2\ d_3^2\ d_0^3\ d_1^3\ d_2^3\ d_3^3\ d_4^3\ d_5^3\ d_6^3\ d_7^3\ =\ S^0, D^0, D^1, D^2, D^3$

Note that this transformation from the original data to the final data corresponds exactly to the transformation done in an intuitive way to get from Equation (3) to Equation (6). Just as in the case of a Fast Fourier transform we have $\log_2(n)$ sweeps to do a full transform. However in the case of the wavelet transform the active data set (the s coefficients) is cut into half in each sweep. If our filters h and g have length $2m$ the operation count is then given by $2m(n+n/2+n/4+...)$. Replacing the finite geometric series by its infinite value, the total operation count is thus given by $4mn$

The backward transform (Equation (27)) can pictorially be represented by the following diagram:

$$\begin{array}{ccccccccc} S^4 & \rightarrow & S^3 & \rightarrow & S^2 & \rightarrow & S^1 & \rightarrow & S^0 \\ & \nwarrow & & \nwarrow & & \nwarrow & & \nwarrow & \\ & & D^3 & & D^2 & & D^1 & & D^0 \end{array}$$

As can easily been seen the operation count is again $4mn$ and again it can be done with 2 arrays of length n.

Since each sweep in a wavelet transform is a linear operation it can be represented by a matrix. Denoting the matrix for one sweep in a forward transform by $\tilde{\mathbf{F}}$ and in a backward transform by $\mathbf{B}$ they have the structures shown in Figure 9.

In the case of biorthogonal wavelets we have

$$\mathbf{F}^T = \tilde{\mathbf{F}}^{-1} = \mathbf{B} \qquad ; \qquad \tilde{\mathbf{B}}^T = \mathbf{B}^{-1} = \tilde{\mathbf{F}}\,, \tag{45}$$

where the tilde on the matrix means that the filter coefficients necessary to fill the matrix are replaced by their dual counterparts.

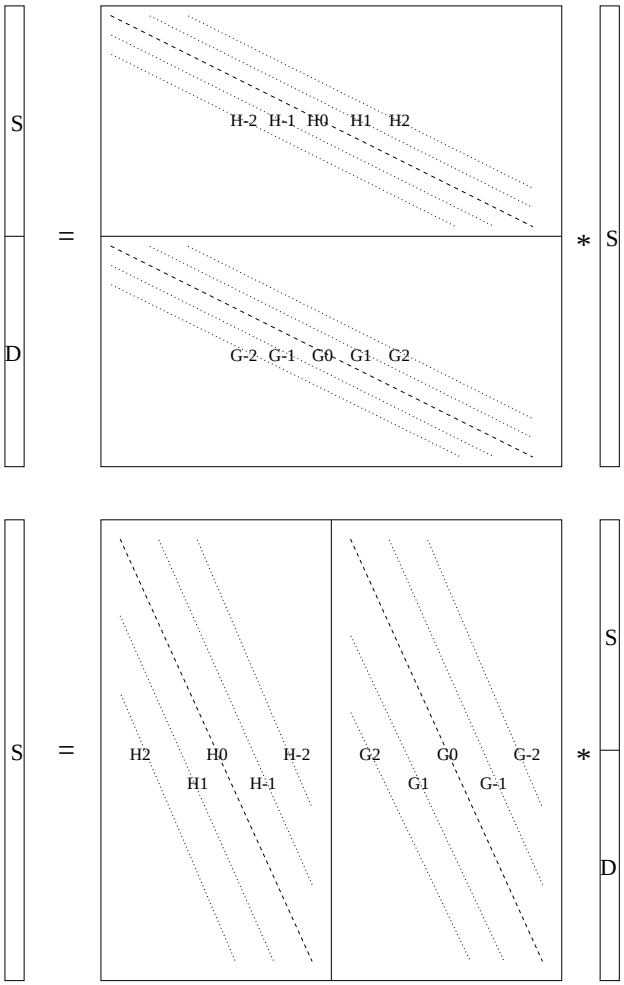


Figure 9: *Graphical representation of matrices* $\mathbf{F}$ *(top) and* $\mathbf{B}$ *(bottom).*

In the case of orthogonal wavelets they satisfy

$$\mathbf{F}^T = \mathbf{F}^{-1} = \mathbf{B} \qquad ; \qquad \mathbf{B}^T = \mathbf{B}^{-1} = \mathbf{F} \, . \tag{46}$$

Each sweep in a transform is thus a unitary transformation and the full wavelet transform is therefore a unitary transformation as well.

Exercises

- 5) Write a computer program to do a wavelet transform (This program will be needed for coming exercises).
- 6) Prove that $\tilde{\mathbf{B}}^T = \tilde{\mathbf{F}}$

7 How to plot wavelets

The easiest way to generate wavelet and scaling function plots is based on backward wavelet transforms. To generate the scaling function we start with a data set where all the wavelet d coefficients on all resolution levels are zero and where only one scaling function s coefficient is nonzero, i.e. $s^0_{Kmin} = 1$. In the case of the wavelet only one d coefficient is nonzero, i.e. $d^0_{Kmin} = 1$. By doing repeated backward transform sweeps, we express these two functions by skinnier and skinnier scaling functions. Since the wavelet and scaling functions have in general certain smoothness properties, they will be nearly constant within a very small interval and neighboring coefficients at very high resolution levels will vary very little. The average over very small intervals will thus be proportional to the s coefficients at sufficiently high resolution. The plots in Figures 1 to 5 were all generated in this way using data sets of 1024 entries. The filter coefficients necessary for these plots are listed in the Appendix.

Exercise

- 7) Plot Daubechies 4-th order scaling function and wavelet using the coefficients given in the Appendix. Zoom into a small interval and look at the smoothness of the functions at that scale. Is it possible to represent a constant as a linear superposition of these scaling functions?

8 Interpretation of a wavelet spectrum

The Fourier spectrum $F(k)$ tells you about the importance of certain frequencies k in a function. The variable k is a continuous or quasi continuous variable, and thus any frequency k can be inspected. In the case of a wavelet analysis, the variable k increases by powers of two, and it is therefore not possible to look at any frequency. In contrast to the traditional Fourier spectrum, the wavelet spectrum gives still a second information, namely at which locations in time x those frequencies can be found. The wavelet spectrum is thus a 2-dimensional entity. This information about the location in time is very important in most practical applications. For the case of differential equations this means, that we can have varying resolution in different regions of space as discussed in the first chapter. In the signal processing community many efforts have been made to use windowed Fourier techniques to get just this time location information. It also turns out that biological devices such as the human ear rather "do" a wavelet transform than a traditional Fourier transform. We perceive music as a temporal sequence of certain frequen-

cies. Such a statement would be invalid in the strict Fourier sense, where frequencies are only defined in the limit of infinite duration.

9 Interpolating wavelets

As will be discussed later, interpolating wavelets have many properties, which make them highly suitable as basis sets for partial differential equations. At the same time they are conceptionally the simplest wavelets. We will therefore describe the construction of the elementary interpolating wavelet [14, 13] in detail.

The construction of interpolating wavelets is closely connected to the question of how to construct a continuous function $f(x)$ if only its values f_i on a finite number of grid points i are known. One way to do this is by recursive interpolation. In a first step we interpolate the functional values on all the midpoints by using for instance the functional values of two grid points to the right and of two grid points to the left of the midpoint. These four functional values allow us to construct a third order polynomial and we can then evaluate it at the midpoint. In the next step, we take this new data set, which is now twice as large as the original one, as the input for a new midpoint interpolation procedure. This can be done recursively ad infinitum until we have a quasi continuous function.

Let us now show, how this interpolation prescription leads to a set of basis functions. Denoting by the Kronecker δ_{i-j} a data set whose elements are all zero with the exception of the element at position j, we can write any initial data set as a linear combination of such Kronecker data sets: $f_i = \sum_j f_j \delta_{i-j}$. Now the whole interpolation procedure is clearly linear, i.e. the sum of two interpolated values of two separate data sets is equal to the interpolated value of the sum of these two data sets. This means that we can instead also take all the Kronecker data sets as the input for separate ad-infinitum interpolation procedures, to obtain a set of functions $\phi(x - j)$. The final interpolated function is then identical to $f(x) = \sum_j f_j \phi(x - j)$. If the initial grid values f_i were the functional values of a polynomial of degree less than four, we obviously will have exactly reconstructed the original function from its values on the grid points. Since any smooth function can locally be well approximated by a polynomial, these functions $\phi(x)$ are good basis functions and we will use them as scaling functions to construct a wavelet family.

The first construction steps of an interpolating scaling function are shown in Figure 10 for the case of linear interpolation. The initial Kronecker data set is denoted by the big dots. The additional data points obtained after the first interpolation step are denoted by medium size dots and the additional data points obtained after the second step by small dots.

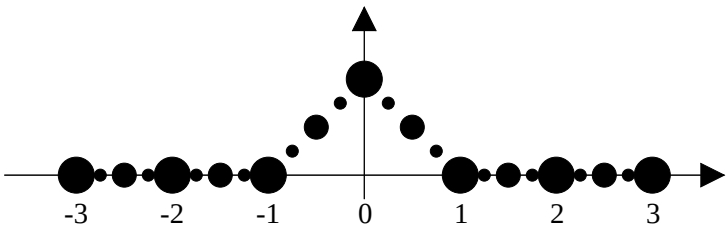


Figure 10: *The first two steps of a recursive interpolation procedure in the case of simple linear interpolation. The original data point are represented by the big dots, data points filled in by the following two interpolation steps by medium and small dots.*

Continuing this process ad infinitum will then result in the function shown in the left panel of Figure 11. If an higher order interpolation scheme is used the function shown in the right panel of Figure 11 is obtained.

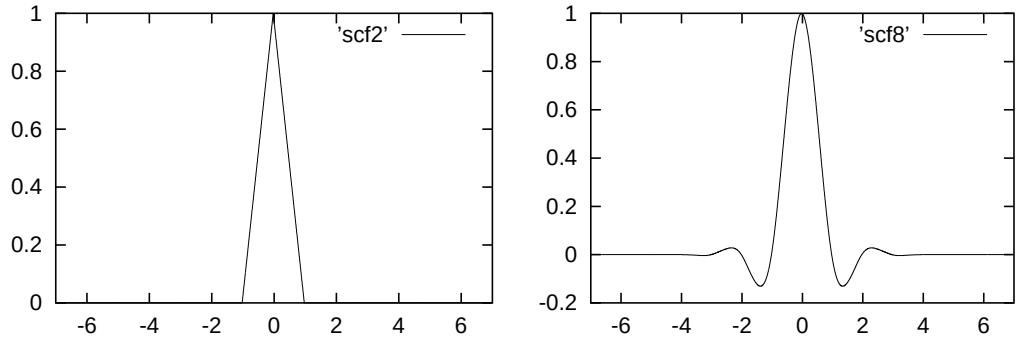


Figure 11: *A Kronecker delta interpolated ad infinitum with linear interpolation (left panel) and 7-th order interpolation (right panel) .*

By construction it is clear, that $\phi(x)$ has compact support. If an $(m-1)$-th order interpolation scheme is used, the filter length is $(m-1)$ and the support interval of the scaling function is $[-(m-1);(m-1)]$.

It is also not difficult to see that the function $\phi(x)$ satisfies the refinement relation. Let us again consider the interpolation ad infinitum of a Kronecker data set which has everywhere zero entries except at the origin. We can now split up this process into the first step where we calculate the half-integer grid point values and a remaining series of separate ad infinitum interpolations for all half-integer Kronecker data sets, which are necessary to represent the data set obtained by the first step. Doing the ad-infinitum interpolation for a half integer data set with a unit entry at (half integer) position j, we obviously obtain the same scaling function, just compressed by a factor of 2,

$\phi(2x-j)$. If we are using a $(m-1)$-th order interpolation scheme (i.e. m input data for the interpolation process) we thus get the relation

$$\phi(x) = \sum_{j=-m+1}^{m-1} \phi(j/2)\, \phi(2x-j)\,. \tag{47}$$

Comparing this equation with the refinement relation Equation (14) we can identify the first filter h as

$$h_j = \phi(j/2)\ ,\ \ j = -m+1, m-1\,.$$

For the case of third order interpolation the numerical values of h follow from the standard interpolation formula and are given by:

```
h =   {-1/16 , 0 , 9/16 , 1 , 9/16 , 0 , -1/16}
j =     -3    -2    -1    0     1    2     3
```

Let us next determine the filter $\tilde{h}$. Let us consider a function $f(x)$ which is band-limited in the wavelet sense, i.e which can exactly be represented by a superposition of scaling functions at a certain resolution level K

$$f(x) = \sum_j c_j \phi_j^K(x)\,.$$

It then follows from the orthogonality relation Equation (28) that

$$c_j = \int \tilde{\phi}_j^K(x) f(x) dx\,.$$

Now we have seen above that with respect to interpolating scaling functions, a band-limited function is just any polynomial of degree less than or equal to $m-1$, and that in this case the expansion coefficients c_j are just the functional values at the grid points (Equation (47)). We therefore have

$$\int \tilde{\phi}_j^K(x) f(x) dx = f_j\,,$$

which shows that the dual scaling function $\tilde{\phi}$ is the delta function. Obviously the delta function satisfies a trivial refinement relation $\delta(x) = 2\delta(2x)$ and from Equation (16) we conclude that $\tilde{h}_j = \delta_j$.

```
ht =   {  0 ,  0  , 1 ,  0  , 0  }
j  =     -2   -1    0    1    2
```

From the symmetry Equations (12), (13) for the filters we can now determine the two remaining filters and we have thus completely specified our wavelet family. For $\tilde{g}_j$ we obtain

```
gt =   {  0 , 0 , -1/16 , 0 , 9/16 , -1 , 9/16 , 0 , -1/16 }
j  =      -4  -3    -2    -1    0      1    2     3     4
```

For g_j we obtain

```
g =   {  0 ,  0 ,  0 ,  1 ,  0 }
j =      -2   -1    0    1    2
```

As expected, these 4 filters satisfy the orthogonality conditions (8) to (11).

Due to the easy structure of the filters in this case, the backward transform can be done by inspection to obtain the form of the 4 fundamental functions ϕ, ψ, $\tilde{\phi}$ and $\tilde{\psi}$. In the case of the scaling function ϕ, the d coefficients at all resolution levels vanish. For the even elements Equation (27) becomes

$$s_{2i}^{k+1} = s_i^k \, .$$

So the even grid points on the finer grid take on the values of the coarser grid. The odd filter elements are just the interpolating coefficients giving:

$$s_{2i+1}^{k+1} = h_3 \, s_{i-1}^k + h_1 \, s_{i+0}^k + h_{-1} \, s_{i+1}^k + h_{-3} \, s_{i+2}^k \, .$$

So the values at the odd fine grid points are just interpolated from the coarse grid points. In summary we thus see that an infinite series of backward transforms just describes the ad-infinitum interpolation process depicted in Figure 10.

In the case of the wavelet ψ the only nonzero d coefficient in the input data will generate in the first step a s data set where again only one coefficient is nonzero, since the g filter has only one nonzero entry. Continuing the procedure one will thus obtain for the wavelet a (negative) scaling function compressed by a factor of 2, $\psi(x) = -\phi(2x-1)$.

To generate the dual functions $\tilde{\phi}$ and $\tilde{\psi}$, one has to replace the filters h and g in the backward transform by the dual counterparts $\tilde{h}$ and $\tilde{g}$. For the case of the dual scaling function $\tilde{\phi}$, one sees by inspection that the backward transform equations Equation (27) become:

$$s_{2i+1}^{k-1} = 0 \qquad ; \qquad s_{2i}^{k-1} = \begin{cases} 1 & \text{if } i = 0 \\ 0 & \text{otherwise} \end{cases}$$

As one should, one thus obtains a delta function

$$\tilde{\phi}(x) = \delta(x) \, . \tag{48}$$

For the case of a dual wavelet $\tilde{\psi}$ the argument is analogous to the non-dual case. In the first step of the backward transform the filter $\tilde{g}$ generates 5

nonzero s coefficients, which will become 5 delta functions through the action of the filter $\tilde{h}$. We get

$$\begin{aligned}\tilde{\psi}(x) = & - \tfrac{1}{16}\delta((x-\tfrac{1}{2})+3/2) + \tfrac{9}{16}\delta((x-\tfrac{1}{2})+1/2) - \delta((x-\tfrac{1}{2})) \\ & + \tfrac{9}{16}\delta((x-\tfrac{1}{2})+1/2) - \tfrac{1}{16}\delta((x-\tfrac{1}{2})+3/2)\,. \end{aligned} \quad (49)$$

We thus see that the interpolating wavelet is a very special case in that its scaling function and wavelet have the same functional form and that the dual functions are related to the delta function. The non-dual functions are shown in Figure 3 . Filters for interpolating wavelets of other degrees are given in the Appendix.

Exercises

- 8) Prove that the support interval of the interpolating scaling function obtained by $(m-1)$th order interpolation is $[-(m-1);(m-1)]$.
- 9) Construct and plot the 6-th order interpolating wavelet family.

10 Expanding polynomials in a wavelet basis

Functions of practical interest are of course not simple polynomials, and it will be discussed later how to expand arbitrary functions in a wavelet basis. For several proofs the expansion of polynomials in a wavelet basis is however important and we will therefore derive the following theorem: The scaling function expansion coefficients $s_i(l)$ of a polynomial of degree l are themselves a polynomial of the same degree l.

Let us first demonstrate the theorem for the trivial case of a constant, i.e. $l=0$. The expansion coefficients $s_i(0)$ are given by $\int \tilde{\phi}(x-i)dx$. Assuming $\int \tilde{\phi}(x)dx$ is normalized to 1 we thus obtain $s_i(0)=1$.

In the linear case (i.e. $l=1$) we have $s_i(1) = \int \tilde{\phi}(x-i)xdx$. For the shifted coefficient we get

$$\begin{aligned} s_{i+1}(1) &= \int \tilde{\phi}(x-i-1)xdx = \int \tilde{\phi}(x-i)(x+1)dx \\ &= s_i(1)+1\,. \end{aligned} \quad (50)$$

So we see that $s_i(1)$ satisfies the difference equation for a linear polynomial and that it is therefore a linear polynomial.

For arbitrary degree l we get

$$s_{i+1}(l) = \int \tilde{\phi}(x-i-1)x^l dx = \int \tilde{\phi}(x-i)(x+1)^l dx \quad (51)$$

$$
\begin{aligned}
&= \sum_{\tau} \int \tilde{\phi}(x-i) \frac{l!}{\tau!(l-\tau)!} x^{\tau} dx \\
&= \sum_{\tau} \frac{l!}{\tau!(l-\tau)!} s_i(\tau) \; .
\end{aligned}
$$

So wee see indeed that $s_i(l)$ is a polynomial of lth degree since it satisfies the corresponding difference equation, which proves the theorem.

11 Orthogonal versus biorthogonal wavelets

The interpolating wavelets constructed above are a special case of so-called biorthogonal wavelet families. The interpolating wavelets have the property of being the smoothest ones for a fixed filter length. On the other hand the dual functions of the interpolating wavelet family are the least smooth ones. Loosely speaking the sum of the smoothness of the dual and non-dual space are a constant for a given filter length. For a given filter length one can therefore either go for maximum smoothness in the dual or non-dual space. The interpolating wavelets are your favorite choice if you want maximum smoothness in the non-dual space.

Wavelets are called orthogonal if the dual quantities are equal to the non-dual quantities. In the case of orthogonal wavelets the smoothness in dual and non-dual space is thus obviously the same. They are therefore not as smooth as the interpolating wavelets. The smoothness properties of the Daubechies family are actually not as bad as on might expect from looking at the "ugly" plots. With the 4-th order family one can exactly represent linear function, with the 6-th order family quadratic and with the 8-th order family cubic polynomials.

Exercise

- Perform a numerical experiment to verify the statements about the smoothness properties of the Daubechies family. In order to find the expansion coefficients of the scaling functions use the theorem of the preceeding section.

12 Expanding functions in a wavelet basis

As we have seen, there are two possible representations of a function within the framework of wavelet theory. The first one is called scaling function representation and involves only scaling functions. The second is called wavelet

representation and involves wavelets as well as scaling functions. Both representations are completely equivalent and exactly the same number of coefficients are needed. The scaling function representation is given by

$$f(x) = \sum_j s_j^{Kmax} \phi_j^{Kmax}(x) \; . \tag{52}$$

Evidently this approximation is more accurate if we use skinnier scaling functions from a higher resolution level $Kmax$. From the orthogonality relations (28) it follows, that the coefficients are given by

$$s_j^{Kmax} = \int \tilde{\phi}_j^{Kmax}(x) \; f(x) dx \; . \tag{53}$$

Once we have a set of coefficients s_j^{Kmax} we can use a full forward wavelet transform to obtain the wavelet representation

$$f(x) = \sum_j s_j^{Kmin} \; \phi_j^{Kmin}(x) + \sum_{K=Kmin}^{Kmax} \sum_j d_j^K \; \psi_j^K(x) \; . \tag{54}$$

Alternatively, one could also directly calculate the d coefficients by integration

$$d_j^K = \int \tilde{\psi}_j^K(x) \; f(x) dx \; . \tag{55}$$

The above Equation (55) follows from the orthogonality relations (29) to (31). So we see that if we want to expand a function either in scaling functions or wavelets, we have to perform integrations at some point to calculate the coefficients. For general wavelet families this integration is fairly cumbersome [4] and requires especially in 2 and 3 dimensions a substantial number of integration points. Furthermore it is not obvious how to do the integration if the function is only given in tabulated form. If one wants to obtain the same number of coefficients as one has functional values, one could either first interpolate the function to obtain the necessary number of integration points, which will introduce additional approximations. If one does not generate additional integration points, then the number of coefficients will necessarily be less than the number of functional values and information is thus lost. The interpolating wavelets discussed above are the glorious exception. Since the dual scaling function is a delta function and since the dual wavelet is a sum of the delta functions, one or a few data points are sufficient to do the integration exactly. In the case of periodic data sets, the filters will wrap around for data points close enough to the boundary of the periodic volume. One will therefore get exactly the same number of coefficients as one has data points and one has an invertible one-to-one mapping between the functional values on the grid and the expansion coefficients.

Non-periodic data sets can also be handled. In this case we have to put the non-periodic data set into a larger periodic data set consisting of zeroes. This composite data set will then contain the nonzero non-periodic data set in the middle surrounded by a layer of zeroes on all sides. If this layer of zeroes is broader than half the filter length $m/2$ opposite ends will not interfere during one sweep of a wavelet transform and one obtains the correct representation of the non-periodic function. Correct means in this context that the value of the nonzero coefficients would not change if we made the surrounding layer of zeroes broader.

The interpolating wavelets are also unbeatable from the point of view of accuracy. The accuracy of a scaling function expansion depends on the smoothness of the scaling function. This is easy to see for the case of interpolating wavelets. The functional value of a scaling function expansion at any midpoint is given by interpolation and thus the error is also given by the well known interpolation error formula. If h is the grid spacing and $(m-1)$-th order interpolation is used then the error is proportional to h^m. So since the interpolating wavelets are the smoothest wavelets they are also the most accurate ones. If on the other hand one is willing to accept a certain error then the interpolating wavelets will meet this error criteria with the smallest number of expansion coefficients.

This fact can also be understood from a different point of view. Let us introduce the moments $\tilde{M}_l$,

$$\tilde{M}_l = \int \tilde{\psi}(x)\, x^l dx \; .$$

Now we know, that locally any smooth function can be approximated by a polynomial. Let us for simplicity consider the coefficient d_0^K at the origin,

$$d_0^K = \sum_{\nu=0}^{\infty} \int f^\nu(0) \frac{x^\nu}{\nu!}\, \tilde{\psi}_0^K(x) dx \; .$$

If the first L moments $l = 0, ..., L-1$ vanish this becomes

$$d_0^K = \sum_{\nu=L}^{\infty} h^\nu C_\nu \; ,$$

where we have used the fact that $\tilde{\psi}$ is a sum of delta functions and where C_ν are appropriate constants. The d coefficients decay therefore as h^L and since the error is proportional to the coefficients of the wavelets which are discarded, the error is proportional to h^L as well. In the case of the 4-th order interpolating wavelet it is easy to see, that the first 4 moments vanish, $\tilde{M}_l = 0, l = 0, 1, 2, 3$ and thus the error is indeed proportional to h^4. The

measured decay of the d coefficients for the case of a Gaussian is shown in Figure 12.

This relation between the error in the expansion of a function and the number of vanishing moments is not only valid for interpolating wavelets but also holds true for other wavelet families.

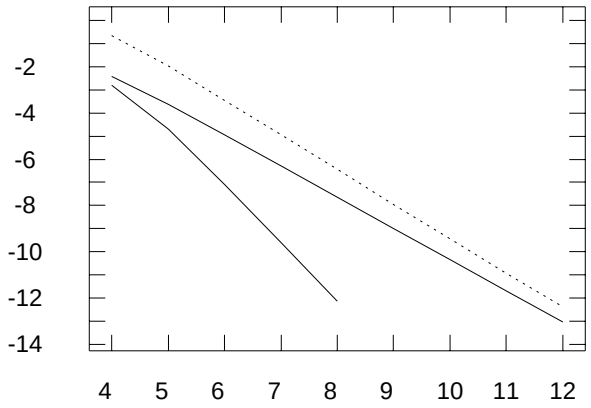


Figure 12: *The decay of the d coefficients as a function of their resolution level on a double logarithmic scale. The solid lines show the result for a 4-th and 8-th order interpolating wavelet, the dashed line is for the 8-th order Daubechies family.*

13 Dynamic versus static data compression

Wavelets were originally designed to compress data. A typical application is a 2-dim picture. On most pictures there are regions with very little variation (such as a blue sky) were the pixel resolution is an overkill. Since in the case of a picture we have only pixels which just have to be stored but need not be used in other more complicated operations such as calculating derivatives, you can consider the pixel values just as the s coefficients. Obviously this mapping does not give you any compression. However if you transform to the wavelet representation you might hope that many of the d coefficients for low variation regions are very small and can thus be neglected, reducing the number of coefficients needed to reconstruct the picture.

Dynamic data compression has the advantage that one does not need to know a priori in which regions high resolution will be required. On the other hand it will not work if one has data sets (or rather functions) that have small regions were much more (we mean by several orders of magnitude) resolution is needed than in most parts of the data set. In this case the initial data set of the s coefficients would be tremendous since, the resolution (or the

pixel size) would be determined by these tiny features in a very small region. In the applications we have in mind, you will find a huge variation in the required resolution, but you will fortunately be able beforehand to identify the regions where very high resolution is needed. An example are quantum-chemistry calculations where the regions which need high resolution are the core regions of the atoms. The higher the nuclear charge of the atom is, the more smaller and smaller core shells it will have, requiring more and more resolution as one approaches the nucleus. Static data compression is therefore the method of choice. With static data compression we can essentially map the functional values of a nonuniform grid as shown in Figure 13 onto a wavelet representation. How to do this will be described in the next section.

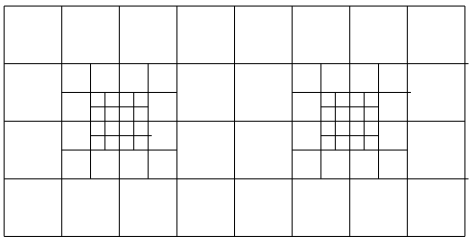


Figure 13: *A grid with 2 centers of increased resolution for a compact representation of nonuniform data sets.*

Exercise

- Expand your favorite function (given by a data set on an equally spaced grid) in a wavelet basis and check the dynamic compression ratio for an acceptable error.

14 Expanding nonuniform data in wavelets

In the case of data sets on nonuniform grids, we can not calculate the d coefficients by wavelet transforms from the s coefficients, but we have to calculate them directly by integration using (55). As was mentioned above this is easy in the case of interpolating wavelets. As follows from Equations (48) and (49), one just needs the functional values at the data point at which the wavelet will be centered and a few data points at one coarser resolution level around this center. In the case of the 4-th order interpolating wavelet one needs 2 coarser data points to the right and 2 to the left. One just has to make sure that each resolution level is broad enough to furnish the necessary

number of coarse grid points as illustrated in Figure 14 for the case of an inhomogeneous grid with 3 levels of resolution.

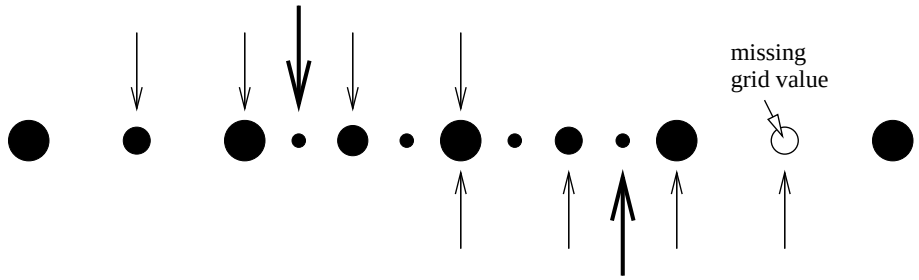


Figure 14: *An inhomogeneous grid with 3 levels of resolution. Grid points belonging to different levels of resolution are denoted by filled circles of varying radius. To calculate the wavelet expansion coefficients of a wavelet centered at the positions indicated by the fat arrows, the grid values on the grid points indicated by thin arrows coming from the same direction as the fat arrow are needed. In the case of the arrows pointing downwards, this is possible since the high resolution region is adjacent to a sufficiently broad medium resolution region. In the case of the arrows pointing upwards, this is however not possible since the high resolution region is immediately neighboring a low resolution region.*

Let us now look at an example. We want to expand the function f,

$$f(x) = \sum_{l=1}^{8} \exp\left(-(xl)^2\right) \ ,$$

in a wavelet basis. Evidently this function exhibits 8 different length scales. If one expands one simple Gaussian $\exp(-x^2)$ in a 4-th order interpolating wavelet basis with a resolution of $1/16$, one gets a reasonably small error of 10^{-6}. For the multi-scale function f, this error increases to more than 10^{-2} with the same resolution. If one however uses a data compression scheme where one uses 32 wavelets on additional 5 (it turns out that the expected 8 additional levels are not all needed) resolution levels one can again represent the function with an error of roughly 10^{-6}. The errors for these 3 cases are plotted in Figure 15. The total number of coefficients needed to represent the function in the interval $[-2; 2]$ is then 4×16 coefficients for the equal resolution $(1/16)$ scaling function part plus 5×32 coefficients for the resolution enhancement with the wavelets, which makes all together 224 coefficients. This has to be compared with the 1024 scaling function coefficients that would be needed to represent the function over the whole interval

with the maximum resolution of (1/256), which we have obtained around the origin with this static data compression scheme.

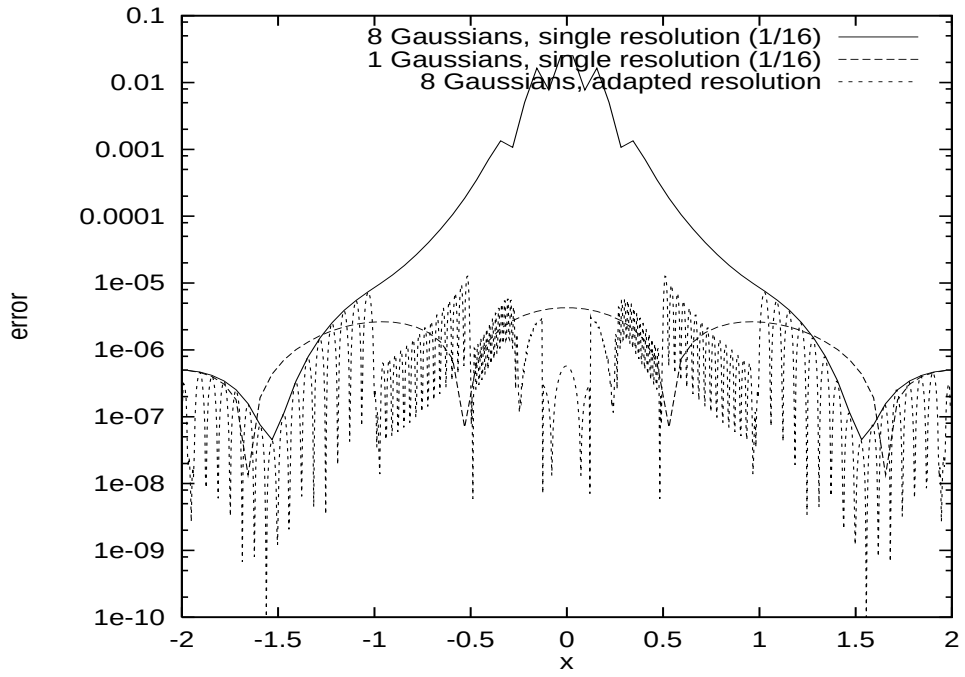


Figure 15: *Error for a static data compression scheme of a multi-scale function as described in the text. In the case of the first two curves only the upper envelope of the error is shown, in the third case the full error with all its oscillations is shown.*

15 Lifted wavelets

Lifting is a very useful technique to modify an existing family of wavelets to meet specific needs. We will discuss how to use lifting to generate interpolating wavelets which have several vanishing moments M_l

$$M_l = \int \psi_j^K(x)\, x^l dx \; .$$

The significance of these moments will be discussed later.

The key idea is to write a lifted wavelet Ψ as a linear combination of the old wavelet ψ and the basic scaling function ϕ. The scaling function will not be modified in this process. If we want to construct an interpolating wavelet with vanishing zero-th and first moment the linear combination is:

$$\Psi(x) = -\tfrac{1}{4}\phi(x-1) + \psi(x) - \tfrac{1}{4}\phi(x) \; .$$

The zero-th moment vanishes because $\int \psi(x)dx = \int \phi(2x-1)dx = \frac{1}{2}\int \phi(x)dx$. Remembering that $\psi(x)$ is symmetric about $x = \frac{1}{2}$, the first moment vanishes by symmetry. The refinement equation for the lifted wavelet is given by

$$\begin{aligned}\Psi(x) &= -\frac{1}{4}\sum_j h_j\phi(2x-j-2) + \sum_j g_j\phi(2x-j) - \frac{1}{4}\sum_j h_j\phi(2x-j) \\ &= \sum_j \left(-\frac{1}{4}h_{j-2} + g_j - \frac{1}{4}h_j\right)\phi(2x-j)\,. \end{aligned} \tag{56}$$

From the refinement equation it follows, that the new filter G associated with the lifted wavelet is given by

$$G_j = -\tfrac{1}{4}h_{j-2} + g_j - \tfrac{1}{4}h_j \tag{57}$$

For the case of our 4-th order interpolating wavelet we obtain

```
-1/4   {  0   , 0 ,-1/16 ,  0  , 9/16 ,  1   , 9/16 , 0 ,-1/16}
  +    {  0   , 0 ,  0   ,  0  ,  1   ,  0   ,  0   , 0 ,  0  }
-1/4   {-1/16 , 0 , 9/16 ,  1  , 9/16 ,  0   ,-1/16 , 0 ,  0  }

   =   { 1/64 , 0 ,-1/8  ,-1/4 , 23/32, -1/4 ,-1/8  , 0 ,-1/64}
```

The remaining filter $\tilde{H}$ follows from the symmetry relations and we have thus completely specified this new lifted wavelet family. It is easy to see, that the lifted wavelet family satisfies all the orthogonality relations. Plots of this wavelet family are given in Figure 16. It can be seen, that the dual scaling function and wavelet are not anymore related to delta functions. The filters for some other degree lifted wavelets are given in the Appendix.

The g filter of the original wavelet had the nice property, that all elements with the exception of one were zero. This together with the properties of the filter h gave rise to an operation count in a backward wavelet transform which was less than the $4mn$ that one would expect from Section 6. The filter G does not anymore have this property. One can however again come up with a particularly efficient wavelet transform [13] if one does not explicitly construct the lifted filters (as done above), but if one applies first the original filters and then synthesizes the data according to Equation (57). Exactly the same technique can also be used for the forward transform.

Exercise

- 10) Construct and plot the 6-th order lifted interpolating wavelet family.

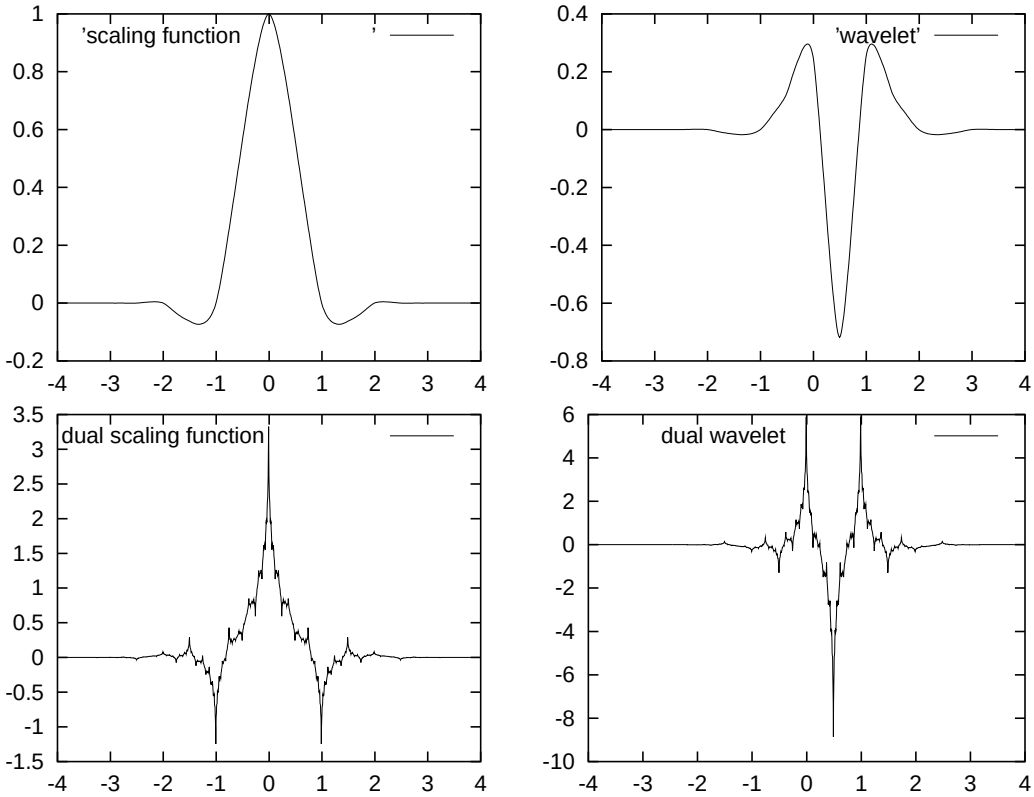


Figure 16: *The 4 fundamental functions of the 4-th order lifted wavelet family. The dual functions are actually not really "plottable" in this case since they are not smooth enough. The values of the dual scaling function at the origin for instance will keep growing as one increases the resolution.*

16 Second generation wavelets

The lifted interpolating wavelets are the easiest case of the so-called second generation wavelets. Second generation wavelets allow you to define interpolation procedures on more complicated nonuniform grid structures. The filter coefficients can consequently be position and level dependent and one can construct wavelets adapted to certain geometries, such as finite intervals or the surface of a sphere [5]. Obviously non-constant filters make things more complicated. On the other hand, it might be possible to find even more compact representations by constructing physically motivated wavelet families, incorporating for instance the spherical symmetry of the core regions of atoms in a molecule.

17 Wavelet based smoothing of a function

If you have a wavelet representation of a function and you drop a level of high resolution wavelet coefficients you will obviously smooth the function. Frequently you want to conserve certain moments $\int f(x)x^l dx$ of your function. If your function represents a charge distribution you probably would like to conserve the zero-th moment (total charge) as well as the dipole moment. The smooth version of the function has the same first $lmax$ moments as the original function if the wavelet moments M_l vanish for $l = 0, ..., lmax$. The smoothening process is illustrated in Figure 17 .

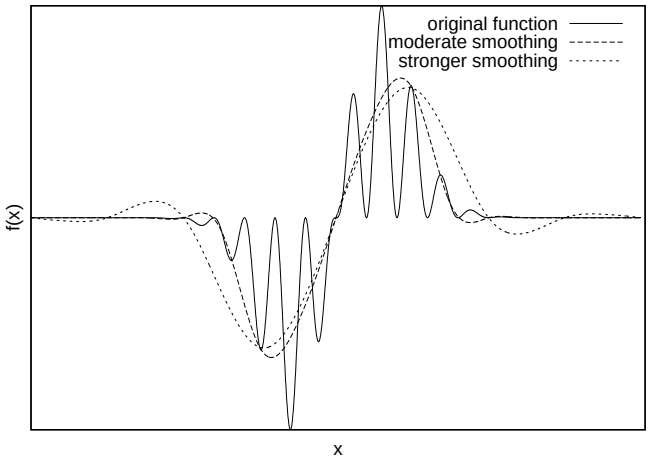


Figure 17: *A function and its smoothed versions which are obtained by discarding successive levels of wavelet coefficients. An eight order lifted interpolating wavelet was used and the first two moments are thus conserved. Note the long range oscillations which are induced in the strongly smoothed version by this moment conservation.*

18 The Fourier spectrum of wavelets

The popularity of plane waves for the solution of Poisson's and Schrödinger's equation is due to two facts. First, because there exists a fast transform algorithm and second, because differential operators are diagonal in this representation. For wavelets we have already demonstrated the existence of a fast transform and we will now discuss their localization properties in Fourier space. A good localization of wavelets in Fourier space means that differential operators are diagonally dominant. One can then use a simple diagonal preconditioning scheme in the iterative solution of the linear systems. This is a very essential point, because iterative methods for large systems without a

good preconditioner require a excessive number of iterations and make such methods prohibitive. Obviously any localized function is always a superposition of an infinite number of Fourier components. So one can just ensure, that the Fourier components decay sufficiently rapidly towards the origin and towards infinity. The Fourier components $F(k)$ are defined by

$$F(k) = \int \psi(x) e^{-ikx} dx \; .$$

As is well known the decay of the Fourier components toward infinity depends on the smoothness of the function. If l derivatives are continuous, $F(k)$ will decay as $1/k^{l+1}$. The magnitude around the origin is mainly determined by how many derivatives at the origin vanish. The derivatives are given by

$$\left. \frac{\partial^l}{\partial k^l} F(k) \right|_{k=0} = (-i)^l \int \psi(x) x^l dx = (-i)^l M_l \; .$$

If the first l moments are zero $F(k)$ will therefore behave as k^l. For large l, $F(k)$ will thus be small around the origin. So we see that if we want a wavelet with good Fourier space localization it has both to be smooth and it has to have several vanishing moments. Figure 18 shows the Fourier spectrum of a 4-th order lifted interpolating wavelet which has vanishing zero-th and first moment and of a 8-th order Daubechies wavelet which has 4 vanishing moments.

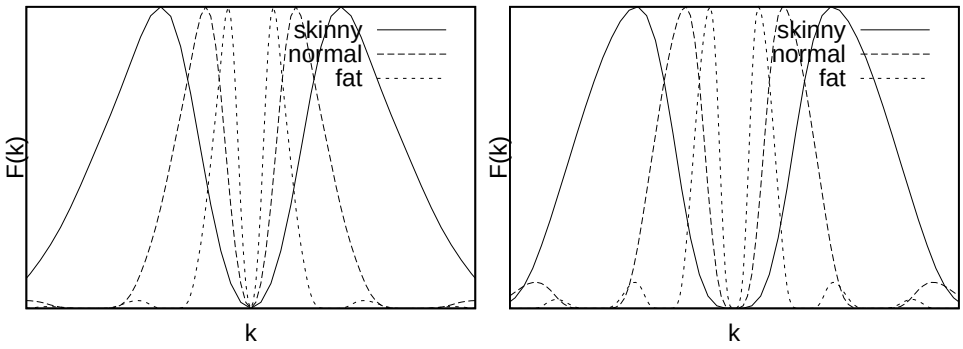


Figure 18: *Fourier spectrum of a 4-th order lifted interpolating wavelet (right panel) and 8th order Daubechies wavelet (left panel). The spectrum $F(K)$ that is shown for 3 wavelets on neighboring resolution levels exhibits reasonable frequency separation. At the peak of $F(K)$ of the medium resolution wavelets the spectra of the other two wavelets are much smaller.*

19 Wavelets in 2 and 3 dimensions

The easiest way to construct a wavelet basis in higher dimensional spaces is by forming product functions [3]. For simplicity of notation we will concentrate on the 2-dimensional case, the generalization to higher dimensional spaces being obvious. The space of all scaling functions of resolution level k is given in the 2-dimensional case by

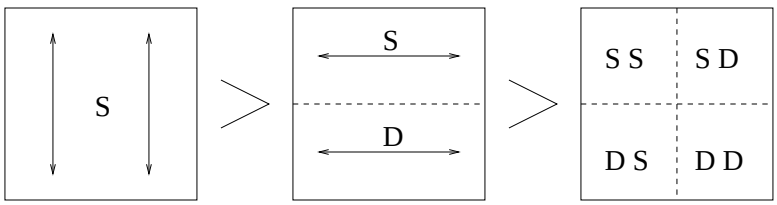
$$\phi^k_{i1,i2}(x,y) = \phi^k_{i1}(x)\phi^k_{i2}(y)\,. \tag{58}$$

The wavelets consist of three types of products

$$\psi[sd]^k_{i1,i2}(x,y) = \phi^k_{i1}(x)\psi^k_{i2}(y)\,, \tag{59}$$
$$\psi[ds]^k_{i1,i2}(x,y) = \psi^k_{i1}(x)\phi^k_{i2}(y)\,, \tag{60}$$
$$\psi[dd]^k_{i1,i2}(x,y) = \psi^k_{i1}(x)\psi^k_{i2}(y)\,. \tag{61}$$

In a 3-dimensional space the scaling functions are correspondingly of $[sss]$ type and there are 7 different classes of wavelets denoted by $[ssd]$, $[sds]$, $[sdd]$, $[dss]$, $[dsd]$, $[dds]$ and $[ddd]$.

It is easy to see, that both the many-dimensional scaling functions and wavelets satisfy refinement and orthogonality relations that are obvious generalizations of the 1-dimensional case. A wavelet transform step in the 2-dimensional setting is done by first transforming along the x and then along the y direction (or vice versa) as shown below.



To do a full 2-dim wavelet analysis one has to do a series of analysis steps. In each step the size of the active data set is reduced by 1/4 as shown in Figure 19. The total numerical effort therefore scales again linearly as in the one-dimensional case.

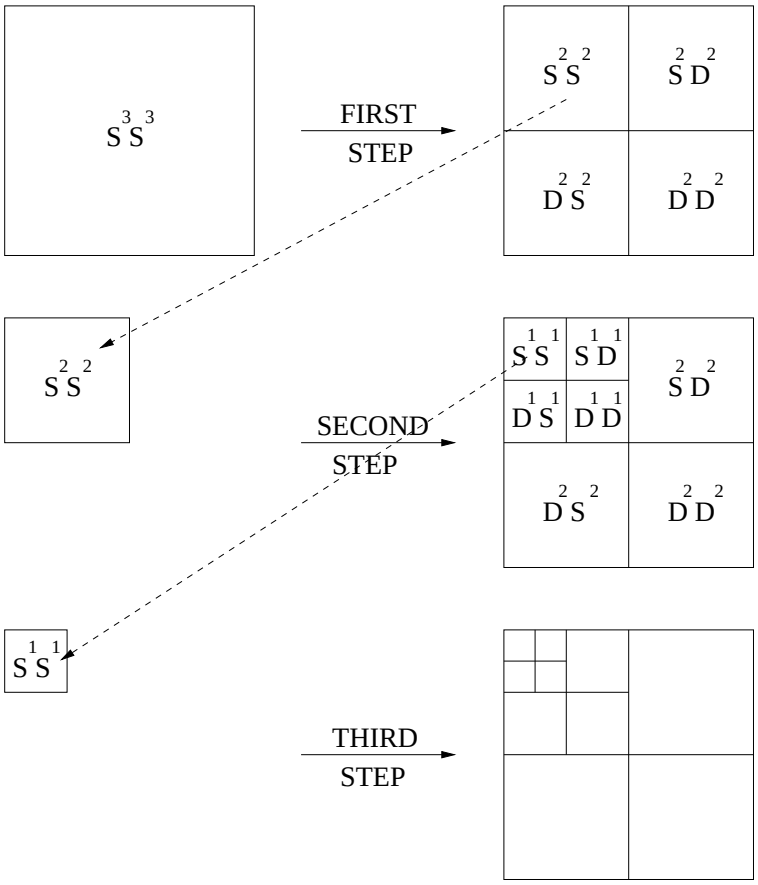


Figure 19: *A full 2-dim wavelet analysis transformation.*

To understand the meaning of the three different types of wavelets in two dimensions let us consider the picture shown at the top of Figure 20 together with its wavelet decomposition shown in the lower part. From Equation (59) we can predict that wavelets of the $[sd]$ type will pick up variations along vertical directions. Consequently the SD type coefficients are large in all the regions with strong vertical variations. In the same way the DS type coefficients are large in regions with strong horizontal variations and the DD type coefficients in regions with strong diagonal variations. It is also apparent from Figure 20 that less ink is needed to plot the wavelet decomposed picture than the original picture. This confirms the well known fact that wavelets are very efficient to compress data.

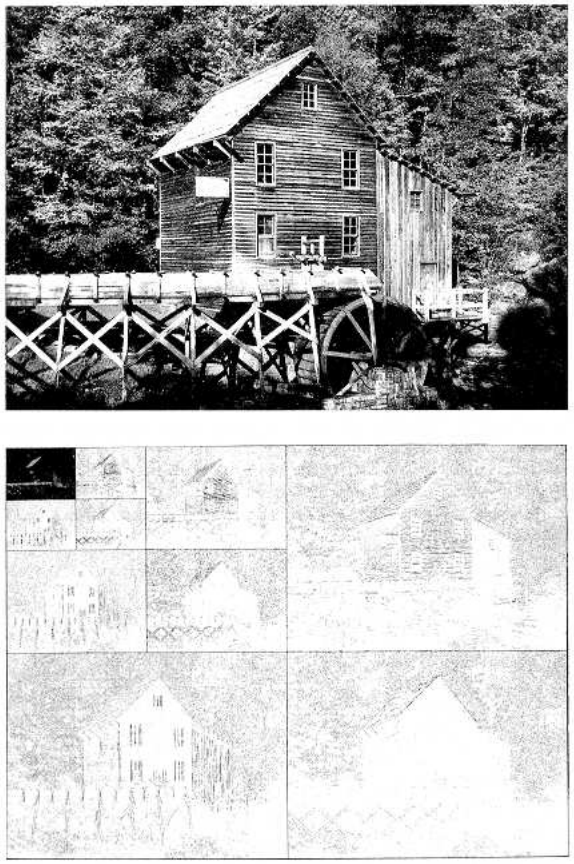


Figure 20:

Figure20: *Illustration of the meaning of the different types of wavelets in two dimensions. (Reprinted with permission from Daubechies, I.,* Ten Lectures on Wavelets. *Copyright 1992 by the Society for Industrial and Applied Mathematics. All rights reserved)*

20 Wavelet grids in higher dimensions

Wavelet based methods allow for highly flexible grid structures. One can associate each additional resolution level with a new rectangular real space grid level where the resolution has been doubled (i.e. grid spacing cut into half). In principle one can activate each level in any arbitrarily shaped regions as shown in Figure 21. To simplify programming, we used however only rectangular refinement grid structures.

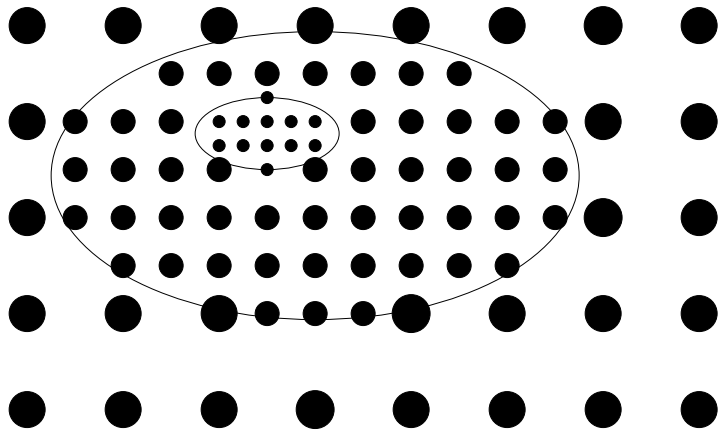


Figure 21: *Refinement regions are allowed to have very general shapes in wavelet based methods*

21 The standard operator form

In a biorthogonal wavelet basis it is natural to solve a differential equation in the collocation sense. Let us recall that in the collocation method one has two functional spaces, the space of the basis function which are used to represent the solution and the space of the test functions which are used to multiply the differential equation from the left to obtain a linear system of equations. In our case the expansion set are the scaling functions and wavelets while the test set are their dual counterparts. Let us consider the case of Poisson's equation

$$\nabla^2 v = -4\pi n \,. \tag{62}$$

Given the expansion of the charge density n in a wavelet basis,

$$n(x) = \sum_j s[n]_j^{Kmin}\, \phi_j^{Kmin}(x) + \sum_{K=Kmin}^{Kmax} \sum_j d[n]_j^K\, \psi_j^K(x)\,, \tag{63}$$

we are looking for the wavelet expansion coefficients of the potential v,

$$v(x) = \sum_{K=Kmin}^{Kmax} \sum_j d[v]_j^K \; \psi_j^K(x) + \sum_j s[v]_j^{Kmax} \; \phi_j^{Kmax}(x) \; . \tag{64}$$

After plugging in the expansion for n and v (Equation (63) and (64)) in Poisson's equation (62), multiplying from the left with the dual wavelet collocation test space and integrating, we obtain a system of equations,

$$\mathbf{A}\mathbf{v} = \mathbf{n} \, , \tag{65}$$

where $\mathbf{v}$ is the vector containing both the s and d coefficients of the potential and $\mathbf{n}$ is the same vector for the charge density n. The matrix $\mathbf{A}$ represents the Laplacian in this wavelet basis and one says that it has standard form. The standard form is the "natural" representation of an operator in a wavelet basis. The standard form is graphically shown in Figure 22.

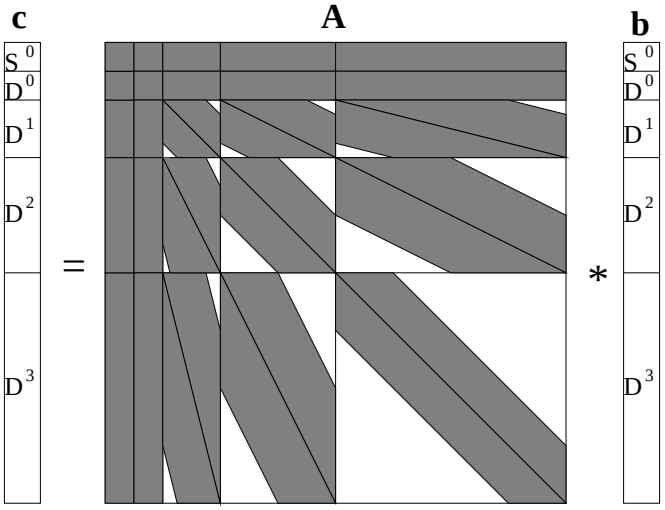


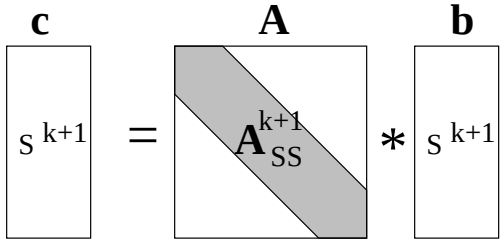
Figure 22: *The structure of a matrix in the standard form. The parts containing nonzero entries are shaded.*

As one sees, there is coupling between all resolution levels and one has to calculate many different types of matrix elements corresponding to all possible products of wavelets and scaling functions at different resolution levels and positions. The scaling function(s) at the resolution level $Kmin$ will for instance have a non-vanishing overlap with (nearly) all wavelets at high resolution. The block S^0, D^3 is therefore completely shaded in Figure 22. This complicated structure makes it inefficient for numerical operations needed for the iterative solution of our equations.

22 The non-standard operator form

As will be explained in more detail later iterative techniques will be used to solve Poisson's equation. These iterative techniques are based on matrix vector multiplications of the type $\mathbf{c} = \mathbf{A}\mathbf{b}$. How to use the vectors $\mathbf{b}$ and $\mathbf{c}$ is determined by the iterative algorithm (such as conjugate gradient algorithm) and will not be discussed here. The required matrix times vector multiplication can now be done much more easily by using the so-called non-standard [21] operator form as will be described in the following.

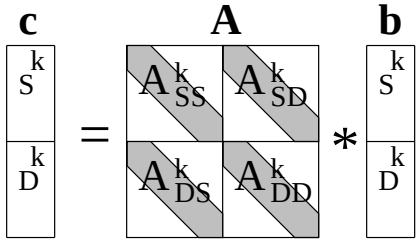
To derive the non-standard operator form let us first assume, that both vectors $\mathbf{c}$ and $\mathbf{b}$ are given in a scaling function basis. The matrix elements $A_{i,j}$ are then given by $\int \tilde{\phi}_i^k(x)\nabla^2\phi_j^k(x)dx$. Most of them will actually be zero because the basis function have zero overlap. Graphically we can represent the matrix in the following way:



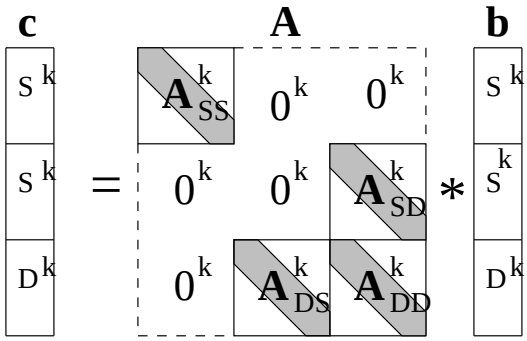
Now we can of course perform one step of a forward wavelet transform on all our data, i.e. both on the vector to be multiplied with the matrix and on the output vector which is the result of this matrix times vector multiplication. Correspondingly we have then to transform the matrix $\mathbf{A}$ as follows using the matrices defined in Equation (45)

$$\tilde{\mathbf{F}}\mathbf{c} = (\tilde{\mathbf{F}}\mathbf{A}\mathbf{F}^T)(\tilde{\mathbf{F}}\mathbf{b}) \, .$$

Using again the same notation for the transformed quantities (i.e. $\mathbf{c} \leftarrow \tilde{\mathbf{F}}\mathbf{c}$; $\mathbf{F} \leftarrow \tilde{\mathbf{F}}\mathbf{A}\mathbf{F}^T$; $\mathbf{b} \leftarrow \tilde{\mathbf{F}}\mathbf{b}$) we obtain the following data structures:



If we continued to recursively apply wavelet transforms to the remaining s parts we would obviously obtain the standard operator form. To get the nonstandard form, we have to add another step where we artificially enlarge our matrix $\mathbf{A}$ by putting in 5 blocks of zeroes as shown below:



We see that our input and output vectors **b** and **c** also have to be adapted to this matrix structure leading to a redundant data structures. In the input vector **b** we have to duplicate the S data set by copying it from the middle part to the upper part in the Figure above. To get back the correct nonredundant output vector **c** we have to add the two S parts of the redundant vector **c** in the Figure above.

We can now recursively apply this 2-step procedure on the SS block of the resulting matrices. Doing this we obtain the so called non-standard form, which is graphically visualized in Figure 23.

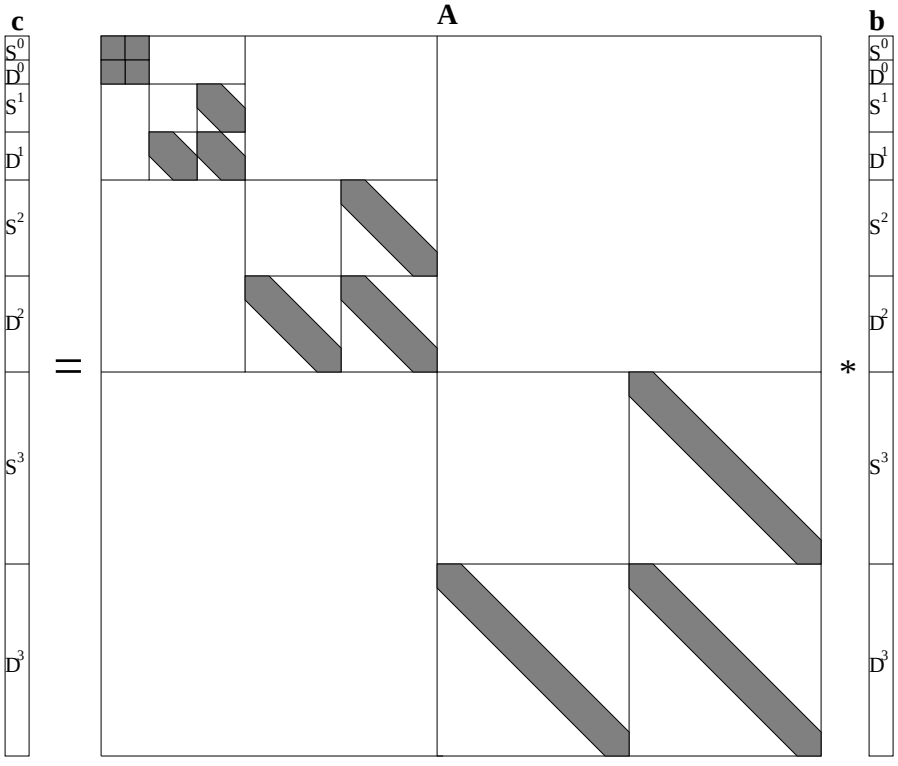


Figure 23: *The structure of a matrix in the non-standard form.*

As we see, the non-standard matrix structure completely decouples different resolution levels, since there are no blocks between different levels. The coupling between different levels just enters trough the wavelet transforms which have to be performed at the beginning and at the end of the non-standard operator application to generate the redundant input vector and to reduce the redundant output vector to a non redundant form. To generate the redundant input vector **b** of Figure 23, we have to generate S^1 from S^0 and D^0, then S^2 from S^1 and D^1 and finally S^3 from S^2 and D^2. The reduction of the output vector **c** is done by splitting up S^3 into S^2 and D^2 and adding this result to the S^2 and D^2 parts on top which are the output of the preceeding matrix times vector multiplication. Next one splits up this modified S^2 part in a S^1 and D^1 part and adds the result again to the existing contributions. Finally the S^1 part is splitted and added to the existing S^0 and D^0 parts.

We also see that all the nonzero blocks of this nonstandard matrix representation are strictly banded and the application of this matrix to a vector scales therefore linearly. The structure of the matrix in Figure 23 is primarily valid for the case of uniform resolution where all the possible d coefficients at the highest resolution level are nonzero. It can however easily be seen that this nonstandard form retains its advantage in a case of varying resolution where only some of the d coefficients are nonzero. If the nonredundant input data set is sparse, the redundant input data set will be sparse as well. Since all the blocks are banded, the redundant output set will be sparse as well. Finally the nonredundant output set will then be sparse as well. Hence the total scaling is linear in this case as well.

23 Calculation of differential operators

As we have seen in the preceeding chapter we need the matrix elements

$$\int \tilde{\phi}_i^k(x) \frac{\partial^l}{\partial x^l} \phi_j^k(x)\, dx\,, \tag{66}$$

$$\int \tilde{\psi}_i^k(x) \frac{\partial^l}{\partial x^l} \phi_j^k(x)\, dx\,, \tag{67}$$

$$\int \tilde{\phi}_i^k(x) \frac{\partial^l}{\partial x^l} \psi_j^k(x)\, dx\,, \tag{68}$$

$$\int \tilde{\psi}_i^k(x) \frac{\partial^l}{\partial x^l} \psi_j^k(x)\, dx\,, \tag{69}$$

to set up the SS, DS, SD and DD parts of the non-standard operator form. Matrix elements on different resolution levels k are obviously related

by simple scaling relations. For instance

$$\int \tilde{\phi}_i^{k+1}(x)\frac{\partial^l}{\partial x^l}\phi_j^{k+1}(x)dx = 2^{l-1}\int \tilde{\phi}_i^{k}(x)\frac{\partial^l}{\partial x^l}\phi_j^{k}(x)dx\ . \tag{70}$$

So we just have to calculate these 4 matrix elements for one resolution level. On a certain resolution level, we can use the refinement relations to express the matrix elements involving wavelets in terms of matrix elements involving scaling functions (at a higher resolution level) only. Denoting the basic integral by a_i, where

$$a_i = \int \tilde{\phi}(x)\frac{\partial^l}{\partial x^l}\phi(x-i)dx\ , \tag{71}$$

we obtain

$$\begin{aligned}
\int \tilde{\phi}_i(x)\frac{\partial^l}{\partial x^l}\phi_j^1(x)dx &= a_{j-i}\ , &(72)\\
\int \tilde{\psi}_i(x)\frac{\partial^l}{\partial x^l}\phi_j^1(x)dx &= 2^{l-1}\sum_{\nu,\mu}\tilde{g}_\nu h_\mu a_{2j-2i+\mu-\nu}\ , &(73)\\
\int \tilde{\phi}_i(x)\frac{\partial^l}{\partial x^l}\psi_j^1(x)dx &= 2^{l+1}\sum_{\nu,\mu}\tilde{h}_\nu g_\mu a_{2j-2i+\mu-\nu}\ , &(74)\\
\int \tilde{\psi}_i(x)\frac{\partial^l}{\partial x^l}\psi_j^1(x)dx &= 2^{l}\sum_{\nu,\mu}\tilde{g}_\nu g_\mu a_{2j-2i+\mu-\nu}\ . &(75)
\end{aligned}$$

To calculate a_i we follow Beylkin [20]. Using the refinement relations Equations (14) and (16) for ϕ and $\tilde{\phi}$ we obtain

$$\begin{aligned}
a_i &= \int \tilde{\phi}(x)\frac{\partial^l}{\partial x^l}\phi(x-i)dx &(76)\\
&= \sum_{\nu,\mu} 2\tilde{h}_\nu h_\mu \int \tilde{\phi}(2x-\nu)\frac{\partial^l}{\partial x^l}\phi(2x-2i-\mu)dx\\
&= \sum_{\nu,\mu} 2\tilde{h}_\nu h_\mu 2^{l-1} \int \tilde{\phi}(y-\nu)\frac{\partial^l}{\partial y^l}\phi(y-2i-\mu)dy\\
&= \sum_{\nu,\mu} \tilde{h}_\nu h_\mu 2^{l} \int \tilde{\phi}(y)\frac{\partial^l}{\partial y^l}\phi(y-2i-\mu+\nu)dy\\
&= \sum_{\nu,\mu} \tilde{h}_\nu h_\mu\ 2^l\ a_{2i-\nu+\mu}
\end{aligned}$$

We thus have to find the eigenvector $\mathbf{a}$ associated with the eigenvalue of 2^{-l},

$$\sum_j A_{i,j}\ a_j = \left(\frac{1}{2}\right)^l a_i\ , \tag{77}$$

where the matrix $A_{i,j}$ is given by

$$A_{i,j} = \sum_{\nu,\mu} \tilde{h}_\nu h_\mu \, \delta_{j,2i-\nu+\mu} \, . \tag{78}$$

As it stands this eigensystem has a solution only if the rang of the matrix $A - 2^{-l} I$ is less than its dimension. For a well defined differential operator, i.e if l is less than the degree of smoothness of the scaling function this will be the case (for the 4-th order interpolating wavelet family the second derivative is for instance not defined). The system (77) is numerically unstable and it is therefore better to solve it using symbolic manipulations with a software such as Mathematica instead of solving it numerically.

The system of equations (77) determines the a_j's only up to a normalization factor. In the following, we will therefore derive the normalization equation. For simplicity, we will give the derivation only for the case of interpolating polynomials, even though the final result Equation (81) will hold in the general case.

From the normalization of the scaling function and from elementary calculus, it follows that

$$\int \phi(x) \frac{\partial^l}{\partial x^l} x^l \, dx = \int \phi(x) \, l! \, dx = l! \, . \tag{79}$$

On the other hand we know, that we can expand any polynomial of low enough degree exactly with the interpolating polynomials. The expansion coefficients are just i^l. So we obtain

$$\int \phi(x) \frac{\partial^l}{\partial x^l} \sum_i i^l \phi(x - i) = \sum_i i^l a_i \, . \tag{80}$$

By comparing Equation (79) and (80) we thus obtain the normalization condition

$$\sum_i i^l a_i = l! \, . \tag{81}$$

The interpolating wavelet family offers also an important advantage for the calculation of differential operators. Whereas in general derivative filters extend over the interval $[-2m; 2m]$ most of the border elements of interpolating wavelets are zero and their effective filter length is only $[-m + 2; m - 2]$.

Derivative filter coefficients for several families are listed in the Appendix.

Exercises

- 11) Generalize the proof of the normalization condition (81) using the theorem from section (10).

- 12) Show that the derivative filter length is $[-m+2; m-2]$ as claimed above.

- Calculate the derivative of a 1-dim Gaussian in an interpolating scaling function basis and compare your result with the derivative obtained by finite differences.

24 Differential operators in higher dimensions

As was pointed out before, higher dimensional wavelets can be constructed as products of one dimensional wavelets. The matrix elements of differential operators can therefore easily be derived.

Let us consider as an example the matrix elements of $\frac{\partial}{\partial x}$ with respect to the 2-dimensional scaling functions,

$$\int \tilde{\phi}^k_{i1,i2}(x,y)\frac{\partial}{\partial x}\phi^k_{j1,j2}(x,y) = \int \tilde{\phi}^k_{i1}(x)\tilde{\phi}^k_{i2}(y)\frac{\partial}{\partial x}\phi^k_{j1}(x)\phi^k_{j2}(y) = \delta_{i2-j2}a_{j1-i1} \; .$$

The remaining matrix elements among the wavelets and scaling functions can be derived along the same lines. Obviously a differential operator acting on x will only couple functions which have the same dependence with respect to y as indicated in Figure 24.

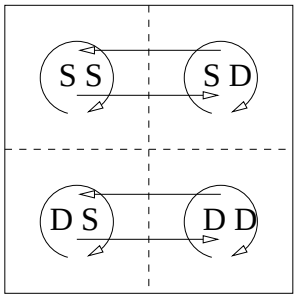


Figure 24: *The coupling of the expansion coefficients under the action of a differential operator acting along the (horizontal) x axis.*

25 Transforming between wavelet families

As we have seen, the interpolating wavelets have many advantages which makes it recommendable to use them. Nevertheless it will be necessary in some steps to use different wavelets families and the problem is how to obtain the expansion coefficients with respect to this new family if we just know the

expansion coefficients with respect to the old family. Let us denote all the quantities which are related to the old family with the usual small letters whereas the quantities related to the new family will be denoted by capital letters. To do this transformation we will again use the nonstandard operator form. As in the case of differential operators, we have to calculate 3 types of matrix elements representing the coupling between wavelets and wavelets, scaling functions and wavelets and wavelets with scaling functions. All these matrix elements can easily be calculated from the matrix elements coupling scaling functions with scaling functions. Proceeding analogously to the case of differential operators we obtain the following eigenvector problem,

$$a_i = \int \tilde{\Phi}(x)\phi(x-i)dx = \sum_{\nu,\mu} \tilde{H}_\nu h_\mu a_{2i-\nu+\mu} \; . \tag{82}$$

The normalization condition comes in this case from the requirement that

$$\int \tilde{\Phi}(x) 1 dx = 1 \; . \tag{83}$$

Considering again the case where ϕ is an interpolating scaling function, we have $1 = \sum_i \phi(x-i)$ and the normalization condition becomes

$$\sum_i a_i = 1 \; . \tag{84}$$

26 Scalar products

Scalar products can be calculated with the nonstandard operator form as well. The matrix we have to represent in this case is the overlap matrix.

$$a_i = \int \phi(x)\phi(x-i)dx = \frac{1}{2} \sum_{\nu,\mu} h_\nu h_\mu a_{2i-\nu+\mu} \; . \tag{85}$$

The normalization condition is again

$$\sum_i a_i = 1 \; . \tag{86}$$

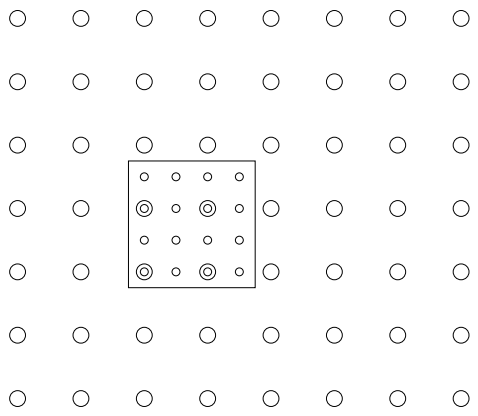
In the case of orthogonal wavelets, $a_i = \delta_i$ and the overlap matrix is the identity matrix.

27 The solution of Poisson's equation

The solution of Poisson's equation consists of several steps which will be described in the following sections. For simplicity, we will discuss here a 2-dimensional setting.

27.1 Expanding the charge density in a wavelet basis

Usually the charge density will be given on a nonuniform grid structure such as the one shown in Figure 13. The first step consists therefore of mapping such a real space data set into a wavelet expansion. The method is a generalization of the 1-dimensional procedure outlined in Section 14. If the data sets are chosen correctly this is also in the 2-dimensional case an unique and invertible mapping. A suitable grid data structure is shown below.

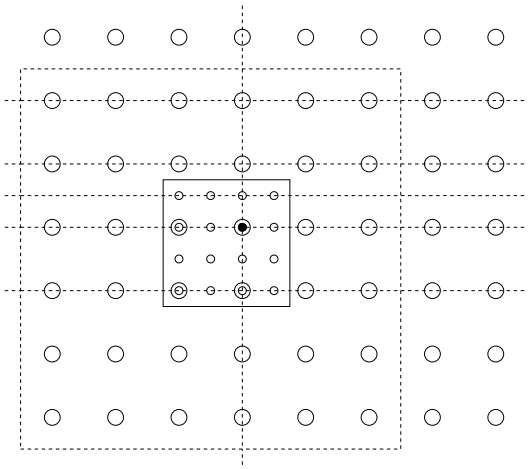


Let us first point out, why the number of wavelet coefficients one obtains is equal to the number of additional higher resolution real data points. The additional resolution grid above has 16 data points. If one does a wavelet transform one will obtain 4 wavelets of 'DS' type, 4 of 'SD' type and 4 of 'DD' type, so all together 12 coefficients. Since out of the 16 data points 4 are redundant since they are already contained in the coarse data set, the number of wavelet coefficients is indeed equal to the number of additional real space data.

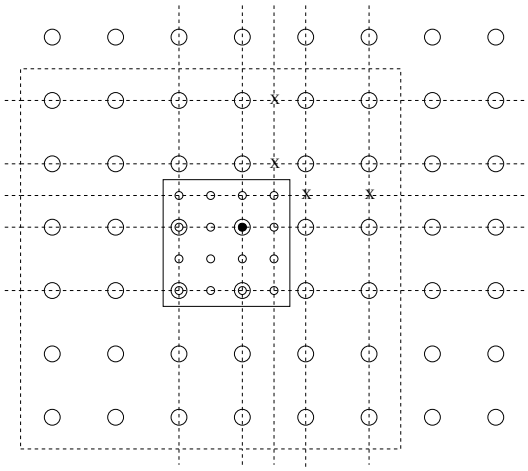
As in the 1-dimensional case the wavelet expansion coefficients are obtained by integration with the dual wavelets. It is easy to see that the dual wavelets are given by

$$\begin{aligned}
\tilde{\psi}[sd]^k_{i1,i2}(x,y) &= \tilde{\phi}^k_{i1}(x)\tilde{\psi}^k_{i2}(y)\,, \\
\tilde{\psi}[ds]^k_{i1,i2}(x,y) &= \tilde{\psi}^k_{i1}(x)\tilde{\phi}^k_{i2}(y)\,, \\
\tilde{\psi}[dd]^k_{i1,i2}(x,y) &= \tilde{\psi}^k_{i1}(x)\tilde{\psi}^k_{i2}(y)\,.
\end{aligned}$$

Remembering the form of the dual fundamental functions (Equation (49)) in the case of 4-th order interpolating wavelets, we see that we need the values at the intersection of the dotted lines in the Figure below to obtain the coefficient of the 'SD' wavelet associated with the filled grid point. Note that all the integration points which are needed are available in the data sets.



The integration points which are needed for the 'DD' coefficients associated with the filled grid point are shown in the Figure below. We see that in addition to values already available in the data sets, we still need the 4 integration points denoted by X. Fortunately these points are on lines connecting coarse grid points and can therefore be calculated by the same 1-dimensional interpolation procedure which defines our interpolating wavelet family.



It follows that in order to calculate the 12 wavelet coefficients associated with the fine grid, we just need all the data contained in the region indicated by the dashed rectangle.

Obviously the wavelet expansion defined by the above integration procedure will reproduce exactly the original values when it is evaluated for any grid point. If the grid data set represents an analytic function the values

of the wavelet expansion on points which are not elements of any grid will of course not be identical to the exact functional value. As discussed in the 1-dimensional case higher order wavelet families will give smaller deviations than low order wavelet families. The error on the finest resolution level is shown for the case of a 2-dimensional Gaussian which was expanded on several resolution levels with a 8-th order wavelet family in Figure 25.

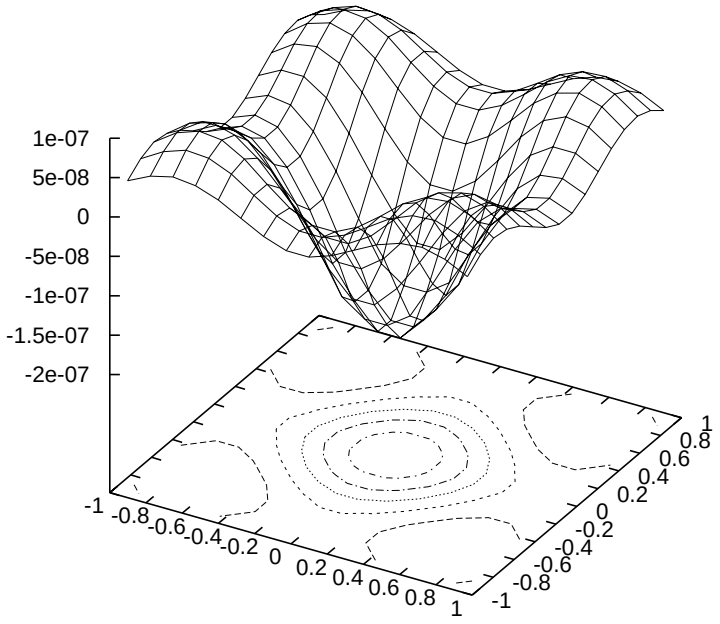


Figure 25: *The error on the finest resolution level for a 2-dimensional Gaussian expanded in 8-th order interpolating wavelets.*

27.2 Applying the Laplace operator

As was pointed out in the introduction an important goal is to obtain a linear scaling of the computational effort with respect to the size of the system, i.e. with respect to the number of scaling functions and wavelets necessary as a basis. Obviously this excludes the possibility to use direct methods for the linear systems of equations arising from a discretization of the differential equation. The usual iterative methods such as the steepest descent and conjugate gradient methods [11] have to be used instead. All these methods require just that one can calculate matrix times vector operations, the full matrix is never needed, just its effect on a vector has to be known. The nonstandard operator form outlined earlier for the 1-dimensional case fulfills this requirement. All one needs to calculate the effect of an operator on a

input vector are a few filter coefficients. The nonstandard form will now be explained in detail for a two-dimensional case with sparse data structures, i.e. where high resolution wavelet expansion coefficients are taken into account only in selected regions. A diagrammatic representation of the algorithm is shown in Figure 26.

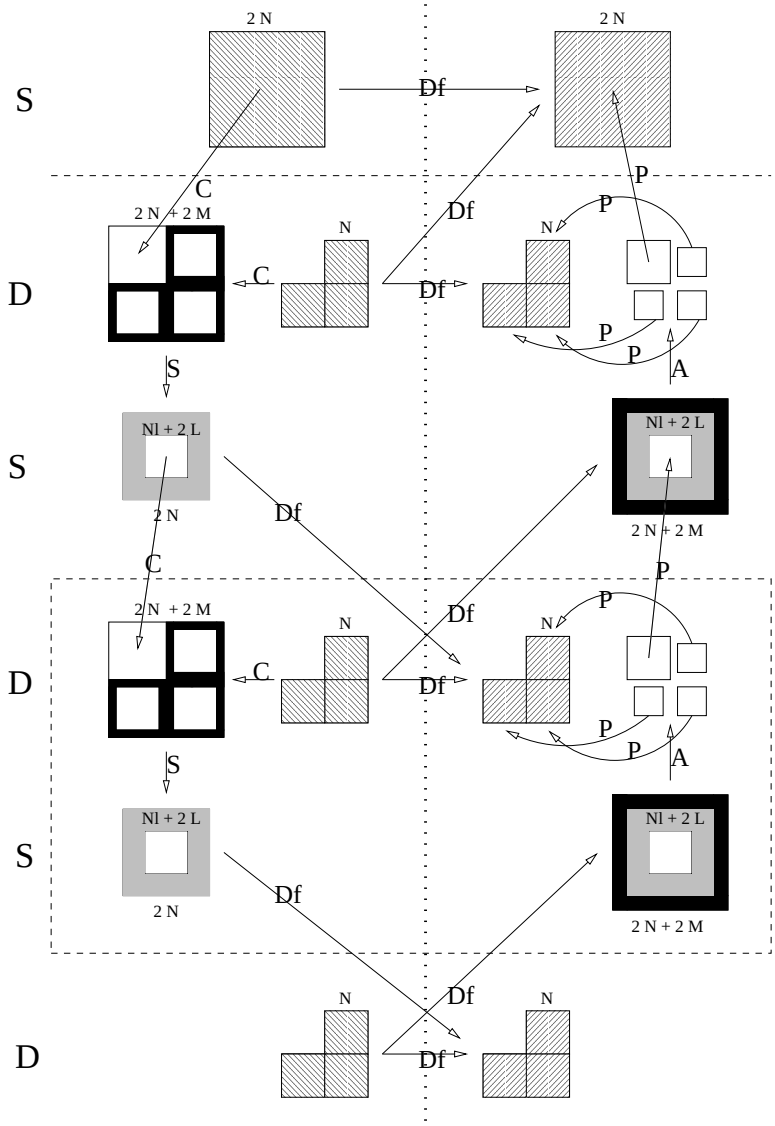


Figure 26: *A diagrammatic representation of the nonstandard operator algorithm in the 2-dimensional case with varying resolution.*

Four resolution levels are shown explicitly. If more than four levels are present, the part of the diagram in the dashed box is repeated several times. The input data are the scaling function coefficients on the uppermost periodic level and the wavelet coefficients on all the lower (in Figure 26) levels.

The output data have exactly the same structure. All the input data are contained in boxes with lines running from north-west to south-east, whereas the output data are contained in boxes with lines running from south-west to north-east. The size of the data sets is indicated at the top of the boxes. It is understood that N is resolution level dependent. In more realistic cases there will also usually be several grids on a certain resolution level which all can be of different sizes. Several abbreviations are used for the different operations. "Df" stands for differentiation, which is a convolution with the appropriate filters presented in Section 23. If one wants to use the non-standard form for any other operation one just has to replace "Df" by the appropriate operation. "S" stands for a wavelet synthesis step, "A" for a wavelet analysis step, "C" for a copy operation and "P" for an addition of the scaling function or wavelet coefficient data sets. Going down on the left hand side one creates the redundant data set necessary as the input to the non-standard operator form, i.e. from the wavelet coefficients at the same level and the scaling function coefficients at the upper (in Figure 26) level one calculates the scaling function coefficients at the present level. In the case of interpolating wavelets this set of scaling function coefficients is of course just the data set on the real space grid. From the 3 data sets containing N $\times$ N wavelet coefficients one can calculate 2N $\times$ 2N scaling function coefficients at each level. In order to get rid of boundary effects, one has however to surround the input wavelet coefficient by a layer of zeroes of thickness M, if M is the filter length for the wavelet analysis step. This is indicated by black shadowing in the diagram 26. The scaling function data set obtained in this way will account for all the scattering of the differential operator from the S type data sets into D type data sets. By using the collocation method, we have restricted our test function space to the 3 $\times$ N $\times$ N wavelets on each level. All the scattering into parts of space outside this test space can therefore be neglected. In order to get the correct scattering into the basis function space we just need (Nl+2L) S coefficients, where Nl denotes the size of the data set at the lower (in the figure) level which is the target of the scattering. Frequently we will thus not need the full S data set. The area of the S data set which is not needed is filled by small dots. For the scattering from the D to the D data sets the same remark applies. Only the part which is within the basis function space is needed. For the scattering from the D data set to the S data set the situation is different. In this case we have to calculate all the (Nl + 2L) nonzero S coefficients. In principle the filter length for the S to D and the D to S scattering can be different even though they are denoted by the same symbol L in the diagram 26. After doing all these 3 types of scattering operations we have now generated a redundant output data set and we have to reduce it to a nonredundant one. To do this we perform an analysis wavelet step on each S data set. Again in order to

get rid of boundary effect we have to embed the input data set into a set of zeroes (indicated by black shadowing). The output D coefficients are then added to the D coefficients obtained by the D to D scattering, whereas the output S coefficients are added to the corresponding upper data set.

The error one obtains by a wavelet based calculation of the Laplacian for a 2-dimensional Gaussian is shown in Figure 27.

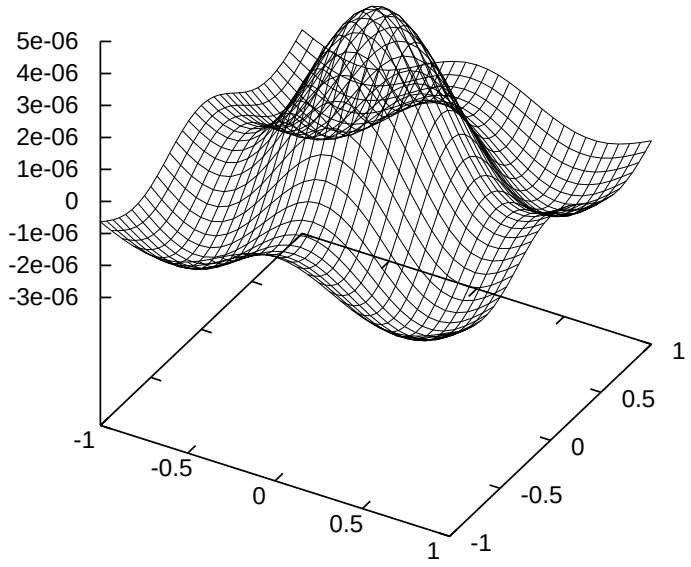


Figure 27: *The error on the finest resolution level for the Laplacian applied to a Gaussian using 8-th order interpolating wavelets.*

27.3 Preconditioning

Once we know how to apply our operator to a vector we can calculate the residual vector $\mathbf{r}$ for our linear system of equations,

$$\mathbf{r} = \mathbf{A}\mathbf{x} - \mathbf{y} \,, \tag{87}$$

where $\mathbf{x}$ is our approximate solution and $\mathbf{y}$ the right hand side of our linear system. Ordinary steepest descent and conjugate gradient method give a prescription how to find a new improved solution using the residual vectors. In most applications it turns out that the condition number of the matrix becomes worse as one goes to larger systems. As a consequence the number of iterations needed in the iterative solver increases and one has not any more a linear scaling even if the application of the operator is strictly linear.

Preconditioning can overcome this problem. One has to calculate the so-called preconditioned residual vector,

$$\mathbf{p} = \mathbf{B}^{-1}\mathbf{r}\,, \tag{88}$$

where $\mathbf{B}^{-1}$ has to be a reasonable approximation to $\mathbf{A}^{-1}$. At the same time it is also necessary that $\mathbf{B}^{-1}$ can be calculated very rapidly. Because of these 2 requirements a good preconditioning scheme is far from granted and there are indeed many methods for which no satisfactory preconditioning is known. The simplest preconditioning scheme is always a diagonal preconditioning scheme, i.e. the matrix $\mathbf{B}$ just contains the diagonal elements of $\mathbf{A}$. Such a preconditioning scheme will be efficient if the matrix $\mathbf{A}$ (in the standard form) is diagonally dominant. We know that the Laplace operator is a strict diagonal matrix in a plane wave basis. In a wavelet basis we will therefore get a diagonally dominant matrix if the wavelets are well localized in Fourier space. This localization and its relation to the number of vanishing moments of the wavelet was discussed in Section 18. It was shown that the ordinary interpolating wavelets have no good localization properties in Fourier space, whereas the lifted wavelets do. This means, that we should not do the preconditioning in the basis of interpolating wavelets but in the basis of the lifted relatives. The transformation from the unlifted to the lifted basis can again be done with the nonstandard operator form. In this case it is actually very easy. All the blocks of the nonstandard form are unit matrices. So one just has to bring the data into redundant form by a series of normal interpolating wavelet transforms and then to reduce this data set again to nonredundant form with lifted wavelet transforms. Doing preconditioning in this lifted wavelet basis, it is possible to reduce the residue (i.e. the length of the residue vector) by one order of magnitude with only 3 iterations. To stress it again, this reduction rate is completely independent of how many resolution levels or of how many basis functions one is using in the calculation.

Exercise

- 13) Show that $\int \tilde{\Phi}(x-i)\phi(x-j)dx = \delta_{i-j}$ where Φ is a member of a lifted wavelet family.

27.4 Boundary conditions

At the coarsest resolution level boundary conditions have to be specified at the surface of the computational box. Three different types of boundary conditions can easily be implemented in this context. Periodic boundary conditions are one obvious choice. In this case one just has to periodically

wrap around indices in all filtering operations whenever an index is out of bound. Another obvious boundary condition is that the solution vanishes outside the computational volume. In this case one has just to add layers of zeroes around the data sets before effectuating some filtering operations. In the case of Poisson's equation the so-called natural boundary conditions, where the potential tends to zero at infinity are however the most important ones. This boundary condition is very difficult to implement with most other methods, but can rather elegantly be implemented within the wavelet context. Since in the far region the potential is varying very slowly, less and less resolution is needed as one goes outward. By doubling both the grid spacing and the spatial extension of each additional coarse resolution layer, one can describe an exponentially large volume with a small number of basis functions. If at some point the volume is large enough, such that the potential practically vanishes at the surface, one can terminate the enlargements and use any of the above mentioned boundary conditions. How many enlargements will be needed depends of course on the multi-pole character of the charge distribution as well as on the accuracy one wants to obtain.

27.5 The electrostatic potential of a uranium dimer

To demonstrate the power of the wavelet method we applied it to the most difficult system we could think of in the area of electronic structure calculations, namely a three dimensional uranium dimer [24]. In this example, we clearly find widely varying length scales. The valence electrons have an extension of 5 atomic units, the 1s core electrons of 2/100 atomic units and the nucleus itself was represented by a charge distribution with an extension of 1/2000 atomic units. So all together the length scales varied by 4 orders of magnitude and two regions of increasing resolution (around each nucleus) were needed. In order to have quasi perfect natural boundary conditions we embedded the molecule in a computational volume of side length 10^4 atomic units. All together this necessitated 22 levels of resolution. Even though the potential itself varies by many orders of magnitude, we were able to calculate the solution with typically 7 digits of accuracy as shown in Figure 28. We believe that it would not be possible with any other method to solve this kind of benchmark problem.

27.6 Similar methods to solve Poisson's equation

Poisson's equation is one of the best studied differential equations and several new algorithms have been proposed in recent years for this equation that are similar in spirit to the present approach in the sense that they incorporate some multi-resolution concepts, even though they are not based on wavelets.

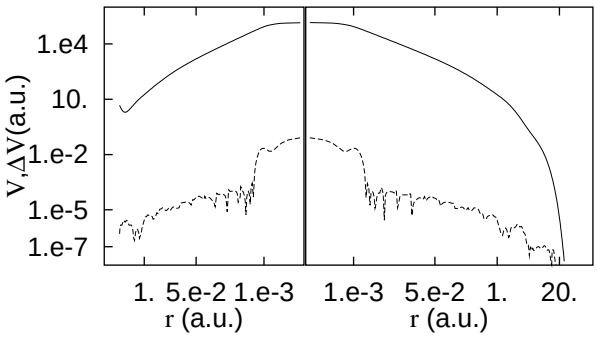


Figure 28: *The potential V (full line) and the numerical error ΔV (dashed line) for an uranium dimer as a function of the distance from of right hand side nucleus. The left panel shows both quantities on a line in the direction of the left nucleus up to the middle of the bond, the right panel shows them on a line in the opposite direction, i.e. pointing away form the left hand nucleus. Both distances are given on a logarithmic scale.*

The first algorithm in this context is the multi-grid algorithm [6]. In this algorithm correction equations for the error are solved on several resolution levels to overcome poor conditioning numbers for iterative matrix techniques on the finest grid. The solution itself is however always represented on the finest grid. The reason for this is that in the multi-grid method there are no exact refinement relations which would allow a multi-resolution representation of the solution itself. The inaccuracy one occurs by shifting the error related quantities across several levels can however be corrected by additional smoothening techniques. The wavelet method contains thus the multi-resolution concept in a more elegant and fundamental way than the multi-grid method. The algorithms of Barnes and Hut [7] and the Fast Multirole Method by Greengard [8] are algorithms with similar features as the one presented here, they however apply only to discrete systems consisting of point particles. In the context of quantum chemistry calculations using Gaussian basis sets, they have however been generalized for "Gaussian particles" [9].

28 The solution of Schrödinger's equation

The solution of Schrödinger's equation is somewhat more complicated than the solution of Poisson's equation since more basic operations are involved. In addition to the application of the Laplace operator, one still has to calculate

the effect of the potential on the wavefunction and one has to orthogonalize the wavefunctions. Using the basic building blocks discussed in the previous section, it is possible to build several algorithms. The use of interpolating wavelets for electronic structure calculations has been reviewed by Arias [16]. Other wavelet families might however also be useful and it is at present not quite clear which of these possibilities is the most efficient and we will therefore briefly discuss the advantages and disadvantages. The first question is whether to use biorthogonal or orthogonal wavelets. The advantages of the biorthogonal wavelets have been discussed at length, their disadvantage is that the application of the overlap matrix using the nonstandard operator form as outlined in the previous sections is rather expensive. If one uses an algorithm which requires many scalar products for the minimization of the total energy such as the DIIS [23] algorithm, this might turn out to be a serious drawback. In the case of orthogonal wavelets scalar products can directly (and quickly) be calculated in the wavelet representation. Another operation which has to be discussed is the application of the local potential. This can be done using a philosophy very similar to the one used in plane wave calculations. One transforms both the potential and the wavefunction into real space, takes the product on the real space grid and transforms back the product. In contrast to plane waves the real space grid however needs not be uniform. Those transformation into real space are in practice only feasible with interpolating wavelets. If one used orthogonal wavelets one would have to transform into a basis of interpolating wavelet before doing the real space operation and back afterwards, which would necessitate two extra nonstandard operator applications. Analogously to the case of plane waves one also has to take into account the aliasing problem. In the product of two quantities (here potential and wavefunction) one has frequency components at twice the frequency of the original data. In the case of plane waves one can exactly eliminate any contamination of the low frequency components by these new high frequency components by using a real space grid with twice the resolution corresponding to the minimal wavelength of the two separate quantities. Since wavelets have no strict frequency localization one can not exactly eliminate aliasing errors, but one can render them completely insignificant by using the same double resolution grid technique as in the case of plane waves. We found that correcting for aliasing errors in this way leads to a faster convergence than schemes which do not correct this error. The implementation of aliasing error correction implies also a variational treatment of the potential energy part. One might then also want to treat the kinetic energy in a variational way. Let us recall that in the case of Poisson's equation we did not use a variational scheme but the collocation scheme. Finding a variational representation of the kinetic energy is no fundamental problem. Instead of using the nonstandard form for an operator

derived from the basic integral $\int \tilde{\phi}(x)\frac{\partial^2}{\partial x^2}\phi(x)dx$ one has to use it for the integral $\int \phi(x)\frac{\partial^2}{\partial x^2}\phi(x)dx$. The drawback is that the nonstandard matrix form corresponding to the new integral is less sparse than in the original setting.

Since wavelet based methods allow you to increase practically without limitation the resolution in any subregion, it would algorithmically be possible to do always all electron calculations and thus circumvent the need of pseudo-potentials. Physically this would however not be a good idea since pseudo-potentials are the easiest method to account for relativistic effects in the deep core states of heavy elements. So pseudo-potentials will certainly also be used in the wavelet context. Wavelet based methods would however offer the flexibility to use harder and therefore more transferable pseudo-potentials. One might then envision electronic structure calculations where one has three levels of resolution. High resolution in the core region of tough elements such as transition metals or fluor, medium resolution in the regions of ordinary valence bonding and low resolution in the tail region of a molecule. Many traditional pseudo-potentials are specifically tailored to plane wave calculations and are therefore not useful in the wavelet context. Purely real space pseudo-potentials [10] have however been developed which could immediately be used within the wavelet framework.

Another important quantity in electronic structure calculations are the forces acting on the nuclei. For position dependent basis sets there are the so-called Pulay forces which have to be added to the electrostatic forces arising through the Hellmann Feynman theorem. They are usually difficult to calculate. Wavelets are not a basis set which depends on the the atomic position and therefore there are no Pulay forces. If an atom however moves a sufficiently large distance the high resolution region around the core has to be shifted by one grid spacing at some point. This will then lead to a discontinuity in both the total energy and forces. Cho, Arias and Joannopoulos [17] have however shown that this discontinuity is very small.

There are some publications in the literature trying to demonstrate the usefulness of wavelets for electronic structure calculations and we will briefly review them. The whole story began with a publication by Cho, Arias and Joannopoulos [17], where they pointed out that all the atomic 1s wavefunctions from H to U can be compactly represented with less than 200 Mexican hat basis functions. The Mexican hat function is a basis function which is similar to a wavelet. It satisfies some approximate refinement relation. Later on the same group (Arias, Cho, Joannopoulos, Lam and Teter) [12] also replaced the Mexican hat function by interpolating polynomials and did some electronic structure calculations in this basis, but they did not use operators in a nonstandard form. In a recent preprint Lippert, Arias and

Edelman [22] also recommend the use of the nonstandard operator form for the Laplace operator to handle inhomogeneous grid structures. There are two other recent papers on wavelets for electronic structure calculations. Wei and Chou [18] did self-consistent electronic structure calculations and Tymczak and Wang [19] also did some electronic structure calculations on simple test systems. In contrast to the papers mentioned above these two papers miss however the essential point of wavelet theory as it was presented in this article. They both used dynamic compression schemes and they did most of the basic operations on equal resolution grids (corresponding to the finest resolution level), whereas it is possible to do all the basic operations with the nonstandard operator scheme without ever resorting to an huge underlying fine grid. It is clear that such an approach does not lead to significant computational advantages, it can at best reduce the memory requirements.

29 Efficient implementation of filter operations

All the theoretical advantages of wavelet based methods are of course independent of the software implementation. In practice an efficient implementation can however be very important for the success of a method. Since this is a new area where no libraries are available, we will briefly discuss some basic principles which are important to get good execution speed on modern RISC computer architectures. All the basic loops are of the form

```
do i=0,n-1
do j=-m,m
y(i) = y(i) + c(j) * x(i+j)
enddo
enddo
```

First of all it is important that one uses the DDOT [25] paradigm instead of the DAXPY paradigm, i.e. that one uses the loop ordering shown above. If the filter is not very long one should also eliminate the innermost loop by unrolling it by hand. This will of course restrict one to a fixed filter length. Additional parallelism can also be exhibited to the compiler by working on several transform simultaneously. In the 3-dim case long inner loops can be obtained by transpositions of the data set [26].

Wavelet transform have a more local data access pattern than Fast Fourier transformations. By following the advice given above as well as by using more machine specific tuning one should therefore be able to get higher execution speed than for Fast Fourier routines. On IBM Power2 RISC architectures,

we were able to obtain between 50 and 75 percent of the peak performance in the different filtering parts. So wavelet based routines are not only mathematically very elegant and powerful, but you can get them also running very fast.

Exercise

- Optimize your wavelet transform routine.

30 Outlook and conclusions

Since we used mainly interpolating wavelets, all we did was essentially interpolating, which is one of the oldest technique in numerical analysis. However the framework provided by wavelet theory puts this whole interpolation business on the new and powerful basis of multi-resolution analysis, expanding thus considerably the scope of interpolation based techniques. In particular it assigns basis functions to certain interpolation schemes. Wavelet based techniques allow us for the first time to solve differential equations which have several length scales and to do this with linear scaling. It is thus to be expected, that wavelet based techniques will catalyze progress in many fields of science and engineering [15], where such problems exist. A lot of exploration has however still to be done to find the most efficient path among all the many different ways which are accessible within wavelet theory.

31 Appendix: Various filter coefficients

31.1 h filter coefficients for Daubechies family

Families with the maximum number of vanishing moments are denoted by "extremal", the most symmetric families by "symmetric".

i	degree 4	degree 6	degree 8 (extremal)
-4			.2303778133088964d0
-3		.3326705529500826d0	.7148465705529154d0
-2	.4829629131445341d0	.8068915093110926d0	.6308807679298587d0
-1	.8365163037378077d0	.4598775021184916d0	-.0279837694168599d0
0	.2241438680420134d0	-.1350110200102546d0	-.1870348117190931d0
1	-.1294095225512603d0	-.0854412738820267d0	.0308413818355607d0
2		.0352262918857095d0	.0328830116668852d0
3			-.0105974017850690d0

i	degree 8 (symmetric)	degree 10 (extremal)	degree 10 (extremal)
-5		.1601023979741929d0	.0273330683449988d0
-4	-.075765714789502d0	.6038292697971897d0	.0295194909257063d0
-3	-.029635527646002d0	.7243085284377729d0	-.0391342493023138d0
-2	.497618667632775d0	.1384281459013207d0	.1993975339768556d0
-1	.803738751805132d0	-.2422948870663820d0	.7234076904040408d0
0	.297857795605306d0	-.0322448695846384d0	.6339789634567921d0
1	-.099219543576634d0	.0775714938400457d0	.0166021057645108d0
2	-.012603967262031d0	-.0062414902127983d0	-.1753280899080562d0
3	.032223100604051d0	-.0125807519990820d0	-.0211018340246890d0
4		.0033357252854738d0	.0195388827352498d0

31.2 First derivative filter a for Daubechies family

The derivative filter for the extremal and symmetric version are identical. The filters for negative indices follow from $a_{-i} = -a_i$:

i	degree 6	degree 8	degree 10
0	0	0	0
1	272/365	39296/49553	957310976/1159104017
2	-53/365	-76113/396424	-265226398/1159104017
3	16/1095	1664/49553	735232/13780629
4	1/2920	-2645/1189272	-17297069/2318208034
5		-128/743295	1386496/5795520085
6		1/1189272	563818/10431936153
7			2048/8113728119
8			5/18545664272

31.3 Second derivative filter a for Daubechies family

The filters for negative indices follow from $a_{-i} = a_i$:

i	degree 6	degree 8	degree 10
0	-295/56	-342643/82248	-2370618501415/618154371936
1	356/105	2852128/1079505	1632655076608/676106344305
2	-92/105	-12053651/17272080	-439132551286/676106344305
3	4/35	162976/1079505	367031529728/2028319032915
4	3/560	-60871/5757360	-80883901277/2704425377220
5		-352/215901	107449600/135221268861
6		55/3454416	148937594/405663806583
7			32000/19317324123
8			4375/1236308743872

31.4 Filter coefficients for interpolating wavelets

For the ordinary interpolating wavelets we have: $\tilde{h}_i = \delta_i$. The h filters for negative indices follow by symmetry $h_{-i} = h_i$, the positive entries are given by:

i	degree 4	degree 6	degree 8	degree 10
0	1	1	1	1
1	9/16	75/128	1225/2048	19845/32768
2	0	0	0	0
3	-1/16	-25/256	-245/2048	-2205/16384
4		0	0	0
5		3/256	49/2048	567/16384
6			0	0
7			-5/2048	-405/65536
8				0
9				35/65536

The h filter coefficients for the lifted wavelets equal those of the unlifted ones. The $\tilde{h}$ filters for a wavelet which is lifted such that the zero and first moment vanishes are given by (where again $\tilde{h}_{-i} = \tilde{h}_i$):

i	degree 4	degree 6	degree 8	degree 10
0	23/32	181/256	2871/4096	45691/65536
1	1/4	1/4	1/4	1/4
2	-1/8	-125/1024	-245/2048	-15435/131072
3	0	0	0	0
4	1/64	11/512	49/2048	819/32768
5		0	0	0
6		-3/1024	-11/2048	-1863/262144
7			0	0
8			5/8192	185/131072
9				0
10				-35/262144

31.5 First derivative filter a for unlifted interpolating wavelets

The filters for negative indices follow from $a_{-i} = -a_i$:

i	degree 6	degree 8	degree 10
0	0	0	0
1	272/365	39296/49553	957310976/1159104017
2	-53/365	-76113/396424	-265226398/1159104017
3	16/1095	1664/49553	735232/13780629
4	1/2920	-2645/1189272	-17297069/2318208034
5		-128/743295	1386496/5795520085
6		1/1189272	563818/10431936153
7			2048/8113728119
8			5/18545664272

31.6 Second derivative filter a for unlifted interpolating wavelets

The filters for negative indices follow from $a_{-i} = a_i$:

i	degree 6	degree 8	degree 10
0	-295/56	-342643/82248	-2370618501415/618154371936
1	356/105	2852128/1079505	1632655076608/676106344305
2	-92/105	-12053651/17272080	-439132551286/676106344305
3	4/35	162976/1079505	367031529728/2028319032915
4	3/560	-60871/5757360	-80883901277/2704425377220
5		-352/215901	107449600/135221268861
6		55/3454416	148937594/405663806583
7			32000/19317324123
8			4375/1236308743872

32 Solution of selected exercises

!!
Exercise 1:
A function can either be written as a sum of N scaling functions and wavelets on level k

$$f(x) = \sum_{j=0}^{N-1} s_j^k \phi_j^k(x) + \sum_{j=0}^{N-1} d_j^k \psi_j^k(x) \tag{89}$$

or as a sum of 2N scaling functions at resolution level $k+1$.

$$f(x) = \sum_{I=0}^{2N-1} s_I^{k+1} \phi_I^{k+1}(x) \tag{90}$$

Using the refinement relations (22) and (23) in Equation (89) one gets

$$f(x) = \sum_{i=0}^{N-1} s_i^k \sum_{j=-m}^{m} h_j \, \phi_{2i+j}^{k+1}(x) + \sum_{i=0}^{N-1} d_i^k \sum_{j=-m}^{m} g_j \, \phi_{2i+j}^{k+1}(x) \tag{91}$$

Let us now split up the sum over j in Equation (91) into a sum over the even and another sum over the odd terms

$$f(x) = \sum_{j=-m/2}^{m/2} \left[\sum_{i=0}^{N-1} (h_{2j} s_i^k + g_{2j} d_i^k) \phi_{2i+2j}^{k+1}(x) + \right. \tag{92}$$
$$\left. \sum_{i=0}^{N-1} (h_{2j+1} s_i^k + g_{2j+1} d_i^k) \phi_{2i+2j+1}^{k+1}(x) \right]$$

Comparing even and odd terms in Equations (90) and (92) we obtain

$$s_{2I}^{k+1} = \sum_{j=-m/2}^{m/2} h_{2j} s_{I-j}^k + g_{2j} d_{I-j}^k \tag{93}$$

$$s_{2I+1}^{k+1} = \sum_{j=-m/2}^{m/2} h_{2j+1} s_{I-j}^k + g_{2j+1} d_{I-j}^k \tag{94}$$

!!

Exercise 2:
The backward formula (27) for **even** indices is given by

$$s_{2I}^{k+1} = \sum_j h_{2j} \, s_{I-j}^k + g_{2j} \, d_{I-j}^k \tag{95}$$

Plugging in the forward formula (26) one gets

$$s_{2I}^{k+1} = \sum_j \sum_l (h_{2j}\tilde{h}_l + g_{2j}\tilde{g}_l) s_{l+2I-2j}^{k+1} \tag{96}$$

Introducing the index $\nu = l - 2j$ one obtains

$$s_{2I}^{k+1} = \sum_\nu \sum_j (h_{2j}\tilde{h}_{\nu+2j} + g_{2j}\tilde{g}_{\nu+2j}) s_{2I+\nu}^{k+1} \tag{97}$$

Let us split up this sum into a sums over even and odd ν.

$$\begin{aligned} s_{2I}^{k+1} = \sum_\nu \sum_j (h_{2j}\tilde{h}_{2\nu+2j} &+ g_{2j}\tilde{g}_{2\nu+2j}) s_{2I+2\nu}^{k+1} + \\ \sum_\nu \sum_j (h_{2j}\tilde{h}_{2\nu+1+2j} &+ g_{2j}\tilde{g}_{2\nu+1+2j}) s_{2I+2\nu+1}^{k+1} \end{aligned} \tag{98}$$

The part involving the filter g in the even terms of Equation (98) can be transformed into a sum involving the filter h by using the symmetry relations (12) and (13).

$$\sum_j g_{2j}\tilde{g}_{2\nu+2j} = \sum_j \tilde{h}_{-2j+1} h_{-2j-2\nu+1} = \sum_j \tilde{h}_{2j+1} h_{2j-2\nu+1} = \sum_j \tilde{h}_{2j+2\nu+1} h_{2j+1}$$

The analogous transformations can be done for the part involving the filter g in the odd terms of Equation (98).

$$\sum_j g_{2j}\tilde{g}_{2\nu+1+2j} = -\sum_j \tilde{h}_{-2j+1} h_{-2j-2\nu} = -\sum_j \tilde{h}_{2j+1} h_{2j-2\nu} = -\sum_j \tilde{h}_{2j+2\nu+1} h_{2j}$$

Equation (98) then becomes

$$\begin{aligned} s_{2I}^{k+1} = \sum_\nu \sum_j (h_{2j}\tilde{h}_{2\nu+2j} &+ h_{2j+1}\tilde{h}_{2j+2\nu+1}) s_{2I+2\nu}^{k+1} + \\ \sum_\nu \sum_j (h_{2j}\tilde{h}_{2\nu+1+2j} &- h_{2j}\tilde{h}_{2j+2\nu+1}) s_{2I+2\nu+1}^{k+1} \end{aligned} \tag{99}$$

Using the orthogonality relations for the filter (8) we thus get the desired result

$$s_{2I}^{k+1} = \sum_\nu \sum_j h_j \tilde{h}_{2\nu+j} s_{2I+2\nu}^{k+1} = s_{2I}^{k+1} \tag{100}$$

The backward formula for **odd** indices is given by

$$s_{2I+1}^{k+1} = \sum_j h_{2j+1}\, s_{I-j}^k + g_{2j+1}\, d_{I-j}^k \tag{101}$$

Plugging in the forward formula one gets

$$s_{2I+1}^{k+1} = \sum_j \sum_l (h_{2j+1}\tilde{h}_l + g_{2j+1}\tilde{g}_l) s_{l+2I-2j}^{k+1} \tag{102}$$

Introducing the index $\nu = l - 2j$ one obtains

$$s_{2I+1}^{k+1} = \sum_\nu \sum_j (h_{2j+1}\tilde{h}_{\nu+2j} + g_{2j+1}\tilde{g}_{\nu+2j}) s_{2I+\nu}^{k+1} \tag{103}$$

Let us split up this sum into a sums over even and odd ν.

$$\begin{aligned} s_{2I+1}^{k+1} = \sum_\nu \sum_j (h_{2j+1}\tilde{h}_{2\nu+2j} \quad + \quad g_{2j+1}\tilde{g}_{2\nu+2j}) s_{2I+2\nu}^{k+1} + \\ \sum_\nu \sum_j (h_{2j+1}\tilde{h}_{2\nu+1+2j} \quad + \quad g_{2j+1}\tilde{g}_{2\nu+1+2j}) s_{2I+2\nu+1}^{k+1} \end{aligned} \tag{104}$$

The part involving the filter g in the even terms of Equation (98) can be transformed into a sum involving the filter h by using the symmetry relations (12) and (13).

$$\sum_j g_{2j+1}\tilde{g}_{2\nu+2j} = -\sum_j \tilde{h}_{-2j} h_{-2j-2\nu+1} = -\sum_j \tilde{h}_{2j} h_{2j-2\nu+1} = -\sum_j \tilde{h}_{2j+2\nu} h_{2j+1}$$

The analogous transformations can be done for the part involving the filter g in the odd terms of Equation (98).

$$\sum_j g_{2j+1}\tilde{g}_{2\nu+1+2j} = \sum_j \tilde{h}_{-2j} h_{-2j-2\nu} = \sum_j \tilde{h}_{2j} h_{2j-2\nu} = \sum_j \tilde{h}_{2j+2\nu} h_{2j}$$

Equation (98) then becomes

$$\begin{aligned} s_{2I+1}^{k+1} = \sum_\nu \sum_j (h_{2j+1}\tilde{h}_{2\nu+2j} \quad - \quad h_{2j+1}\tilde{h}_{2\nu+2j}) s_{2I+2\nu}^{k+1} + \\ \sum_\nu \sum_j (h_{2j+1}\tilde{h}_{2\nu+1+2j} \quad + \quad h_{2j}\tilde{h}_{2\nu+2j}) s_{2I+2\nu+1}^{k+1} \end{aligned} \tag{105}$$

Using the orthogonality relations for the filter (8) we thus get the desired result

$$s_{2I+1}^{k+1} = \sum_\nu \sum_j h_j \tilde{h}_{2\nu+j} s_{2I+2\nu+1}^{k+1} = s_{2I+1}^{k+1} \tag{106}$$

!!!

Exercise 3:

Let us first prove the orthogonality relations for the case $k = q$.

$$\begin{aligned}\int \tilde{\phi}(x)\phi(x-i)dx &= 2\sum_{\mu,\nu} \tilde{h}_\mu h_\nu \int \tilde{\phi}(2x-\mu)\phi(2x-2i-\nu)dx \\ &= \sum_{\mu,\nu} \tilde{h}_\mu h_\nu \delta_{\mu-\nu-2i} = \sum_\mu \tilde{h}_\mu h_{\mu-2i} = \delta_i\end{aligned} \tag{107}$$

Analogously we get

$$\int \tilde{\psi}(x)\psi(x-i)dx = \sum_\mu \tilde{g}_\mu g_{\mu-2i} = \delta_i \tag{108}$$

$$\int \tilde{\psi}(x)\phi(x-i)dx = \sum_\mu \tilde{g}_\mu h_{\mu-2i} = 0 \tag{109}$$

$$\int \tilde{\phi}(x)\psi(x-i)dx = \sum_\mu \tilde{h}_\mu g_{\mu-2i} = 0 \tag{110}$$

Let us now consider the case where $k \neq q$. One can always write a scaling function or wavelet at a coarse level q as a linear combinations of scaling functions at the fine level k. Because as shown above these integrals vanish integrals among different levels vanish as well, unless one has on integral involving a wavelet at a coarse level and a scaling function at a fine level.

!!!

Exercise 4:

$$1 = \int \phi(x)dx = \sum_\mu h_\mu \int \phi(2x-\mu)dx = \frac{1}{2}\sum_\mu h_\mu \tag{111}$$

!!
Exercise 5:

```
        subroutine fortrans(nd,nt,x,y)
c forward wavelet transform
c nd: length of data set
c nt length of data in data set to be transformed
c m filter length (m has to be even!)
c x input data, y output data
        implicit real*8 (a-h,o-z)
        dimension x(0:nd-1),y(0:nd-1)
c include the approriate filter coefficients
        include 'lazy4.inc'

        do 100,i=0,nt/2-1
        y(     i)=0.d0
        y(nt/2+i)=0.d0

        do 50,j=-m+1,m

c periodically wrap index if necessary
        ind=j+2*i
99      continue
        if (ind.lt.0) then
        ind=ind+nt
        goto 99
        endif
        if (ind.ge.nt) then
        ind=ind-nt
        goto 99
        endif

        y(     i)=y(     i)+cht(j)*x(ind)
        y(nt/2+i)=y(nt/2+i)+cgt(j)*x(ind)
50      continue

100     continue

        return
        end



        subroutine backtrans(nd,nt,x,y)
c backward wavelet transform
c nd: length of data set
c nt length of data in data set to be transformed
c m filter length (m has to be even!)
c x input data, y output data
```

```
        implicit real*8 (a-h,o-z)
        dimension x(0:nd-1),y(0:nd-1)
c include the approriate filter coefficients
        include 'lazy4.inc'

        do 100,i=0,nt/2-1
        y(2*i+0)=0.d0
        y(2*i+1)=0.d0

        do 50,j=-m/2,m/2-1

c periodically wrap index if necessary
        ind=i-j
99      continue
        if (ind.lt.0) then
        ind=ind+nt/2
        goto 99
        endif
        if (ind.ge.nt/2) then
        ind=ind-nt/2
        goto 99
        endif

        y(2*i+0)=y(2*i+0) + ch(2*j-0)*x(ind)+cg(2*j-0)*x(ind+nt/2)
        y(2*i+1)=y(2*i+1) + ch(2*j+1)*x(ind)+cg(2*j+1)*x(ind+nt/2)
50      continue

100     continue

        return
        end
```

File lazy4.inc

```
        parameter(m=4)
        dimension ch(-m:m),cg(-m:m),cht(-m:m),cgt(-m:m)

c******** coefficients for wavelet transform *********************
        do i=-m,m
        ch(i)=0.d0
        cht(i)=0.d0
        cg(i)=0.d0
        cgt(i)=0.d0
        enddo

c h coefficients
        ch(-3)=-1.d0/16.d0
        ch(-2)=0.d0
```

```
        ch(-1)=9.d0/16.d0
        ch( 0)=1.d0
        ch( 1)=9.d0/16.d0
        ch( 2)=0.d0
        ch( 3)=-1.d0/16.d0
c
        cht(0)=1.d0

c g coefficients from h coefficients
        do i=-m,m-1
        cg(i+1)=cht(-i)*(-1)**(i+1)
        cgt(i+1)=ch(-i)*(-1)**(i+1)
        enddo
```

!!!

Exercise 6:

From the Equation (26) for the forward transform it follows that (neglecting periodic wrap around) the matrix elements of a N × N forward transformation matrix are given by

$$\begin{aligned} \tilde{\mathbf{F}}_{i,j} &= \tilde{h}_{j-2i} \ , \ 0 \leq i \leq N/2-1 \\ &= \tilde{g}_{j-2i} \ , \ N/2 \leq i \leq N-1 \end{aligned}$$

Analogously the matrix elements of the backward matrix follow from (27)

$$\begin{aligned} \mathbf{B}_{i,j} &= h_{i-2j} \ , \ 0 \leq j \leq N/2-1 \\ &= g_{i-2j} \ , \ N/2 \leq j \leq N-1 \end{aligned}$$

So obviously $\tilde{\mathbf{B}}^T = \tilde{\mathbf{F}}$.

!!!

Exercise 7:

The plot is shown in Figure 2. It is self similar, i.e. it looks the same on any scale. A constant can exactly be represented.

!!!

Exercise 8:

The interpolation of an nonzero data set with finite support will add on both side nonzero layers of width $(m/2 - 1/2) * 2^k$. The total with of an infinite interpolation process starting with a delta at resolution $k = 0$ is therefore given by

$$2 \sum_{\nu=0}^{\infty} \left(\frac{1}{2}\right)^{\nu} \frac{m-1}{2} = 2(m-1) \qquad (112)$$

!!

Exercise 9:

The filter coefficients are given in the Appendix.

!!

Exercise 10:

The filter coefficients are given in the Appendix.

!!

Exercise 11:

Setting the scaling function expansion coefficients s_i^0 equal to i^l will give a function $f = x^l + a_1 x^{l-1} + ... + a_l$ according to the theorem of section 10. Equation (79) remains valid since

$$\int \phi(x) \frac{\partial^l}{\partial x^l}(x^l + a_1 x^{l-1} + ... + a_l) dx = \int \phi(x) l! dx = l! \qquad (113)$$

!!

Exercise 12:

In the case of interpolating wavelets the dual scaling function is a delta function. The value of the filter a_i is therefore the value of the derivative at the point $x = i$.

$$a_i = \int \tilde{\phi}(x) \frac{\partial^l}{\partial x^l} \phi(x-i) dx = \int \delta(x) \frac{\partial^l}{\partial x^l} \phi(x-i) dx = \left. \frac{\partial^l}{\partial x^l} \phi(x) \right|_{x=i} \qquad (114)$$

We know, that ϕ extends over the interval [-(m-1) ; (m-1)]. Outside this intervall the function and all its derivatives vanish. All continous derivatives therefore also vanish at $\pm(m-1)$. The first point with nonvanishing derivative is thus $\pm(m-2)$

!!

Exercise 13:

The scaling function is not modified by the lifing process: $\Phi(x) = \phi(x)$ Therfore

$$\int \tilde{\Phi}(x-i) \phi(x-j) dx = \int \tilde{\Phi}(x-i) \Phi(x-j) dx = \delta_{i-j}$$

by the orthogonality relations for the new lifted family.

References

[1] F. Gygi, *Europhys. Lett.* **19**, 617 (1992); Phys. Rev. B **48** 11692 (1993); *Phys. Rev.* B **51** 11190 (1995).

[2] Y. Meyer, *"Ondelettes et opérateurs"* Hermann, Paris, 1990.

[3] I. Daubechies, *"Ten Lectures on Wavelets"*, SIAM, Philadelphia (1992).

[4] W. Sweldens and R. Piessens, *"Wavelets sampling techniques"*, Proceedings of the Joint Statistical Meetings, San Fransisco, August 1993.

[5] P. Schröder and W. Sweldens, University of South Carolina preprint.

[6] W. L. Briggs, *"A Multigrid Tutorial"*, SIAM, Philadelphia, 1987.

[7] J. Barnes and P. Hut, *Nature* **32**, 446 (1986).

[8] L. Greengard, *Science* **265**, 909 (1994).

[9] M. C. Strain, G. E. Scuseria, M. J. Frisch, *Science* **271**, 51, (1996) ; M. Challacombe, E. Schwegler and J. Almlöf, *J. Chem. Phys* **104**, 4685, (1996).

[10] S. Goedecker, M. Teter, and J. Hutter, *Phys. Rev. B* **54**, 1703, (1996).

[11] W. H. Press, B. P. Flannery, S. A. Teukolsky and W. T. Vetterling, *"Numerical Recipes, The Art of Scientific Computing"* Cambridge University Press, Cambridge, England, 1986.

[12] "Wavelet–transform representation of the electronic structure of materials," T.A. Arias, K. Cho, Pui Lam, J.D. Joannopoulos, M.P. Teter, Second Mardi Gras Conference: Toward Teraflop Computing and New Grand Challenge Applications, Baton Rouge, Louisiana, *February 1994.*

[13] W. Sweldens, *Appl. Comput. Harmon. Anal.* **3**, 186 (1996).

[14] G. Deslauriers and S. Dubuc, *Constr. Approx.* **5**, 49 (1989).

[15] *"Wavelets and their applications"* M. B. Ruskai et al., Ed., Jones and Bartlett, Boston, 1992.

[16] T. Arias, to appear in Rev. of Mod. Phys..

[17] K. Cho, T. Arias, J. Joannopoulos and P. Lam, *Phys. Rev. Lett.* **71**, 1808 (1993).

[18] S. Wei and M. Y. Chou, *Phys. Rev. Lett.* **76**, 2650 (1996).

[19] C. Tymczak and X. Wang, *Phys. Rev. Lett.* **78**, 3654 (1997).

[20] G. Beylkin, *SIAM J. on Numerical Analysis* **6**, 1716 (1992).

[21] G. Beylkin, R. Coifman and V. Rokhlin, *Comm. Pure and Appl. Math.* **44**, 141 (1991).

[22] R. A. Lippert, T. Arias and A. Edelman, *J. Comp. Physics* **140**, 278 (1998).

[23] J. Hutter, H.P. Lüthi and M. Parrinello, *Comp. Mat. Sci.* **2** 244 (1994).

[24] S. Goedecker, O. Ivanov, *Sol. State Comm.* **105** 665 (1998).

[25] Lapack User's Guide, SIAM, Philadelphia, 1992.

[26] S. Goedecker, *Comp. Phys. Commun.* **76**, 294 (1993).