1994

The Recurrence Relation of B-Wavelets

Rumi Kumazawa '94
Illinois Wesleyan University

The Recurrence Relation of B-Wavelets

Rumi Kumazawa
Mathematics Department, IWU
Advisor: Dr. Tian-Xiao He

1. Introduction

The study of wavelet functions is a relatively new area in mathematics. It is gaining more popularity in areas where Fourier transforms were traditionally used because of particular properties wavelets possess that Fourier transforms do not have. Wavelets are functions which can be written as translations and dilations of a function, and can be used to expand functions in $L^2(R)$. They are especially useful in picture and sound analyses [1] because they can decompose and reconstruct data rapidly and efficiently - details in the signals are not lost and there is no need to use large amounts of data.

This localization property is one of the advantages of using wavelets over Fourier transforms [5] because the Fourier method gives a global approximation, rather than a local one. Therefore, any irregularity in a local part of a signal can distort the whole Fourier approximation while this will not be the case when the wavelet approach is used. Finally, rather than the bigger trigonometric waves using cosines and sines as in the Fourier case, wavelets are much smaller waves, and easier to focus in detail.

Wavelets were first developed during the early 1980's by a French geophysicist, Jean Morlet, to help analyze earthquake data. Seismic signals were described in terms of translations and dilations of a specific function, and this process was considered better than the Fourier method, although the reason for this was not clear at the time. Since then, wavelets have been studied more extensively, and in 1985, Yves Meyer of the Universite de Paris-Dauphine discovered the first orthogonal family of wavelets, and in 1988, Ingrid Daubechies of AT&T Bell Laboratories published a paper on compactly supported wavelets.

Our goal is to construct smooth wavelet functions. In constructing such wavelet functions, we need a smooth scaling function to begin with. B-spline functions are suitable as our scaling function because they are piecewise polynomials with compact supports and are relatively smooth. B-wavelet functions are just dilations and translations of these B-spline functions. In addtion, we can find a recurrence relation of the B-spline functions with different order. Hence B-wavelets of any order can be constructed successively from the lower order ones.

The following section is on the properties of wavelet systems in general. The section following the properties is on the Haar wavelet, which is a very simple wavelet function. An example illustrates how the wavelet coefficients can be calculated using matrices. Next, we discuss a particular scaling function, the B-spline function, and how these functions of any order can be constructed successively from the lower order ones with a recurrence relation.

Then we examine the recurrence relations of both the B-spline functions and B-wavelet functions. Using both these relations, I wrote a computer program which first calculates the coefficients for the B-spline functions, and using these coefficients, it calculates the corresponding B-wavelet coefficients. The B-wavelet coefficients can in turn be used to graph the B-wavelet functions, and as we will see later, the higher order B-wavelet functions are indeed smooth.

2. Properties of Wavelet Systems

Wavelet constructions are based on a process known as "multiresolution," discovered by Stephane G. Mallat and Yves Meyer in 1986. Any square integrable function,f, defined in R can be represented as a limit of a series of approximations, where each approximation is a smoother version of f.

In general, a scaling function $\phi(t)$ must be found, which satisfy the following:

(1) $\int_{-\infty}^{\infty} \phi(t)\,\phi(t-n)\,dt = 0$, $n \in Z$, $n \neq 0$

(2) $\phi(t) = \sum_n c_n\,\phi(2t-n)$, $\sum_n |c_n|^2 < \infty$

(3) Each square integrable function f can be represented by the series

$$\sum_n a_n\ \phi(2^m t-n),\ \sum_n |a_n|^2 < \infty$$

Then we can define spaces V_m as

$$V_m = \{ \text{square integrable } f \mid f(t) = \sum_n c_n\,\phi(2^m t-n)\}.$$

Therefore,

$$f(t) \in V_m \Leftrightarrow f(t/2) \in V_{m-1}\text{ ,and}$$

$$\cdots \subseteq V_{-1} \subseteq V_0 \subseteq V_1 \subseteq \cdots \subseteq V_m \subseteq \cdots \subseteq L^2$$

(or $L^2(R)$ = the closure of $\bigcup_m V_m$)

Wavelets are defined as $\psi(t) := \sum_n c_{1-n}\,(-1)^n\,\phi(2t-n)$, and

$$\psi_{mn}(t) = 2^{m/2}\,\psi(2^m t-n),\ m,n \in Z.$$

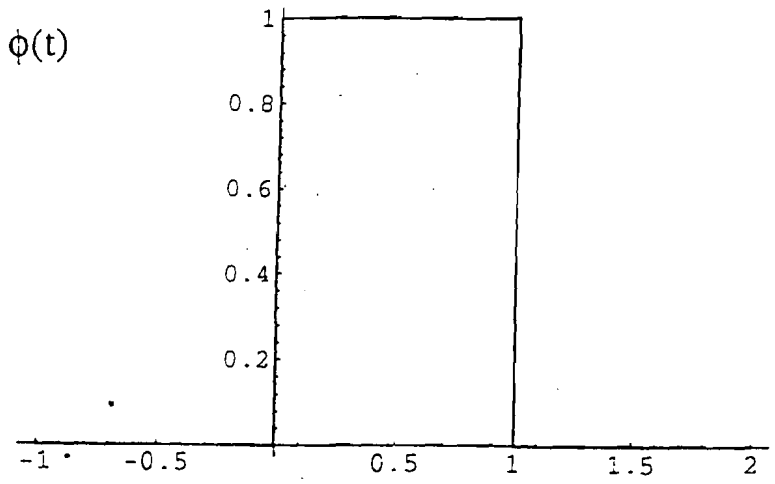
Every square integrable function can be written as a series

$$\sum_m \sum_n a_{mn}\,\psi_{mn}(t)$$

3. Example - The Haar Wavelet

The Haar wavelet system starts with the characteristic function ø(t) on the interval [0,1] where

$$ø(t) = \begin{cases} 1, & 0 \le t \le 1; \\ 0, & \text{otherwise.} \end{cases}$$



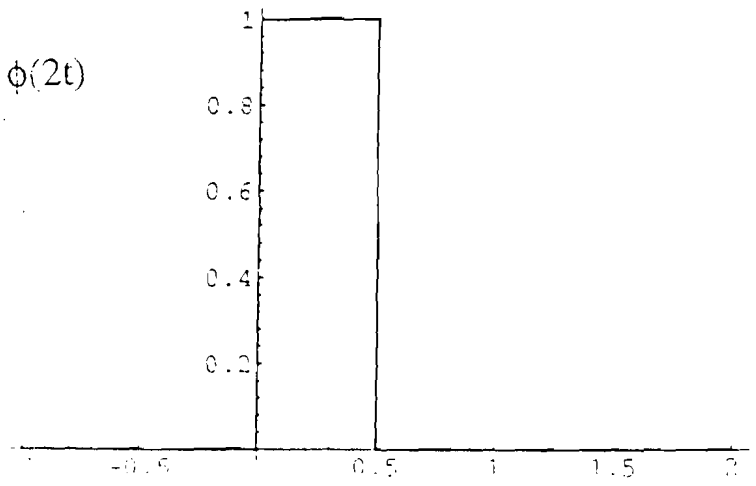
ø(t) satisfies the orthogonal property since the inner product of ø(t) and a translate of ø(t) is equal to zero, and hence

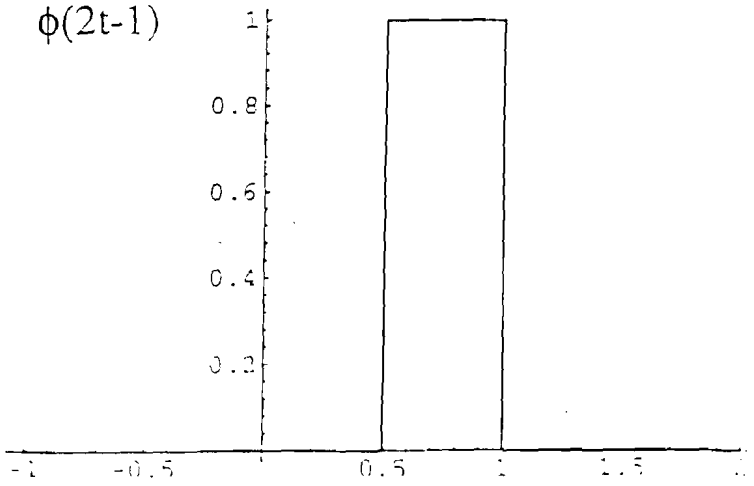
$$\int_{-\infty}^{\infty} ø(t)ø(t-n)\,dt = 0, \text{ for } n \neq 0.$$

If t is substituted by $2^m t$, then the equation still holds,

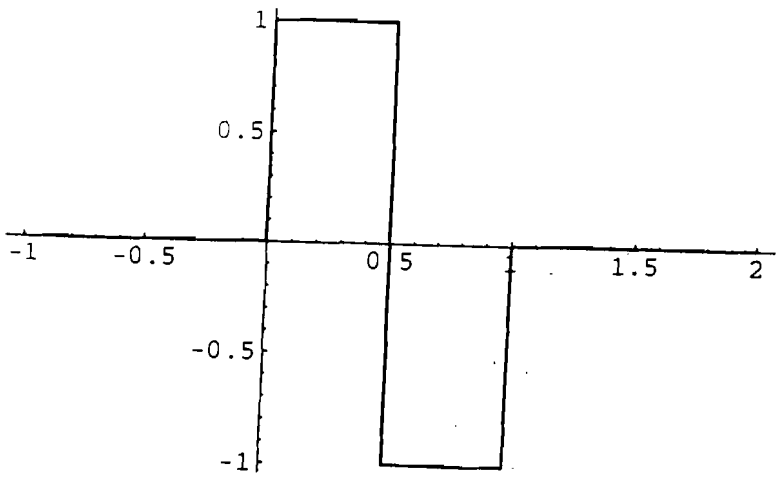
i.e. $\int_{-\infty}^{\infty} ø(2^m t)ø(2^m t-n)\,dt = 0, \; m \in Z.$

However, a problem arises if we try to show that ø(2t-k) and ø(t-n) are orthogonal by a similar argument. Therefore, ø(t) cannot be a so-called "mother wavelet." This problem can be overcome if we let ψ(t) be equal to ø(2t) - ø(2t-1).





$\psi(t) = \phi(2t) - \phi(2t-1)$



Now we can show that $\psi(t-k)$ and $\psi(2t-n)$ are orthogonal.
First of all, $\psi(t)$ and $\psi(t-n)$ are orthogonal since

$$\int_{-\infty}^{\infty} \psi(t)\ \psi(t-n)\ dt = 0 \text{ for } n \neq 0.$$

Next, $\psi(t-n)$ and $\phi(t-k)$ are orthogonal since

$$\int_{-\infty}^{\infty} \psi(t-n)\ \phi(t-k)\ dt = 0.$$

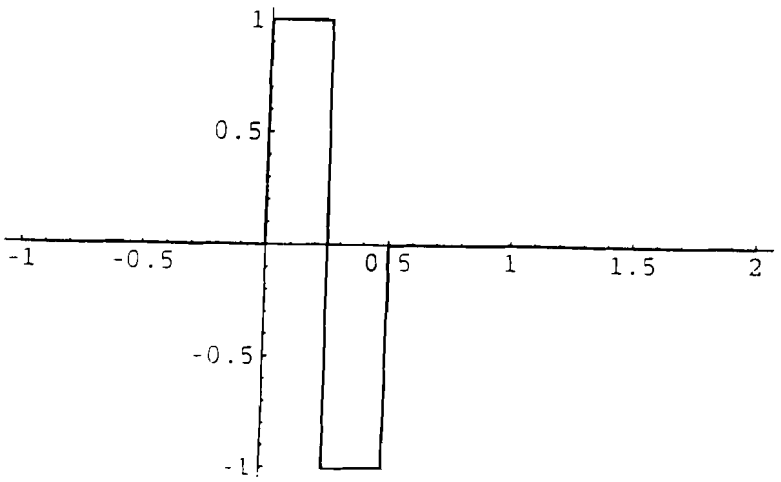
Substituting 2t for t, we can show that $\psi(2t-n)$ and $\phi(2t-k)$ are also orthogonal.
Then it follows that $\psi(2t-k)$ and $\psi(2t-n)$ must be orthogonal because

$$\int_{-\infty}^{\infty} \psi(t-k)\ \psi(2t-n)\ dt$$

$$= \int_{-\infty}^{\infty} [\phi(2t-k) - \phi(2t-2k-1)]\ \psi(2t-n)\ dt = 0.$$

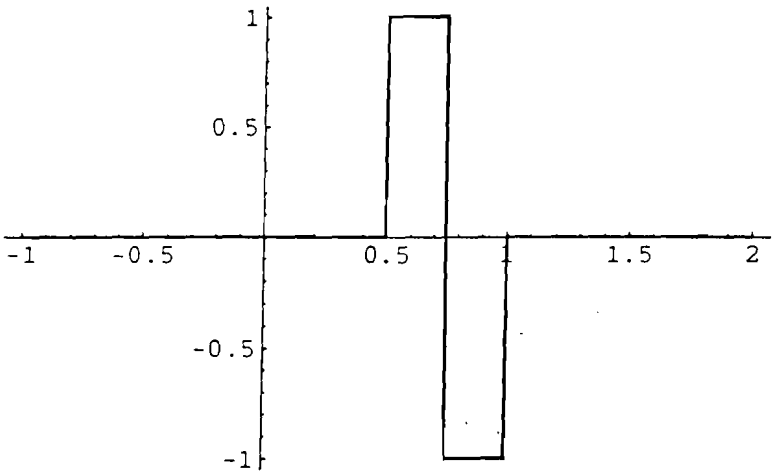
The graphs of $\psi(2t)$ and $\psi(2t-1)$ are shown as the following :

$\psi(2t)$



Note that $\psi(2t)$ is the dilation of $\psi(t)$ by one half.

ψ(2t-1)



Any compactly supported function f(x) defined on [0,1] that are constant on 4 quarter subintervals of [0,1] can be represented as a linear combination of ø(t), ψ(t), ψ(2t), and ψ(2t-1).

For example, suppose the values of f(x) on four quarter subintervals are 9,1,2,0. Then it can be written as the 4x1 matrix $\begin{bmatrix} 9 \\ 1 \\ 2 \\ 0 \end{bmatrix}$.

ø(t) can be written as the matrix $\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$

since it is defined as 1 on the four quarter intervals [0, 1/4], [1/4, 1/2], [1/2, 3/4], and [3/4, 1]. By a similar method, the matrices for ψ(t), ψ(2t), and ψ(2t-1) are found to be

$\begin{bmatrix} 1 \\ 1 \\ -1 \\ -1 \end{bmatrix}, \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0 \end{bmatrix}$, and $\begin{bmatrix} 0 \\ 0 \\ 1 \\ -1 \end{bmatrix}$ respectively.

So

$$\begin{bmatrix} 9 \\ 1 \\ 2 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} a_1 + \begin{bmatrix} 1 \\ 1 \\ -1 \\ -1 \end{bmatrix} a_2 + \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0 \end{bmatrix} a_3 + \begin{bmatrix} 0 \\ 0 \\ 1 \\ -1 \end{bmatrix} a_4$$

and a_1, a_2, a_3, and a_4 are called the wavelet coefficients.

They can be solved by solving simultaneously from

$$\begin{bmatrix} 9 \\ 1 \\ 2 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & -1 & 0 \\ 1 & -1 & 0 & 1 \\ 1 & -1 & 0 & -1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a3 \\ a_4 \end{bmatrix}$$ to get $a_1 = 3$, $a_2 = 2$, $a_3 = 4$, and $a_4 = 1$.

The matrix $\begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & -1 & 0 \\ 1 & -1 & 0 & 1 \\ 1 & -1 & 0 & -1 \end{bmatrix}$ is called the wavelet matrix and has some zero entries.

Using wavelets over the Fourier method is faster because the wavelet matrices tend to be sparse, or have many zero entries, while Fourier matrices are full. In addition, wavelet coefficients come from values at n subintervals rather than from n points as in the case of the Fourier method, so calculations are much easier [4].

One major drawback in approximations using the Haar wavelet system is that it cannot be used to represent a smooth continuous function. Because of the rigid boundaries, approximations at these points would be very poor. Therefore, we need higher order piecewise polynomials in order to approximate smoother functions.

4. B-Spline Functions

As mentioned above, a desirable property for the scaling function is some degree of smoothness so that wavelet functions of higher order will be much smoother than the Haar wavelets.

A B-spline function of order m, $B_m(x)$, which has support [-m/2, m/2] and is symmetric about the y-axis, could be a suitable scaling function. We can define it on the interval [0,m] by shifting the function m/2 units to the right,

i.e. let $N_m(x) = B_m(x - m/2)$.

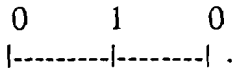
$N_1(x)$ is defined to be $\begin{cases} 1/2, & x=0, \\ 1, & 0<x<1, \\ 1/2, & x=1. \end{cases}$

$N_2(x)$ is the piecewise polynomial function $\begin{cases} x, & 0 \le x \le 1, \\ 2-x, & 1 \le x \le 2. \end{cases}$

On each interval [i,i+1], $B_m(x)$ can be written as

$$\Sigma\, a_{ij}\, m!/i!j!\, u^i v^j,\quad u = (i+1)-x,\quad v = x-i,$$

where aij are called Bernstein-Bezier coefficients, or B-B coefficients. For example, the B-B coefficients of $N_2(x)$ on [0,2] are



Higher order B-spline functions can be constructed from lower order ones, by first calculating the Bernstein-Bezier coefficients.
The coefficients of $N_{m+1}(x)$ can be calculated by first looking at

$$1/m[N_m(x) - N_m(x-1)].$$

To find the coefficients of N_3 on [0,3], we can do the following

$N_2(x)$ 0 1 0

$N_2(x-1)$ 0 1 0

$N_2(x) - N_2(x-1)$ 0 1 -1 0

$1/2[N_2(x)-N_2(x-1)]$ 0 1/2 -1/2 0

0 0 1/2 1 1/2 0 0

The coefficients of $N_3(x)$ on [0,3] are

0 0 1/2 1 1/2 0 0

Similarly, we can find the coefficients of $N_4(x)$ on [0,4] :

$N_3(x)$ 0 0 1/2 1 1/2 0 0

$N_3(x-1)$ 0 0 1/2 1 1/2 0 0

$N_3(x)-N_3(x-1)$ 0 0 1/2 1 0 -1 1/2 0 0

$1/3[N_3(x)-N_3(x-1)]$ 0 0 1/6 1/3 0 -1/3 -1/6 0 0

0 0 0 1/6 1/3 2/3 2/3 2/3 1/3 1/6 0 0 0

The coefficients of $N_4(x)$ on [0,4] are 0 0 0 1/6 1/3 2/3 2/3 2/3 1/3 1/6 0 0 0

Now, the actual B-spline functions can be constructed on any unit subinterval [k-1, k] by the following formula [2].

$$N_m(x)\,|_{[k-1,k)} = \sum a_l^{m-1}\, \phi_l^{m-1}(u,v)$$

where $\phi_l^{m-1}(u,v) = (m-1)!\,/\,i!j!\;u^i v^j, \quad i+j=m-1$

and $u = (x_1-x)/(x_1-x_0) = (k-x)/[k-(k-1)]$,

$v = (x-x_0)/(x_1-x_0) = [x-(k-1)]/[k-(k-1)]$.

a_l^{m-1} are the Bernstein-Bezier coefficients.

$$N_2(x)|_{[0,1)} = \sum a_l^1\, \phi_l^1(u,v)$$
$$= a_0^1\, \phi_0^1(u,v) + a_1^1\, \phi_1^1(u,v)$$
$$= (0)\, \phi_0^1(u,v) + (1)\, \phi_1^1(u,v)$$
$$= (1)\, v$$
$$= (x-0)/(1-0)$$
$$= x.$$

0 1 0

|-------|-------|

a_0^1 a_1^1

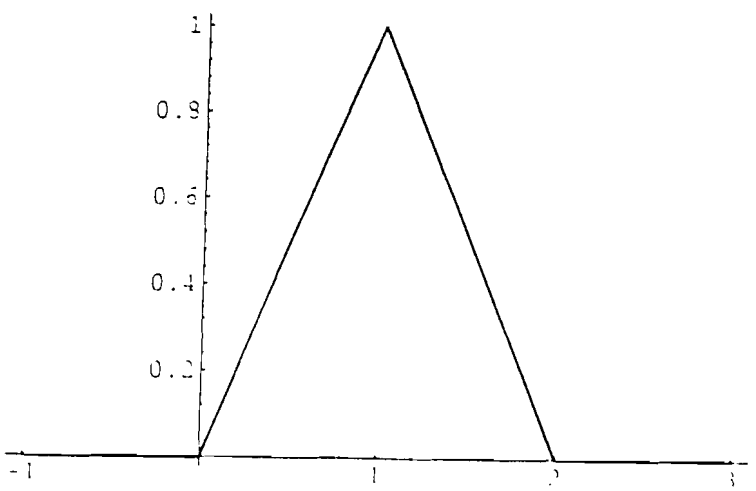
$$N_2(x)|_{[1,2)} = \sum a_l^1\, \phi_l^1(u,v)$$
$$= a_0^1\, \phi_0^1(u,v) + a_1^1\, \phi_1^1(u,v)$$
$$= (1)\, \phi_0^1(u,v) + (0)\, \phi_1^1(u,v)$$
$$= (2-x)/(2-1)$$
$$= 2-x.$$

0 1 0

|-------|-------|

a_0^1 a_1^1

Therefore, as we have shown earlier, $N_2(x) = \begin{cases} x, & 0 \le x < 1, \\ 2-x, & 1 \le x < 2. \end{cases}$



Similarly, $N_3(x)$ can be found on each unit interval of [0,1].

$N_3(x)|_{[0,1)} = \sum a_l^2 \, \phi_l^2(u,v)$

$= a_0^2 \, \phi_0^2(u,v) + a_1^2 \, \phi_1^2(u,v) + a_2^2 \, \phi_2^2(u,v)$

$= (0) \, \phi_0^2(u,v) + (0) \, \phi_1^2(u,v) + 1/2 \;\; \phi_2^2(u,v)$

$= 0 + 0 + 1/2 \; (3-1)!/2!0! \; u^0 v^2$

$= 1/2 \; v^2$

$= 1/2 \;\; [(x-0)/(1-0)]^2$

$= 1/2 \; x^2.$

0	0	1/2
a_0^2	a_1^2	a_2^2

$N_3(x)|_{[1,2)} = \sum a_l^2 \, \phi_l^2(u,v)$

$= a_0^2 \, \phi_0^2(u,v) + a_1^2 \, \phi_1^2(u,v) + a_2^2 \, \phi_2^2(u,v)$

$= 1/2 \; (2-x)^2 \;+\; 2(2-x)(x-1) \;+\; 1/2 \;\; (x-1)^2$

$= 1/2 \; (4-4x+x^2) + (4-2x)(x-1)$

$+ 1/2 \; (x^2-2x+1)$

$= 2-2x+1/2x^2+4x-4-2x^2+2x+1/2x^2-x+1/2$

$= -x^2+3x-3/2.$

1/2	1	1/2
a_0^2	a_1^2	a_2^2

$N_3(x)|_{[2,3)} = \sum a_l^2 \, \phi_l^2(u,v)$

$= a_0^2 \, \phi_0^2(u,v) + a_1^2 \, \phi_1^2(u,v) + a_2^2 \, \phi_2^2(u,v)$

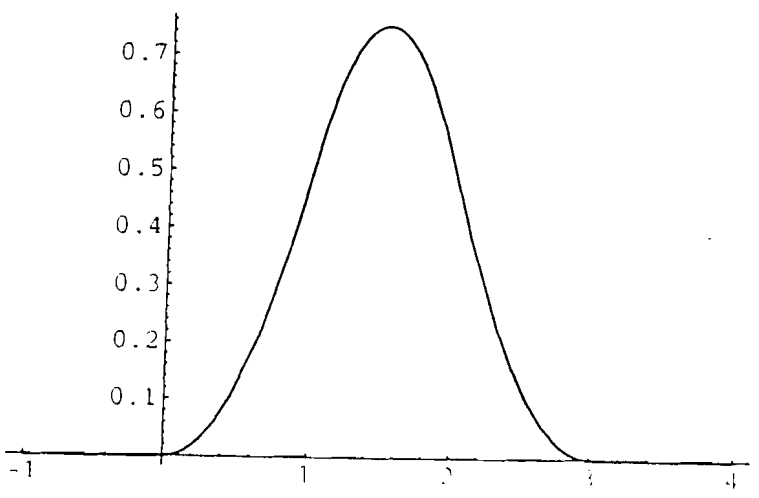
$= 1/2 \; u^2 \;+\; (0)(2uv) \;+\; (0)v^2$

$= 1/2 \; (3-x)^2.$

1/2	0	0
a_0^2	a_1^2	a_2^2

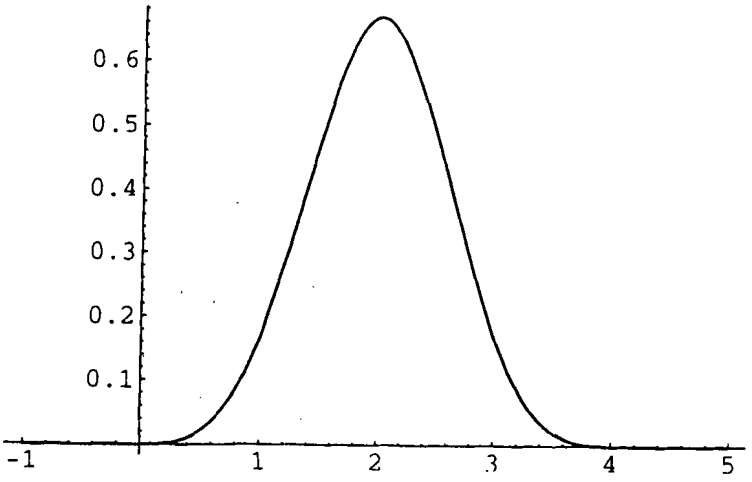
Graphing the piecewise defined polynomial functions on each interval, we get the graph of $N_3(x)$, which is a smoother version of $N_2(x)$, whose support is longer by one unit.

$N_3(x)$

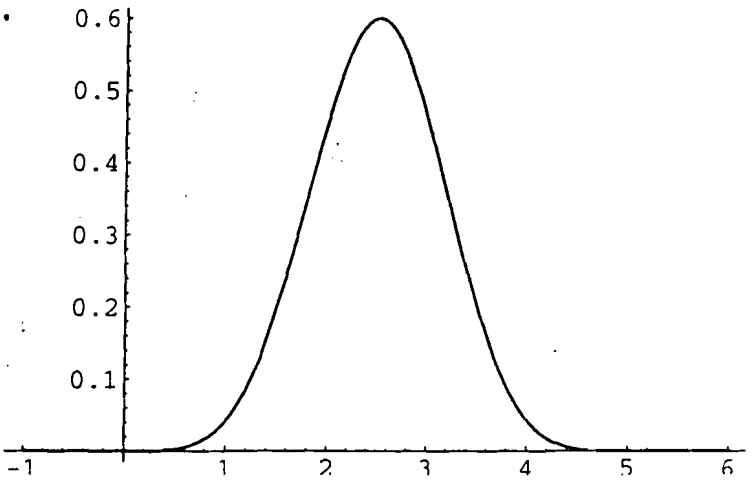


The following are higher order B-spline functions, graphed by using *Mathematica*.

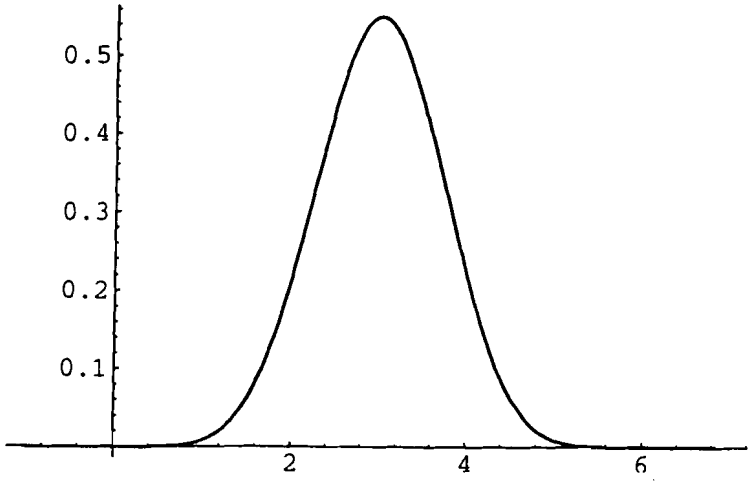
$N_4(x)$



$N_5(x)$



$N_6(x)$



5. The Recurrence Relation of B-Splines

It was shown earlier that the coefficients of a B-spline function of order m can be constructed from those of order m-1. This means that the coefficients can be found recursively, and hence a recurrence relation exists between B-splines.

Polynomials with B-B coefficients can be written as

$P_m(x) = \sum_{k=0}^{m} a_k^m \phi_k^m(x)$, $x \epsilon [0,1]$

a_k^m = B-B coefficient

$\phi_k^m = m! / (k!\,(m-k)!)\; u^{m-k} v^k$, $u=1-x$, $v=x$

$= m! / (k!(m-k)!)\; (1-x)^{m-k} x^k$

Take the derivative of $P_m(x)$

$$P'_m(x) = d/dx \sum_{k=0}^{m} a_k^m \phi_k^m(x)$$

$$= \sum_{k=0}^{m} a_k^m \, (m! / k!(m-k)!) \, [-(m-k)(1-x)^{m-k-1} x^k + k(1-x)^{m-k} x^{k-1}]$$

$$= \sum_{k=0}^{m} a_k^m \, \{-m \, [(m-1)! / k!\,(m-1-k)!] \, (1-x)^{m-1-k} x^k +$$

$$m[(m-1)! / (k-1)!(m-k)!] \, (1-x)^{m-k} x^{k-1} \}$$

Define $\phi_m^{m-1}(x) = 0$ if $k=m$ and define $\phi_{-1}^{m-1}(x) = 0$ if $k=0$

$$= \sum_{k=0}^{m-1} (-m) a_k^m \phi_k^{m-1}(x) + \sum_{k=1}^{m} (m) a_k^m \phi_{k-1}^{m-1}(x)$$

Let $k' = k-1$. Then if $k=1$, $k'=0$. If $k=m$, $k'=m-1$.

$$= m \sum_{k=0}^{m-1} (-a_k^m) \phi_k^{m-1}(x) + m \sum_{k'=0}^{m-1} a_{k'+1}^m \phi_{k'}^{m-1}(x)$$

Substituting k' back to k,

$$= m \sum_{k=0}^{m-1} (-a_k^m) \phi_k^{m-1}(x) + m \sum_{k=0}^{m-1} a_{k+1}^m \phi_k^{m-1}(x)$$

$$= m \sum_{k=0}^{m-1} (a_{k+1}^m - a_k^m) \phi_k^{m-1}(x)$$

If $P'_{m+1}(x) = P_m(x)$ then $\int_0^{\infty} P_m(t)\, dt = P_{m+1}(t) \,|_0^x = P_{m+1}(x) - P_{m+1}(0)$

$P_{m+1}(0) = a_0^{m+1}$

$$P'_{m+1}(x) = (m+1) \sum (a_{k+1}^{m+1} - a_k^{m+1}) \phi_k^m(x)$$

$$= P_m(x) = \sum_{k=0}^{m} a_k^m \phi_k^m(x)$$

$a_j^m = (m+1)(a_{j+1}^{m+1} - a_j^{m+1}) \Leftrightarrow \sum_{j=0}^{k-1} a_j^m = (m+1)(a_k^{m+1} - a_0^{m+1})$

$\Leftrightarrow 1/(m+1) \sum_{j=0}^{k-1} a_j^m + a_0^{m+1} = a_k^{m+1}$

For each 0,...,m there are m+1 coefficients (in each unit interval).

The recurrence relation of the B-splines is

$$a_{l+1}^m(k) = a_l^m(k) + 1/(m-1)\,[a_l^{m-1}(k) - a_l^{m-1}(k-1)]$$

6. The Recurrence Relation of B-Wavelets

Since a recurrence relation exists between B-splines of order m and $m+1$, the recurrence relation can be extended to that of the B-wavelets - which are linear combinations of these B-splines with the appropriate wavelet coefficients.

1. Any square integrable function f can be written as a linear combination of ψ,

$\Sigma \ \ldots + a_j^{-1}\psi(1/2\ x-j) + a_j^0\ \psi(x-j) + a_j^1\ \psi(2x-j) + \ldots$

where $f \in L^2$, i.e., $\int_{-\infty}^{\infty} |f(x)^2| < \infty$.

2. A scaling function satisfies the following (cf. section 2) :

(i) L^2 = the closure of $\bigcup_i V_i$.

(ii) If $\{\phi(x-k)\}$ is a basis of V_0, then $\{\phi(2^j x-k)\}$ is a basis of V_j.

(iii)

$$\begin{array}{ccccccc} \ldots & W_0 & & W_1 & & W_2 & \ldots \\ & \oplus & & \oplus & & \oplus & \\ \ldots & V_0 & & V_1 & & V_2 & \ldots \end{array}$$

$V_1 = V_0 \oplus W_0$

$V_2 = V_1 \oplus W_1 = V_0 \oplus W_0 \oplus W_1$

$V_3 = V_2 \oplus W_2 = V_1 \oplus W_1 \oplus W_2 = V_0 \oplus W_0 \oplus W_1 \oplus W_2$

.

.

.

From the scaling function ϕ (which in the case of B-splines is $N_m(x)$), ψ can be found by the expression

$$\psi_m(x) = \sum_l q_l^m\ N_m(2x-l),\quad 0 \le l \le 3m-2,$$

where q_l^m are the wavelet coefficients and they are defined as

$$q_l^m = (-1)^l/2^{m-1} \sum m!/j!(m-j)!\ N_{2m}\ (l-j+1).$$

(cf. [2])

Construction of $\psi_2(x)$. Evaluate

$$q_0^2 = (-1)^0/2^1 \sum_{j=0}^{2} 2!/j!(2-j)!\, N_4(0-j+1) = 1/2\,[N_4(1) + 2N_4(0) + N_4(-1)]$$
$$= 1/12 = 0.083333\,,$$
$$q_1^2 = (-1)^1/2^1 \sum_{j=0}^{2} 2!/j!(2-j)!\, N_4(2-j) = -1/2\,[N_4(2) + 2N_4(1) + N_4(0)]$$
$$= -1/2 = -0.5\,,$$
$$q_2^2 = (-1)^2/2^1 \sum_{j=0}^{2} 2!/j!(2-j)!\, N_4(3-j) = 1/2\,[N_4(3) + 2N_4(2) + N_4(1)]$$
$$= 1/2[1/6 + 2(2/3) + 1/6] = 5/6 = 0.8333333\,,$$
$$q_3^2 = (-1)^3/2^1 \sum_{j=0}^{2} 2!/j!(2-j)!\, N_4(4-j) = -1/2\,[N_4(4) + 2N_4(3) + N_4(2)]$$
$$= -1/2[0 + 2(1/6) + 2/3] = -1/2 = -0.5\,,$$
$$q_4^2 = (-1)^4/2^1 \sum_{j=0}^{2} 2!/j!(2-j)!\, N_4(5-j) = -1/2\,[N_4(5) + 2N_4(4) + N_4(3)]$$
$$= 1/2[0 + 0 + 1/6] = 1/12 = 0.083333\,.$$

Thus

$$\psi_2(x) = \sum q_l^n\, N_n(2x-l) = \sum q_l^2\, N_2(2x-l)$$
$$= 0.083333\, N_2(2x) - 0.5N_2(2x-1) + 0.833333N_2(2x-2) -$$
$$0.5N_2(2x-3) + 0.083333N_2(2x-4)\,.$$

Construction of $\psi_3(x)$. Evaluate

$$q_0^3 = (-1)^0/2^2 \sum_{j=0}^{3} 3!/j!(3-j)!\, N_6(1-j)$$

$$= (-1)^0/2^2 [\, N_6(1) + 3N_6(0) + 3N_6(-1) + N_6(-2)$$

$$= 1/4\,[0.0083 + 0 + 0 + 0] = 0.002083 = 1/480\,,$$

$$q_1^3 = (-1)^1/2^2 \sum_{j=0}^{3} 3!/j!(3-j)!\, N_6(2-j)$$

$$= (-1)^1/2^2 [\, N_6(2) + 3N_6(1) + 3N_6(0) + N_6(-1)$$

$$= -1/4\,[0.2167 + 3(0.0083) + 0 + 0] = -0.0604167 = -29/480\,,$$

$$q_2^3 = (-1)^2/2^2 \sum_{j=0}^{3} 3!/j!(3-j)!\, N_6(3-j)$$

$$= (-1)^2/2^2 [\, N_6(3) + 3N_6(2) + 3N_6(1) + N_6(0)$$

$$= 1/4\,[0.55 + 3(0.2167) + 3(0.0083) + 0] = 0.30625 = 147/480\,,$$

$$q_3^3 = (-1)^3/2^2 \sum_{j=0}^{3} 3!/j!(3-j)!\, N_6(4-j)$$

$$= (-1)^3/2^2 [\, N_6(4) + 3N_6(3) + 3N_6(2) + N_6(1)$$

$$= -1/4\,[0.2167 + 3(0.55) + 3(0.2167) + 0.0083] = -0.63125 = -303/480\,,$$

$$q_4^3 = (-1)^4/2^2 \sum_{j=0}^{3} 3!/j!(3-j)!\, N_6(5-j)$$

$$= (-1)^4/2^2 [\, N_6(5) + 3N_6(4) + 3N_6(3) + N_6(2)$$

$$= 1/4\,[0.0083 + 3(0.2167) + 3(0.55) + 0.2167] = 0.63125 = 303/480\,,$$

$$q_5^3 = (-1)^5/2^2 \sum_{j=0}^{3} 3!/j!(3-j)!\, N_6(6-j)$$

$$= (-1)^5/2^2 [\, N_6(6) + 3N_6(5) + 3N_6(4) + N_6(3)$$

$$= -1/4\,[0 + 3(0.0083) + 3(0.2167) + 0.55] = -0.30625 = -147/480\,,$$

$$q_6^3 = (-1)^6/2^2 \sum_{j=0}^{3} 3!/j!(3-j)!\, N_6(7-j)$$

$$= (-1)^6/2^2 [\, N_6(7) + 3N_6(6) + 3N_6(5) + N_6(4)$$

$$= 1/4\,[0 + 0 + 3(0.0083) + 0.2167] = 0.0604167 = 29/480\,,$$

$$q_7^3 = (-1)^7/2^2 \sum_{j=0}^{3} 3!/j!(3-j)!\, N_6(8-j)$$

$$= (-1)^7/2^2 [\, N_6(8) + 3N_6(7) + 3N_6(6) + N_6(5)$$

$$= 1/4\,[0 + 0 + 0 + 0.0083] = 0.0083 = -1/480\,.$$

Thus

$$\psi_3(x) = \sum q_l^3\, N_3(2x-l)$$

$$= 0.002083\, N_3(2x) - 0.0604167\, N_3(2x-1) + 0.30625\, N_3(2x-2)$$
$$-0.63125\, N_3(2x-3) + 0.63125\, N_3(2x-4) - 0.30625\, N_3(2x-5)$$
$$+0.0604167\, N_3(2x-6) - 0.002083\, N_3(2x-7)\,.$$

A computer program was written in Turbo Pascal, which calculates the B-wavelet coefficients of higher order from the ones of lower order.

The user is prompted to enter the order (m) of the B-wavelet function. First, B-B coefficients of the B-spline of order 2m are calculated for each of the unit subintervals of [0, 2m]. Then the wavelet coefficients are determined and listed in the output.

The actual code for the program is as follows :

```
Program Wavelet_Coefficients (Input,Output);
{ This program calculates the wavelet coefficients by using B-B
  coefficients of B-splines of up to order 8.}

Uses crt;
Const Max=20;
      Range=15;
Type Degree = 1..8;
     Interval = -5..10;
     Coefficient_Num = 0..8 ;
     Coefficient_Array = Array[Degree,Coefficient_Num,Interval] of real;
     Pascal_Triangle = Array[0..Range] of real;
     Wavelet_Array = Array[0..Max,0..Max] of real;

Var  j,k,l,Poly_Degree, Wave_Degree : integer;
     Coefficient : Coefficient_Array;
     Wavelet_Coefficient : Wavelet_Array;
     Pascal,Sum : Pascal_Triangle;
     Odd : Boolean;
     Power,Wavelet_Factor,Negative:real;

{*****************************************************************************}
Procedure Second_Order_Polynomial(Var C : Coefficient_Array;
                                      D : Integer);
 This procedure sets up the initial conditions. Included are the
 coefficients of the B-spline function, of order 2. }
Var S,T : Integer;
Begin
 C[1,0,1] := 0;
 C[1,0,2] := 1;
 C[1,1,1] := 1;
 C[1,1,2] := 0;
 For S := 0 to (D-1) do
  begin
    For T:=-5 to 0 do
     C[D-1,S,T] := 0;
     C[D-1,S,D+1] := 0;
  end;
End;
{*****************************************************************************}
```

```
Procedure N_Order_Polynomial(Var C : Coefficient_Array ;
                             Var P : integer);
{ This procedure calculates B-B coefficients of polynomials of order greater
  than 2. It also sets everything outside of the domain to zero. }
Var  Q, R, S  : Integer;
Begin

  For Q := 0 to (P-1) do
   begin
    For S:=-5 to 0 do
     C[P-1,Q,S] := 0;
     C[P-1,Q,P+1] :=0;
     C[P-1,Q,P+2] :=0;
     C[P-1,Q,P+3] :=0;
   end;

   For R := 1 to P do
   For Q := 0 to (P-1) do
       If Q=0 then C[P-1,Q,R] := C[P-1,P-1,R-1]
        Else
         C[P-1,Q,R] := C[P-1,Q-1,R] + (C[P-2,Q-1,R] - C[P-2,Q-1,R-1]) /(P-1);
End;

{*************************************************************************}
Function N_Fact(n:integer):integer;
{ This function calculates the factorial of a number.}
 begin
   If (n=0) or (n=1) then N_Fact := 1
   Else N_Fact := n * N_Fact(n-1);
 end;

Function Combin(n,m:integer):real;
{ This function calculates the combination "n choose m."}
 begin
   If (m=n) or (m=0) then Combin:=1.0
   Else Combin:= N_Fact(n)/N_Fact(m)/N_Fact(n-m);
 end;

Function Exponential(Base, Exponent : integer) : real;
{ This function calculates "base" raised to the power "exponent."}
 begin
   Exponential := exp(Exponent * ln(Base));
 end;

Function Reciprocal(Number : real) : real;
{ This function calculates the reciprocal of a number. }
 begin
   Reciprocal := 1.0/Number;
 end;
{*************************************************************************}
```

```
Begin {Main}
  clrscr;
  Odd := False;
  Poly_Degree := 2;
  Second_Order_Polynomial(Coefficient,Poly_Degree);
  Writeln('Enter B-wavelet degree. [1..4]');
  Readln(Wave_Degree);
  If (Wave_Degree mod 2 = 1) then Odd := true
    Else Odd := false;

  {Repeats procedure until the desired degree of the B-pline is obtained,
   which is 2 * Wave_Degree. }

     while Wave_Degree <> (Poly_Degree/2) do
       begin Poly_Degree := Poly_Degree + 1;
             N_Order_Polynomial(Coefficient,Poly_Degree);
       end; {while}

  {This part writes down the B-B coefficients for each interval. }

  Writeln('Coefficients of B-spline with order ',Poly_Degree,' : ');
  For k := 1 to Poly_Degree do
    begin
       Write('Interval ',k,' :  ');
       For j := 0 to (Poly_Degree - 1) do
         Write(Coefficient[Poly_Degree-1,j,k]:2:4,'  ');
       Writeln;
    end; {for-loop}
 Writeln;
```

```
{This part calculates the wavelet coefficients.}

Writeln('Coefficients of B-wavelet with order ',Wave_Degree,' :');
For l:= 0 to (Wave_Degree) do
  begin
    Pascal[l] := Combin(Wave_Degree,l);
  end;
Power := Exponential(2,Wave_Degree-1);
Wavelet_Factor := Reciprocal(Power);
Odd:= True;
Wavelet_Factor := Wavelet_factor * -1;

For l := 0 to (3*Wave_Degree-2) do
  For j:= 0 to Wave_Degree do
       Wavelet_Coefficient[l,j] :=
           Pascal [j] * Coefficient[Poly_Degree-1,Poly_Degree-1,l-j+1];

For l := 0 to (3*Wave_Degree-2) do
  begin
     Sum[l] := 0.0;
     For j := 0 to Wave_Degree do
        Sum[l] := Sum[l] + Wavelet_Coefficient[l,j];
     If Odd= true
       then Wavelet_Factor := -1 * Wavelet_factor;
     Writeln(Wavelet_Factor*Sum[l]:2:8);
  end; {for-loop}

End. {main}
```

The Bernstein-Bezier coefficients and the wavelet coefficients are stored in arrays, as it can be seen from the type declaration. The B-B coefficients are in a 3-dimensional array [m,k,x] where the first index is the order, the second is the coefficient number, and the third is the number of the subinterval. Similarly, the B-wavelet coefficients are stored in a 2-dimensional array [l,m] where the first index represents the coefficient number and the second index is for the order of the B-wavelet function.

The first procedure, Second_Order_Polynomial, sets the initial condition. That is, it always starts with the B-spline of order 2. The next procedure, N_Order_Polynomial makes use of the recurrence relation

$$a_{l+1}^{m}(k) = a_l^{m}(k) + 1/(m-1)\,[a_l^{m-1}(k) - a_l^{m-1}(k-1)].$$

This procedure is repeated until the order is twice that of the user's input (i.e. the order of the B-wavelet.) This procedure also sets everything outside the support of the B-spline to 0. Then the B-B coefficients on each unit interval are listed.

The wavelet coefficients are calculated and listed using the relation

$$\psi_m(x) = \sum_{l} q_l^{m}\, N_m(2x-l),\quad 0 \le l \le 3m-2$$

$$\text{where } q_l^{m} = (-1)^{l}/2^{m-1} \sum_{j=0}^{m} m!/j!(m-j)!\; N_{2m}(l-j+1).$$

The following are the resulting wavelet coefficients for the inputs 1, 2, 3, and 4.

```
Enter B-wavelet degree. [1..4]
1
Coefficients of B-spline with order 2 :
Interval 1 :  0.0000  1.0000
Interval 2 :  1.0000  0.0000

Coefficients of B-wavelet with order 1 :
1.00000000
-1.00000000
```

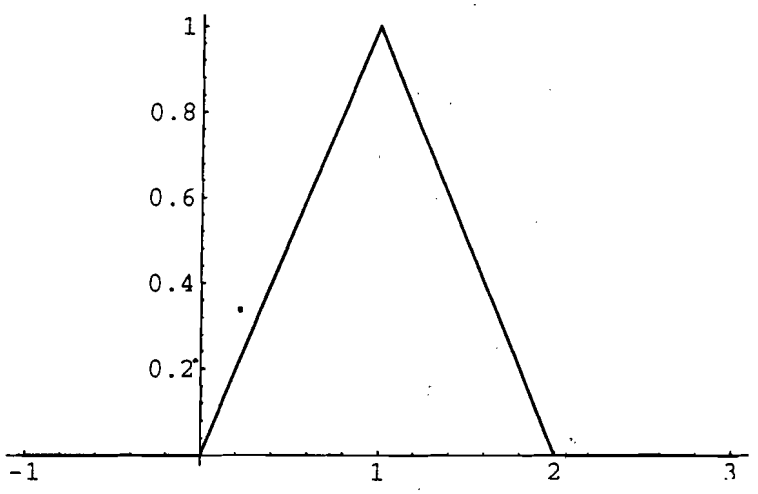
```
Enter B-wavelet degree. [1..4]
2
Coefficients of B-spline with order 4 :
Interval 1 :  0.0000  0.0000  0.0000  0.1667
Interval 2 :  0.1667  0.3333  0.6667  0.6667
Interval 3 :  0.6667  0.6667  0.3333  0.1667
Interval 4 :  0.1667  0.0000  0.0000  0.0000

Coefficients of B-wavelet with order 2 :
0.08333333
-0.50000000
0.83333333
-0.50000000
0.08333333
```

```
Enter B-wavelet degree. [1..4]
3
Coefficients of B-spline with order 6 :
Interval 1 :  0.0000  0.0000  0.0000  0.0000  0.0000  0.0083
Interval 2 :  0.0083  0.0167  0.0333  0.0667  0.1333  0.2167
Interval 3 :  0.2167  0.3000  0.4000  0.5000  0.5500  0.5500
Interval 4 :  0.5500  0.5500  0.5000  0.4000  0.3000  0.2167
Interval 5 :  0.2167  0.1333  0.0667  0.0333  0.0167  0.0083
Interval 6 :  0.0083  0.0000  0.0000  0.0000  0.0000  0.0000

Coefficients of B-wavelet with order 3 :
0.00208333
-0.06041667
0.30625000
-0.63125000
0.63125000
-0.30625000
0.06041667
-0.00208333
```

```
Enter B-wavelet degree. [1..4]
4
Coefficients of B-spline with order 8 :
Interval 1 :  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0002
Interval 2 :  0.0002  0.0004  0.0008  0.0016  0.0032  0.0063  0.0127  0.0238
Interval 3 :  0.0238  0.0349  0.0508  0.0730  0.1032  0.1421  0.1877  0.2363
Interval 4 :  0.2363  0.2849  0.3365  0.3873  0.4317  0.4635  0.4794  0.4794
Interval 5 :  0.4794  0.4794  0.4635  0.4317  0.3873  0.3365  0.2849  0.2363
Interval 6 :  0.2363  0.1877  0.1421  0.1032  0.0730  0.0508  0.0349  0.0238
Interval 7 :  0.0238  0.0127  0.0063  0.0032  0.0016  0.0008  0.0004  0.0002
Interval 8 :  0.0002  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000  0.0000

Coefficients of B-wavelet with order 4 :
0.00002480
-0.00307540
0.04159226
-0.19603175
0.45838294
-0.60178571
0.45838294
-0.19603175
0.04159226
-0.00307540
0.00002480
```

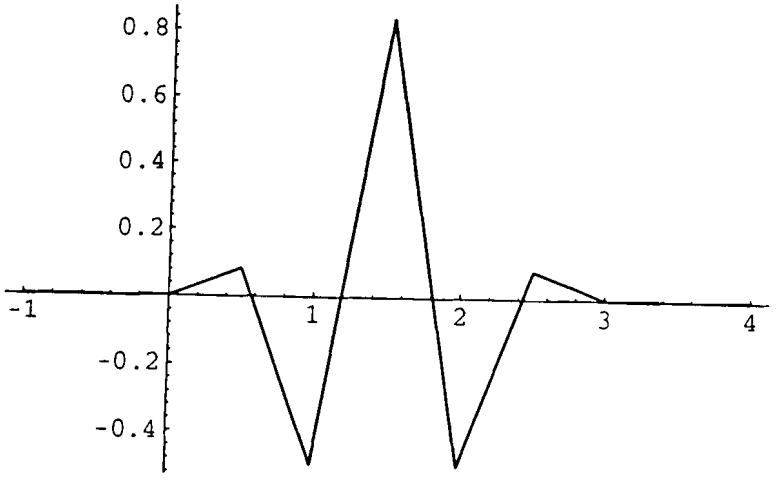
The actual graphs of $N_2(x)$, $\psi_2(x)$, $N_3(x)$, and $\psi_3(x)$ were drawn using *Mathematica*.

$N_2(x)$



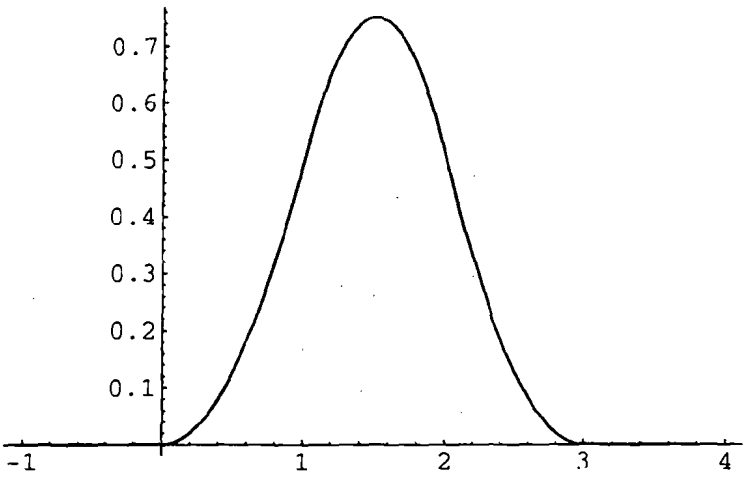
$$\psi_2(x) = \sum q_1^{\,n} N_n(2x-1) = \sum q_1^{\,2} N_2(2x-1)$$

$$= 0.083333\, N_2(2x) - 0.5N_2(2x-1) + 0.833333N_2(2x-2) - 0.5N_2(2x-3) + 0.083333N_2(2x-4)$$



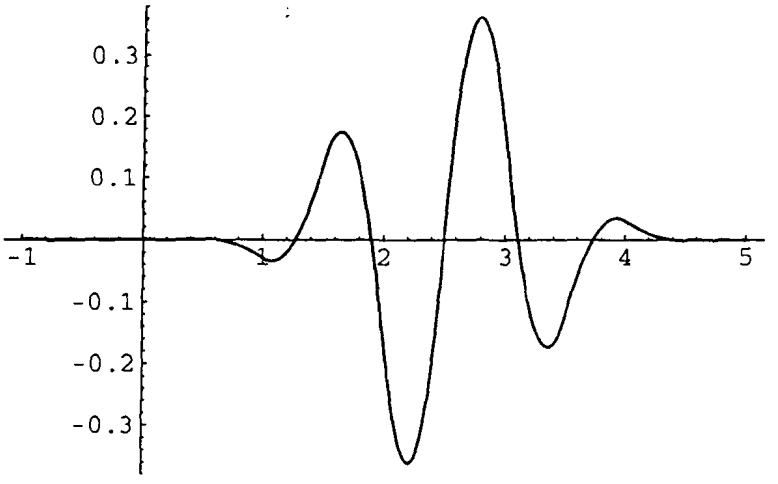
$\psi_2(x)$ is a linear combination of $N_2(x)$ with the appropriate wavelet coefficients.

$N_3(x)$



$$\psi_3(x) = \sum q_l^3 \, N_3(2x-l)$$

$$= 0.002083\, N_3(2x) - 0.0604167\, N_3(2x-1) + 0.30625\, N_3(2x-2)$$
$$-0.63125\, N_3(2x-3) + 0.63125\, N_3(2x-4) - 0.30625\, N_3(2x-5)$$
$$+0.0604167\, N_3(2x-6) - 0.002083\, N_3(2x-7)$$



$\psi_3(x)$ is a linear combination of $N_3(x)$ with the corresponding wavelet coefficients.

If the B-wavelet functions of higher order can be graphed, then they will be continuous and smoother, unlike the Haar wavelets mentioned earlier.

7. Conclusion

The Haar wavelet is in fact the B-wavelet function of order one. We can see that it is not smooth since the scaling function is not smooth either. However, using a recurrence relation, higher order B-splines can be constructed, which are relatively smooth. So, higher order B-splines as the scaling function produce smooth wavelets.

Using the recurrence relation, we may construct higher order B-wavelet functions with more oscillations and smoothness, which can be used to approximate functions with sensitive changes. Important applications can be seen in areas such as signal processing, where only a local part of the signal needs to be studied, but details cannot be lost. Wavelet functions are good tools in such applications because of the the advantages of using smaller waves rather than the bigger sine and cosine waves used in the Fourier method, which may distort approximations if there are sharp changes in the signal.

References

[1] Daubechies, Ingrid "Orthonormal Bases of Compactly Supported Wavelets," Communications on Pure and Applied Mathematics Vol. XLI, 1988.

[2] He, Tian-Xiao "Spline Interpolation and Its Wavelet Analysis, Comp. Math. Appl." To appear.

[3] Mallat, Stephane G. "Multiresolution Approximations and Wavelet Orthogonal Bases of $L^2(R)$," Transactions of the American Mathematical Society Vol. 315 No.1, Sept. 1989.

[4] Strang, Gilbert "Wavelet Transforms Versus Fourier Transforms," ·Bulletin of the American Mathematical Society Vol.28 No.2, April.1993.

[5] Walter, Gilbert G. "Wavelets : A New Tool in Applied Mathematics," The UMAP Journal, Vol.14 No.2, 1993.