Comparison of algorithms for the fast computation of the continuous wavelet transform

Michael J. Vrhel, Chulhee Lee, and Michael Unser

Biomedical Engineering and Instrumentation Program, Bldg. 13, Room 3N17,
National Center for Research Resources, National Institutes of Health,
Bethesda, MD 20892, USA

ABSTRACT

We introduce a general framework for computing the continuous wavelet transform (CWT). Included in this framework is an FFT implementation as well as fast algorithms which achieve $O(1)$ complexity per wavelet coefficient. The general approach that we present allows a straight forward comparison among a large variety of implementations. In our framework, computation of the CWT is viewed as convolving the input signal with wavelet templates that are the oblique projection of the ideal wavelets into one subspace orthogonal to a second subspace. We present this idea and discuss and compare particular implementations.

Keywords: wavelet, continuous wavelet transform, fast algorithms, oblique projection.

1. INTRODUCTION

The continuous wavelet transform (CWT) is an often used tool in the analysis of non-stationary and fractal signals (e.g. EEG)[8, 16]. The typically long length of these signals and the large number of scales computed in their analysis provides a motivation for considering fast algorithms to compute the CWT. In this paper, we present a general framework for computing the CWT. This framework encompasses fast algorithms with complexity $O(1)$ per wavelet coefficient as well as an FFT implementation. Here we define the real CWT of the signal $s(t)$ as the inner product

$$W_\psi s(\alpha,\tau) = \frac{1}{\sqrt{\alpha}}\left\langle s(t), \psi\left(\frac{\tau - t}{\alpha}\right)\right\rangle = \frac{1}{\sqrt{\alpha}}\int_{-\infty}^{+\infty} s(t)\psi\left(\frac{\tau - t}{\alpha}\right)dt \qquad (1)$$

where α and τ are respectively the continuously varying scaling and shifting parameters, and the real function $\psi(t)$ is the mother wavelet* .

In practice, the variables α and τ are sampled over the plane of values. Fast algorithms exist for computing the wavelet transform at the dyadic scales $\alpha = 2^i$ when the wavelet is associated with a multi-resolution [1, 3, 7, 11]. In this paper, we are interested in a finer sampling of the scale axis.

Previous methods for calculating the CWT include an approach which obtained the coefficients at the integer sample values $\alpha = i$, $\tau = k$, with $O(N)$ number of operations per scale [16]. Another approach which computes the CWT along arbitrary scales, again with $O(N)$ complexity per scale, is discussed in [13]. This last method is restricted to Gabor-like wavelets (i.e., modulated Gaussians). A method which achieved $O(N)$ operations per scale, with no restrictions on the shape of the wavelet, and with arbitrarily fine exponential sampling along the scale axis was introduced in [17]. This last method approximated wavelets at several scales by their orthogonal projection onto a space defined by a compact scaling

* : To simplify the notation throughout the paper, we use a definition of the wavelet transform that is a time-reversed version of the conventional one.

0-8194-2213-4/96/$6.00

function. Most other algorithms to date are implemented using an FFT-based approach and require $O(N \log(N))$ computations per scale [4, 5, 6, 9].

Here we introduce a general framework that includes the method [17] and the FFT implementation as particular cases, while providing additional flexibility. In the most efficient implementation, the algorithm consists of an FIR filter bank (P FIR filters for computing P scales or voices per octave) and a fast recursive IIR filter. This system is shown in Fig. 1 where the additional FIR filter $[h(k)]_{\uparrow 2^i}$ performs the two scale stretching of the approximating scaling function. This two-scale relationship enables us to use the same FIR filters in computing the CWT for P scales within each octave (with appropriate zero padding between the filter coefficients). In the simplest form, the filter coefficients for the FIR filter bank are analytically determined by integration of the wavelet

$$q_{\alpha_i}(k) = \int_{k-1/2}^{k+1/2} \psi(t/\alpha_i)\, dt, \qquad i = 0,\ldots,P-1,$$

where the α_i's are the corresponding scale parameters.

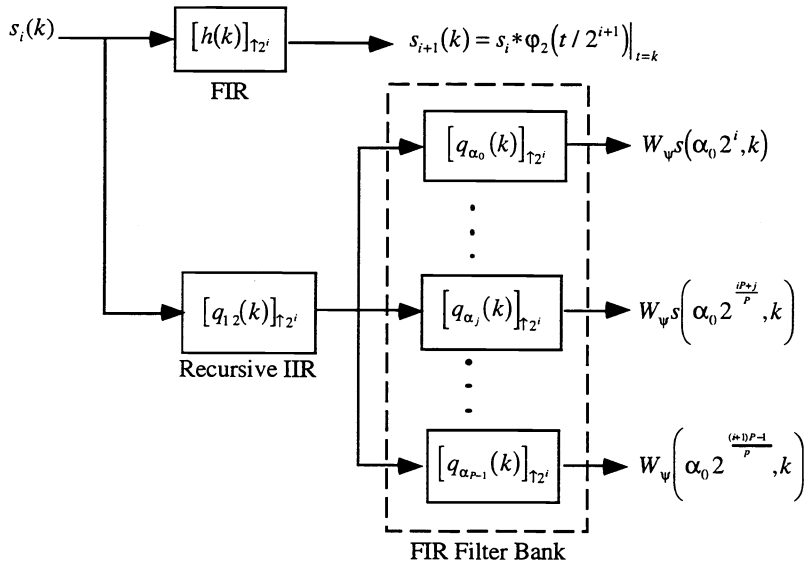

Fig. 1. System diagram for the computation of the CWT in the ith octave. All filters are expanded by the factor 2^i using zero padding. The same (but expanded) FIR filter bank is used for each octave. The scale axis is sampled exponentially.

2. THE GENERAL FRAMEWORK

Our goal is to efficiently compute the convolution given in (1) where the support of the wavelet template varies by the scale variable. As mentioned, the scale and shift variables are sampled over the plane of values. In addition, in most applications the input signal will consist of samples and therefore the continuous convolution given in (1) is approximated by digital filtering. Since we know the exact shape of our wavelet (as opposed to only having samples of the input signal), we consider the problem of approximating the continuous wavelet such that the digital filtering is performed efficiently while providing us with an acceptable approximation to the continuous convolution. In this paper, we approximate the wavelet by its oblique projection which is a generalization of an orthogonal projection since there are two subspaces involved. One subspace defines where the signal is projected and the other subspace defines the direction of the projection (the projection direction is orthogonal to this second subspace). We denote the set of projected wavelets at P scales by $\{\psi_i(t) \equiv \alpha_i^{-1/2}\, \psi(t/\alpha_i)\}_{i=0,\ldots,P-1}$ which are the oblique projections of the wavelets $\{\alpha_i^{-1/2}\, \psi(t/\alpha_i)\}_{i=0,\ldots,P-1}$ into a space

defined by the function $\varphi_2(t)$ in the direction orthogonal to the space defined by the function $\varphi_1(t)$. We now consider the properties that these functions must satisfy and introduce the oblique projection operation.

2.1. Scaling functions and oblique projections

An Lth order scaling function is a function $\varphi(t) \in L_2$ which satisfies the following three conditions:

(i) $0 < A \le \sum_{k\in Z} |\hat{\varphi}(\omega + 2\pi k)|^2 \le B < +\infty$

(ii) $\hat{\varphi}(0) = 1$ and $\hat{\varphi}^{(m)}(2\pi k) = 0$, $k \in Z, k \neq 0$ for $m = 0,..., L-1$

(iii) $\varphi\left(\frac{t}{2}\right) = \sum_{k\in Z} h(k)\varphi(t-k)$

where $\hat{\varphi}(\omega)$ is the Fourier transform of $\varphi(t)$ and $\hat{\varphi}^{(m)}(2\pi k)$ is the mth derivative of $\hat{\varphi}(\omega)$ evaluated at $2\pi k$. Property (i) insures that the subspace

$$V(\varphi) = \left\{ h(t) = \sum_{k\in Z} c(k)\varphi(t-k) \qquad c \in l_2 \right\}$$

is a well defined (closed) subspace of L_2[2]. Property (ii) implies that $\varphi(t)$ reproduces all polynomials of degree $L-1$. Property (iii) is referred to as the two-scale relation and it allows us to dilate all the wavelet filters by a power of two. An analysis function is defined as a function that satisfies only properties (i) and (ii). Criteria for the selection of a scaling or analysis function should be based on its approximating power, smoothness, and shortness of its support to name a few.

For a pair of analysis functions φ_1 and φ_2, the oblique projection of the wavelet $\psi_{\alpha_i}(t) = \psi(t/\alpha_i)$ into $V(\varphi_2)$ orthogonal to $V(\varphi_1)$ can be expressed in terms of the basis generated by φ_2 and a set of coefficients $p_{\alpha_i}(k)$ where

$$\psi_i(t) = \sum_{k\in Z} p_{\alpha_i}(k)\varphi_2(t-k). \qquad (2)$$

The direction of the projection is orthogonal to the space defined by φ_1 which implies the following

$$\left\langle \sum_{k\in Z} p_{\alpha_i}(k)\varphi_2(t-k) - \psi(t/\alpha_i), \varphi_1(t-l) \right\rangle = 0 \qquad l \in Z.$$

This orthogonality condition gives rise to the following equation

$$p_{\alpha_i}(k) = \left(q_{\alpha_i} * q_{12}\right)(k). \qquad (3)$$

where

$$q_{\alpha_i}(k) = \left\langle \psi_{\alpha_i}(t), \varphi_1(t-k) \right\rangle. \qquad (4)$$

and the digital filter q_{12} is the convolution inverse of the cross-correlation sequence $\langle \varphi_1(t-k), \varphi_2(t) \rangle$; i.e. $Q_{12}(z) = \left(\sum_{k\in Z} \langle \varphi_1(t-k), \varphi_2(t) \rangle z^{-k} \right)^{-1}$ (cf. [14]).

Equations (2)-(4) describe how the oblique projection can be computed. If ψ and φ_1 both have compact support then the FIR filters in the filter bank of Fig. 1 $\left\{ q_{\alpha_i} \right\}_{i=0,...P-1}$, also have finite support. Note also that we have an orthogonal projection if $\varphi_1(t) \in V(\varphi_2)$.

2.2 CWT approximation

We will now replace the computation of the convolution in (1) by its approximation

$$\tilde{W}_\psi s(\alpha_i, \tau) = (s * \psi_i)(\tau). \qquad (5)$$

Substituting (2) into the above equation we obtain:

$$s_0(t) = (s * \varphi_2)(t) \qquad (6)$$

$$\tilde{W}_\psi s(\alpha_i, \tau) = \sum_{k\in Z} p_{\alpha_i}(k) s_0(\tau - k). \qquad (7)$$

In principle, we need a filter $p_{\alpha_i}(k)$ for every scale we wish to compute. If φ_2 is a scaling function, then we can take advantage of the two scale relationship given as property (iii) which allows us to use the same set of filters for each octave.

In practice, the input signal will actually be the sample values $s[k] = s(t)|_{t=k}$. Computing the continuous convolution in (6) requires an interpolation of these sample values. Alternatively, one can use a Riemann approximation for the convolution integral resulting in the simple initialization

$$s_o[k] \cong (s * b_2)(k),$$

where $b_2(k) = \varphi_2(t)|_{t=k}$.

Incorporating property (iii) into (6) and (7), sampling, and performing simple algebraic manipulations, we obtain the following algorithm

$$s_{i+1}(k) = \left(s_i * [h]_{\uparrow 2^i}\right)(k) \tag{8}$$

$$\tilde{s}_i(k) = \left(s_i * [q_{12}]_{\uparrow 2^i}\right)(k) \tag{9}$$

$$\tilde{W}_\psi s(2^i \alpha_j, k) = \left(\tilde{s}_i * \left[q_{\alpha_j}\right]_{\uparrow 2^i}\right)(k), \qquad j = 0,\ldots,P-1, \tag{10}$$

where $[h]_{\uparrow 2^i}$ is $h(k)$ with $2^i - 1$ zeros between each sample (i.e., expanded by a factor of 2^i). Equation (8) performs the two scale filtering defined by property (iii) for φ_2; equation (9) is the IIR correction filtering which insures that we have an oblique projection of our wavelet; and equation (10) is the FIR filtering which constructs the approximated wavelets in terms of the scaling function and computes the wavelet coefficients. These equations describe the algorithm as it is shown in Fig. 1.

3. PARTICULAR IMPLEMENTATIONS

Now that we have introduced this general framework for computing the CWT, we will show several implementations. We start with the slowest most often used approach and end with the most efficient algorithm.

3.1. FFT Implementation

Consider the case when both $\varphi_2(t)$ and $\varphi_1(t)$ are the sinc interpolator. The filter q_{12}, which is the convolution inverse of the cross-correlation sequence $\langle \varphi_1(t-k), \varphi_2(t) \rangle$, becomes $q_{12}(k) = \delta(k)$. From (3) and (4) we then have

$$q_{\alpha_i}(k) = p_{\alpha_i}(k) = \left\langle \psi_{\alpha_i}(t), \varphi_1(t-k) \right\rangle. \tag{11}$$

Equation (11) states that the coefficients of the filter $p_{\alpha_i}(k)$ are sample values of the band-limited version of the wavelet $\psi_{\alpha_i}(t)$. Note that the filters $p_{\alpha_i}(k)$ are not compact and that the decay of the sinc function (and hence $p_{\alpha_i}(k)$) is very slow. For the sinc function, the two-scale relationship (property (iii)) requires that $h(k)$ be an ideal low pass filter, which will of course be IIR with the slow sinc decay. One approach to implement such a system is to do FIR approximations of the filters which would result in an $O(N)$ algorithm with a very large constant. Alternatively one could use an FFT algorithm and achieve and $O(N \log(N))$ efficiency per scale with a much smaller constant. Also one could use a different filter $p_{\alpha_i}(k)$ for each scale at each octave, which implies the system shown in Fig. 2 where $s_0(k) = s(k)$.

3.2. Cubic Spline Least Squares Implementation

We can improve on the efficiency of the algorithm given above by considering the case when $\varphi_1 = \varphi_2 = \beta^3$, where β^3 is the cubic B-spline. This selection provides us with a least squares (LS) projection into the cubic spline space. Since the B-splines are scaling functions with a compact 2-scale filter $h(k)$, we can make efficient use of the system shown in Fig. 1 for our implementation. For the cubic spline case, the FIR filters are given by

$$q_{\alpha_i}(k) = \langle \psi(t/\alpha_i), \beta^3(t-k) \rangle = \int_{k-1/2}^{k+1/2} \psi(t/\alpha_i)\beta^3(t-k)\,dt$$

$$H(z) = \frac{1}{8}\left(z^{-2} + 4z^{-1} + 6 + 4z + z^2\right)$$

and the IIR filter is

$$Q_{12}(z) = \frac{5040}{\left(z^3 + 120z^2 + 1191z + 2416 + 1191z^{-1} + 120z^{-2} + z^{-3}\right)}$$

which can be implemented in a fast recursive $O(N)$ algorithm [15]. If the wavelet is compact, then the filters $q_{\alpha_i}(k)$ will be FIR and the overall complexity of the system will be $O(N)$ per scale.

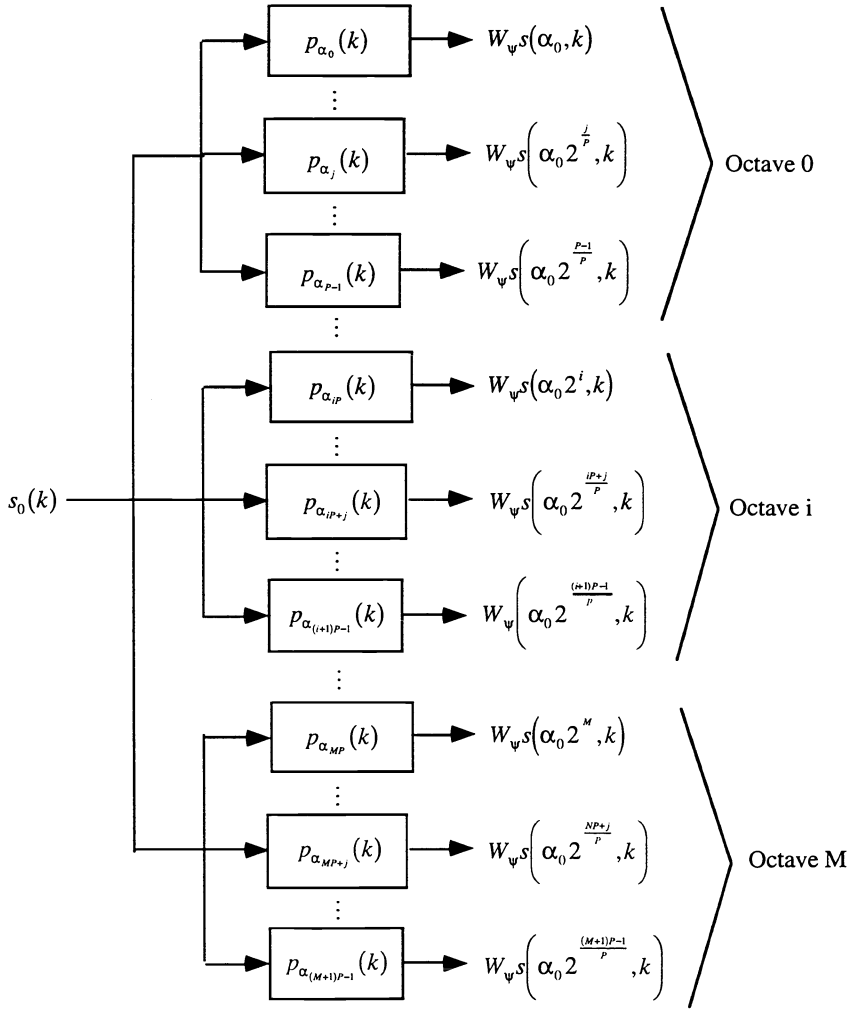

Figure 2: FFT Implementation for P scales per octave over M octaves with exponential sampling of the scale axis.

3.3. An Oblique Implementation

While the cubic B-spline implementation has advantages over the sinc implementation, we can make additional improvements in the system with our general framework. To keep things as simple as possible but still maintain a high degree of accuracy we will use the zero degree B-spline (first order approximation) on the analysis side: $\varphi_1 = \beta^0$; and the cubic B-spline on the synthesis side: $\varphi_2 = \beta^3$. The advantages of this particular selection are:

(i) The FIR filters $\{q_{\alpha_i}(k)\}_{i=0,\ldots,M-1}$ are determined analytically by simple integration of the wavelet $\psi(t/\alpha_i)$

$$q_{\alpha_i}(k) = \langle \psi(t/\alpha_i), \beta^0(t-k) \rangle = \int_{k-1/2}^{k+1/2} \psi(t/\alpha_i)\, dt \qquad (12)$$

(ii) The FIR filters are shorter and the IIR filter is of a lower order as compared to the orthogonal projection into $V(\varphi_2)$ considered in Sec. 3.2. This also implies that the oblique algorithm will be faster than the orthogonal (or least squares) algorithm.

(iii) The approximation error, discussed in the next section is very close to the minimum which is achieved by the orthogonal projection.

In this case, we gain in simplicity and speed with little loss in accuracy. The IIR filter is given by

$$Q_{12}(z) = \frac{384}{\left(z^2 + 76z + 230 + 76z^{-1} + z^{-2}\right)}$$

which can be implemented in a fast recursive fashion [15]. The two-scale filter $H(z)$ is the same as that given in Sec. 3.2 and the efficiency is $O(N)$ per scale.

4. APPROXIMATION ERROR

The approximation power of a scaling function depends upon its ability to reproduce polynomials up to a specific degree n. This degree plus one ($L = n+1$) provides the order of accuracy of the approximation function. The properties of the approximation error for the orthogonal projection case are given by the Strang-Fix conditions [12]. We will first describe the relationship between the oblique and the orthogonal approximation errors, and then translate this relationship to the Strang-Fix conditions.

The oblique projection into $V(\varphi_2)$, orthogonal to $V(\varphi_1)$, is related to the orthogonal projection into $V(\varphi_2)$ by the following (cf. [Unser, 1994 #287]):

$$\|\psi_\alpha - P_2\psi_\alpha\| \le \|\psi_\alpha - P_{2\perp 1}\psi_\alpha\| \le \frac{1}{\cos(\theta)}\|\psi_\alpha - P_2\psi_\alpha\| \qquad (13)$$

where P_2 is the orthogonal projection operator into $V(\varphi_2)$, $P_{2\perp 1}$ is the oblique projection operator into $V(\varphi_2)$ orthogonal to $V(\varphi_1)$, θ is the largest angle between $V(\varphi_1)$ and $V(\varphi_2)$, and

$$\cos(\theta) = \operatorname*{ess\,inf}_{\omega\in(0,1/2]} \frac{\sum_{k\in Z}\hat{\varphi}_2(\omega+2\pi k)\cdot\sum_{k\in Z}\hat{\varphi}_1^*(\omega+2\pi k)}{\sqrt{\sum_{k\in Z}|\hat{\varphi}_1(\omega+2\pi k)|^2}\cdot\sqrt{\sum_{k\in Z}|\hat{\varphi}_2(\omega+2\pi k)|^2}}.$$

As the angle between the subspaces decreases, the bound (13) becomes tighter and the oblique projection error approaches the orthogonal error. In particular, if $\varphi_1 \in V(\varphi_2)$, then $\cos(\theta) = 1$ and $V(\varphi_1) = V(\varphi_2)$.

As already mentioned, the behavior of the least squares approximation error as a function of the scale α is described by the Strang-Fix theory (cf. [17]). For the L_2th order scaling function φ_2 (cf. condition (ii)), the theory provides the relationship

$$\|\psi_\alpha - P_2\psi_\alpha\| \le C_{\varphi_2} \frac{\|\psi^{(L_2)}\|}{\alpha^{L_2}}$$

where the constant C_{φ_2} is a function of φ_2 and $\|\psi^{(L_2)}\|$ is the norm of the L_2th derivative of ψ. In other words, the error typically decays like $O(\alpha^{-L_2})$. This bound translates to a bound on the oblique projection in the following fashion using relationship (12)

$$\|\psi_\alpha - P_{2\perp 1}\psi_\alpha\| \le \frac{C_{\varphi_2}}{\cos(\theta)} \frac{\|\psi^{(L_2)}\|}{\alpha^{L_2}}.$$

Hence, we can expect the same kind of decay as in the least squares case.

In practice, the oblique error is much closer to the orthogonal error than what is indicated by the above worst case bound. In fact, it can be shown that the oblique and orthogonal errors have the same limiting behavior as $\alpha \to \infty$. Specifically, for α sufficiently large, the approximation error behaves as

$$\|\psi_\alpha - P_{2\perp 1}\psi_\alpha\| = C_2 \frac{\|\psi^{(L_2)}\|}{\alpha^{L_2}} + O\left(\frac{1}{\alpha^{L_2+1}}\right) \qquad (14)$$

where L_2 is again the order of approximation of φ_2 and

$$C_2 = \frac{1}{L_2!}\left(\sum_{k\neq 0}\left|\hat{\varphi}_2^{(L_2)}(2\pi k)\right|^2\right)^{1/2}.$$

This is exactly the same relation as the one given for the least squares case in [17] (including the value of the constants).

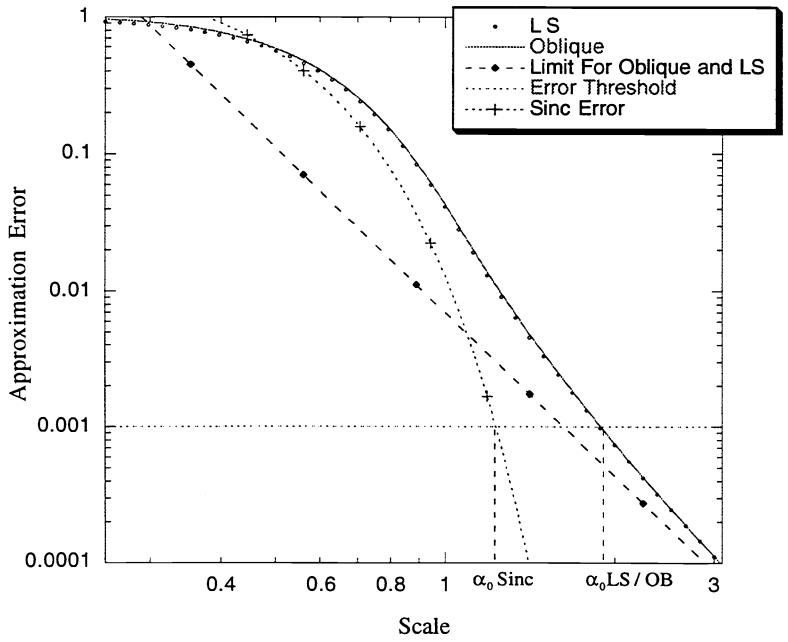

Fig. 3. Approximation errors for the 1st derivative Gaussian wavelet

Figure 3 displays the root square approximation error curve for the cubic LS approximation of a 1st derivative Gaussian wavelet and an oblique approximation using the zero and third degree B-splines for φ_1 and φ_2 respectively. Remarkably the difference between the LS and oblique errors is very small. It is also clear that both curves have the same asymptote which is given by (14) (for cubic splines, the value of the constant is $C_2 = 9.09241 \times 10^{-4}$). In both examples,

the loss of performance is negligible; in fact, the discrepancy is much less than the factor $(\cos(\theta))^{-1}$ (here $\cos(\theta) = 0.892$) predicted by the theory (worst case scenario). Thus, we can conclude that for all practical purposes the orthogonal and oblique projections are equivalent in terms of their approximation power. Also shown on the plot is the root square approximation error for a sinc based implementation which provides the anti-aliasing error as a function of scale.

5. COMPARISON OF METHODS

In this section, we compare the performance of the methods discussed in the previous section. For illustrative purposes, we consider a truncated version of the first derivative of a Gaussian as our wavelet shape, given by

$$\psi_{deriv}(t) = \begin{cases} -K_o \dfrac{t}{\alpha_o} e^{\frac{-(t/\alpha_o)^2}{2}} & ; \quad \left|\dfrac{t}{\alpha_o}\right| \le 5 \\ 0 & ; \quad \text{otherwise} \end{cases}$$

where K_o is a constant that insures that $\|\psi\| = 1$.

To select a fine scale resolution α_0, we selected an error threshold of 0.001 which is shown as a dash line in Fig. (3). By selecting 0.001 our fine scale resolution is α_0 Sinc=1.23 for the sinc implementation, α_0 LS= 1.88 for the cubic spline least squares implementation, and α_0 OB=1.89 for the first derivative wavelet in the oblique case of Sec. 3.3. We used P=12 scales per octave where $\alpha_i = \alpha_o 2^{i/12}$ $i = 0,...,11$ which corresponds to the musical notes $(A, A^{\#}, B, C, C^{\#}, ...)$.

We implemented the algorithm in MATLAB with specific routines (zero-padded filtering) coded in C to speed up the computations. The impulse response of the oblique system over four octaves is shown in Figure 4. For the oblique and the least squares implementation, the fine scale within each octave contains the worst case error. In Figure 5, the fine scale slice in the fourth octave of Figure 4 is compared to the actual function. Note that the oblique projection is almost indistinguishable from the actual function.

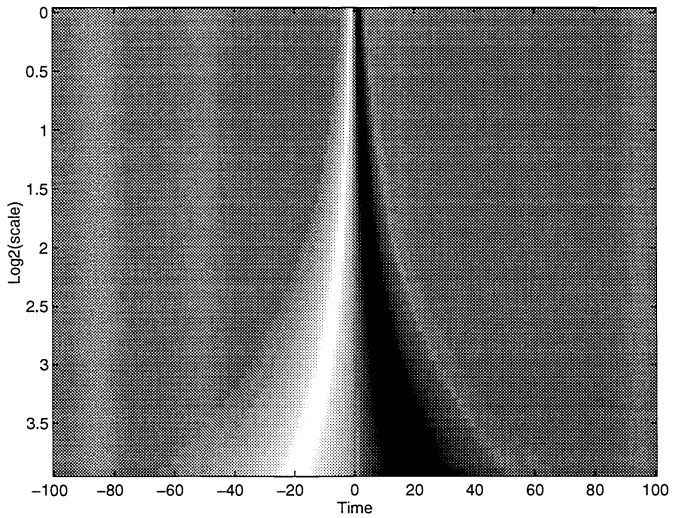

Fig. 4. Impulse response of the oblique implementation for the 1st derivative Gaussian wavelet.

We compared the performance of the oblique and LS algorithms to an FFT-based method (cf. Fig. 2) which used a radix-2 algorithm when the signal length N was a power of 2, and a mixed radix method for other signal lengths (MATLAB's FFT algorithm). The input was an electroencephalograph signal (EEG). Fast algorithms for analyzing such signals are of interest for applications that require real time detection of brain seizures [10].

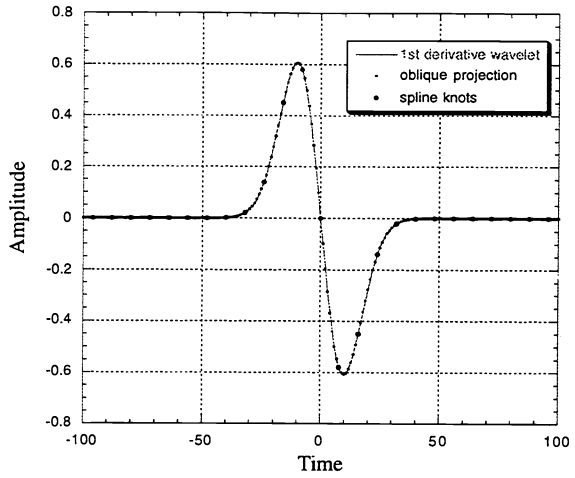

Fig. 5. Comparison of the 1st derivative Gaussian wavelet and its oblique approximation.

The length of the signal was varied and the time required to compute four octaves as a function of the signal length is shown in Figure 6. The dashed line with the x markers represents the FFT algorithm when the signal was padded to a power of two. The solid line with the small circles is the FFT algorithm for various signal lengths. The solid line, which is the oblique projection algorithm, clearly demonstrates the $O(N)$ characteristics of the method. Just above the oblique projection line is the cubic LS line. In this comparison, the oblique and LS algorithms appears to be advantageous even for relatively small signal lengths e.g. $N<128$. The speed improvement of the oblique algorithm over the LS is roughly 11%. The more important advantage of the oblique over the LS though is that the determination of the wavelet filters is much more straightforward. For the oblique algorithm, the majority of the computation was spent in the FIR filter bank. Additional speed up could be achieved by performing each FIR filter in parallel.

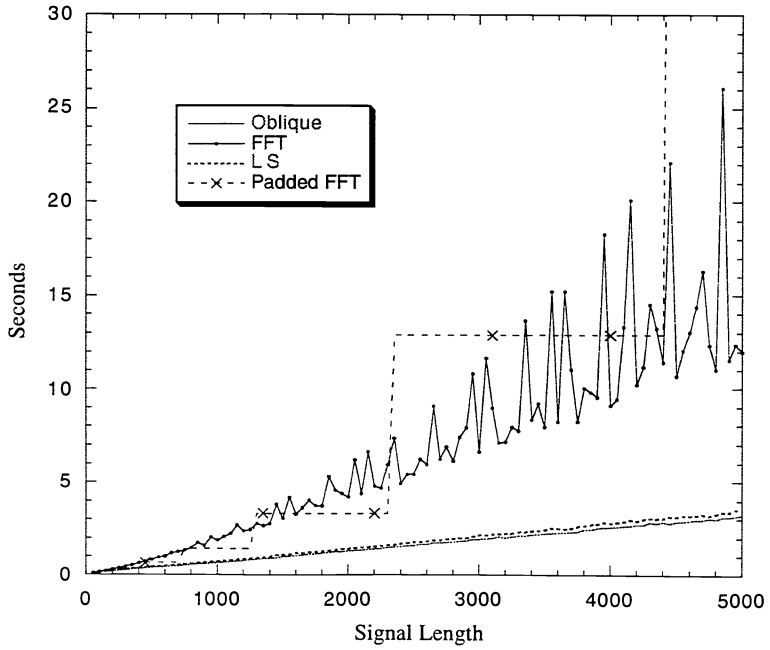

Fig. 6. Speed comparison of the FFT, Cubic spline LS, and Oblique

6. CONCLUSION

We have introduced a general framework that encompasses many approaches to implementing the CWT and allows a straightforward comparison and understanding of the methods. The results indicate that the use of spline spaces have several advantages. These advantages include a complexity of $O(1)$ per computed wavelet coefficient, easily computed FIR filter coefficients (for the oblique implementation), control over the approximation error, and flexibility in regards to wavelet shape.

7. REFERENCES

[1] P. Abry and A. Aldroubi, "Designing multiresolution analysis-type wavelets and their fast algorithms", *Journal of Fourier Analysis and Applications*, Vol. 2, No. 2, 1995, pp. 135-159.

[2] A. Aldroubi and M. Unser, "Sampling procedures in function spaces and asymptotic equivalence with Shannon's sampling theory", *Numer. Funct. Anal. and Optimiz.*, Vol. 15, No. 1&2, February 1994, pp. 1-21.

[3] P. Dutilleux, "An Implementation of the algorithme à trous to compute the wavelet transform", *Proc. Wavelets: Time-Frequency Methods and Phase Space*, 1989, pp. 298-304.

[4] D.L. Jones and R.G. Baraniuk, "Efficient approximation of continuous wavelet transforms", *Elect. Lett.*, Vol. 27, No. 9, 1991, pp. 748-750.

[5] S. Maes, "A Fast Quasi-Continuous Wavelet Transform Algorithm", *Proc. Workshop on Time, Frequency, Wavelets and Multiresolution Theory*, INSA-Lyon, 1994,

[6] S. Maes, "The wavelet transform in signal processing, with application to the extraction of the speech modulation model features", Ph.D., Universite Catholique De Louvain, 1994.

[7] S.G. Mallat, "A theory of multiresolution signal decomposition: the wavelet representation", *IEEE Trans. PAMI*, Vol. PAMI-11, No. 7, 1989, pp. 674-693.

[8] J.F. Muzy, E. Barcy and A. Areneodo, "Wavelets and multifractal formalism for singular signals: Application to turbulence data", *Physical Review Letters*, Vol. 67, No. 25, 1991, pp. 3515-3518.

[9] O. Rioul and P. Duhamel, "Fast algorithms for discrete and continuous wavelet transforms", *IEEE Trans. Info. Th.*, Vol. IT-38, No. 2, 1992, pp. 569-586.

[10] S.J. Schiff, A. Aldroubi and M. Unser, "Controlled wavelet transforms for electroencephalographic spike detection", *Proc. Second Annual Computation and Neural System Meeting*, Washinghton D.C., 1993,

[11] M.J. Shensa, "The discrete wavelet transform : wedding the à trous and Mallat algorithms", *IEEE Trans. Sig. Proc.*, Vol. 40, No. 10, 1992, pp. 2464-2482.

[12] G. Strang and G. Fix, "A Fourier analysis of the finite element variational method", in: *Constructive Aspect of Functional Analysis*, Edizioni Cremonese, Rome, 1971, pp. 796-830.

[13] M. Unser, "Fast Gabor-like windowed Fourier and continuous wavelet transforms", *IEEE Sig. Proc. Lett.*, Vol. 1, No. 5, 1994, pp. 76-79.

[14] M. Unser and A. Aldroubi, "A general sampling theory for non-ideal acquisition devices", *IEEE Trans. Sig. Proc.*, Vol. 42, No. 11, 1994, pp. 2915-2925.

[15] M. Unser, A. Aldroubi and M. Eden, "B-spline signal processing. Part II : efficient design and applications", *IEEE Trans. Signal Processing*, Vol. 41, No. 2, February 1993, pp. 834-848.

[16] M. Unser, A. Aldroubi and S.J. Schiff, "Fast implementation of the continuous wavelet transform with integer scales", *IEEE Trans. Sig. Proc.*, Vol. 42, No. 12, 1994, pp. 3519-3523.

[17] M.J. Vrhel, C. Lee and M. Unser, "Fast Continuous Wavelet Transform", *Proc. IEEE ICASSP 95*, Detroit, MI, 1995, pp. 1165-1168.