International Journal of Modern Physics C, Vol. 13, No. 6 (2002) 771–776
© World Scientific Publishing Company

DISCRETE WAVELETS AND FILTERING CHAOTIC SIGNALS

WILLI-HANS STEEB[*,†,‡], YORICK HARDY[†], and RUEDI STOOP[*,†]

[*]*University of Applied Sciences, Solothurn, Switzerland*

[†]*International School for Scientific Computing at Rand Afrikaans University*
P. O. Box 524, Auckland Park 2006, South Africa

[‡]*whs@na.rau.ac.za*

Received 17 December 2001
Revised 19 February 2002

We study the behavior of one-dimensional chaotic signals and filtering using the discrete Haar wavelet and Daubechies wavelets.

Keywords: Discrete wavelets; filter; chaos.

Most discrete signals are filtered using linear difference equations, for example

$$y[n] = ay[n-1] + (1-a)x[n]\,, \quad n = 0, 1, \ldots\,, \tag{1}$$

where $x[n]$ is the input signal, $y[n]$ the output, $y[-1] = 0$, $x[0] = 0$, and a $(0 < a < 1)$ is the filter parameter. Notice that $y[n]$ depends upon previously computed values of the filter output. Thus the chosen filter is an infinite response filter. It is well-known that filtered chaotic signal can exhibit increases in observed fractal dimensions,[1–5] for example the Ljapunov dimension. In the present note we study the Ljapunov exponent of filtered chaotic one-dimensional maps using wavelets.

Let us first summarize some facts about the discrete wavelet transform.[6–10] We notice that within the discrete wavelet transform we distinguish between redundant discrete systems (frames) and orthonormal, biorthonormal, ... bases of wavelets. In our case the discrete wavelet transform (or DWT) is an orthogonal function, which can be applied to a finite group of data. Functionally, it is very much like the discrete Fourier transform, in that the transforming function is orthogonal, a signal passed twice (i.e., a forward and a backward transform) through the transformation is unchanged, and the input signal is assumed to be a set of discrete-time samples. Both transforms are convolutions. Whereas the basis function of the Fourier transform is sinusoidal, the wavelet basis is a set of functions, which are defined by a recursive difference equation for the scaling function ϕ:

$$\phi(x) = \sum_{k=0}^{M-1} c_k \phi(2x - k)\,, \tag{2}$$

where the range of the summation is determined by the specified number of nonzero coefficients M. Here k is the translation parameter. The number of the coefficients is not arbitrary and is determined by the constraints of orthogonality and normalization. Owing to the periodic boundary condition we have

$$c_k \equiv c_{k+nM} \,, \tag{3}$$

where $n \in \mathbf{N}$. We notice that periodic wavelets are only one possibility to deal with signals defined on an interval. Generally, the area under the scaling function over all space should be unity, i.e.,

$$\int_{\mathbf{R}} \phi(x)\, dx = 1 \,. \tag{4}$$

From Eq. (2) it follows that

$$\sum_{k=0}^{M-1} c_k = 2 \,. \tag{5}$$

In the Hilbert space $L_2(\mathbf{R})$, the function ϕ is orthogonal to its translations; i.e.,

$$\int_{\mathbf{R}} \phi(x)\phi(x-k)\, dx = 0 \,, \quad k \neq 0 \,. \tag{6}$$

What is desired is a function ψ, which is also orthogonal to its dilations, or scales, i.e.,

$$\int_{\mathbf{R}} \psi(x)\psi(2x-k)\, dx = 0 \,. \tag{7}$$

Such a function ψ does exist and is given by (the so-called associated wavelet function):

$$\psi(x) = \sum_{k=1}^{M} (-1)^k c_{1-k}\phi(2x-k) \,, \tag{8}$$

which is dependent on the solution of ϕ. The following equation follows from the orthonormality of scaling functions

$$\sum_k c_k c_{k-2m} = 2\delta_{0m} \,, \tag{9}$$

which means that the above sum is zero for all m not equal to zero, and that the sum of the squares of all coefficients is two. Another equation, which can be derived from $\psi(x) \perp \phi(x-m)$ is:

$$\sum_k (-1)^k c_{1-k} c_{k-2m} = 0 \,. \tag{10}$$

A way to solve for ϕ is to construct a matrix of coefficient values. This is a square $M \times M$ matrix where M is the number of nonzero coefficients. The matrix is designated L with entries

$$L_{ij} = c_{2i-j} \,. \tag{11}$$

This matrix has an eigenvalue equal to 1, and its corresponding (normalized) eigenvector contains, as its components, the value of the function ϕ at integer values of x. Once these values are known, all other values of the function ϕ can be generated by applying the recursion equation to get values at half-integer x, quarter-integer x, and so on, down to the desired dilation. This determines the accuracy of the function approximation.

An example for ψ is the Haar function

$$\psi(x) := \begin{cases} 1 & 0 \le x < \dfrac{1}{2}\,, \\ -1 & \dfrac{1}{2} \le x < 1\,, \\ 0 & \text{otherwise}\,, \end{cases} \tag{12}$$

and ϕ is given by:

$$\phi(x) = \begin{cases} 1 & 0 \le x < 1\,, \\ 0 & \text{otherwise}\,. \end{cases} \tag{13}$$

The functions

$$\psi_{m,n}(x) := 2^{m/2}\psi(2^m x - n)\,, \quad m, n \in \mathbf{Z}\,, \tag{14}$$

form a basis in the Hilbert space $L_2(\mathbf{R})$. If we restrict m to $m = 0, 1, 2, \ldots$ and

$$n = 0, 1, 2, \ldots, 2^m - 1\,, \tag{15}$$

we obtain a basis in the Hilbert space $L_2[0, 1]$.

This class of wavelet functions is constrained, by definition, to be zero outside of a small interval. This is what makes the wavelet transform able to operate on a finite set of data, a property which is formally called compact support. The recursion relation ensures that a scaling function ϕ is nondifferentiable everywhere. Of course this is not valid for Haar wavelets. The following table (Table 1) lists coefficients for three wavelet transforms. We notice that for Daubechies-6 the sum of the coefficients is normalized to $\sqrt{2}$ and not to 2 as it is for the first two cases.

The pyramid algorithm operates on a finite set on N input data $x_0, x_1, \ldots, x_{N-1}$, where N is a power of two; this value will be referred to as the input block size. These data are passed through two convolution functions, each of which creates

Table 1. Coefficients for three wavelet functions.

Wavelet	c_0	c_1	c_2	c_3	c_4	c_5
Haar	1.0	1.0				
Daubechies-4	$\frac{1}{4}(1+\sqrt{3})$	$\frac{1}{4}(3+\sqrt{3})$	$\frac{1}{4}(3-\sqrt{3})$	$\frac{1}{4}(1-\sqrt{3})$		
Daubechies-6	0.332671	0.806891	0.459877	−0.135011	−0.085441	0.035226

an output stream that is half the length of the original input. These convolution functions are filters, one half of the output is produced by the "low-pass" filter:

$$a_i = \frac{1}{2} \sum_{j=0}^{N-1} c_{2i-j+1} x_j \,, \quad i = 0, 1, \ldots, \frac{N}{2} - 1 \,, \tag{16}$$

and the other half is produced by the "high-pass" filter function:

$$b_i = \frac{1}{2} \sum_{j=0}^{N-1} (-1)^j c_{j-2i} x_j \,, \quad i = 0, 1, \ldots, \frac{N}{2} - 1 \,, \tag{17}$$

where N is the input block size, c_j are the coefficients, x is the input function, and a and b are the output functions. In the case of the lattice filter, the low- and high-pass outputs are usually referred to as the odd and even outputs, respectively. In many situations, the odd or low-pass output contains most of the information content of the original input signal. The even, or high-pass output contains the difference between the true input and the value of the reconstructed input if it was to be reconstructed from only the information given in the odd output. In general, higher order wavelets (i.e., those with more nonzero coefficients) tend to put more information into the odd output, and less into the even output. If the average amplitude of the even output is low enough, then the even half of the signal may be discarded without greatly affecting the quality of the reconstructed signal. An important step in wavelet-based data compression is finding wavelet functions, which cause the even terms to be nearly zero. However, note that details can only be neglected for very smooth time series and smooth wavelet filters, a situation which is not satisfied for chaotic time signals.

The Haar wavelet represents a simple interpolation scheme. After passing these data through the filter functions, the output of the low-pass filter consists of the average of every two samples, and the output of the high-pass filter consists of the difference of every two samples. The high-pass filter contains less information than the low pass output. If the signal is reconstructed by an inverse low-pass filter of the form:

$$x_j^L = \sum_{i=0}^{N/2-1} c_{2i-j+1} a_i \,, \quad j = 0, 1, \ldots, N-1 \,, \tag{18}$$

then the result is a duplication of each entry from the low-pass filter output. This is a wavelet reconstruction with $2\times$ data compression. Since the perfect reconstruction is a sum of the inverse low-pass and inverse high-pass filters, the output of the inverse high-pass filter can be calculated. This is the result of the inverse high-pass filter function

$$x_j^H = \sum_{i=0}^{N/2-1} (-1)^j c_{j-1-2i} b_i \,, \quad j = 0, 1, \ldots, N-1 \,. \tag{19}$$

The perfectly reconstructed signal is:

$$x = x^L + x^H \,, \tag{20}$$

where x is the vector with elements x_j. Using other coefficients and other orders of wavelets yields similar results, except that the outputs are not exactly averages and differences, as in the case using the Haar coefficients.

Our input signal is given as the solution of the logistic map

$$x[t+1] = 4x[t](1 - x[t]) \,, \tag{21}$$

where $x[0] \in [0,1]$. The properties of the logistic map are well-known.[11] For almost all initial values we find that the Ljapunov exponent of the logistic map is given by $\ln(2)$. We apply the Haar wavelet to filter the time series $(x_0, x_1, \ldots, x_{N-1})$ resulting from the logistic map in the chaotic regime. The filter decomposes the signal into the moving average coefficients $\{a_i\}$ and details $\{b_i\}$ with $i = 0, 1, \ldots, N/2 - 1$. Owing to Broomhead *et al.*[4] the complete set of coefficients $\{a_i, b_i\}$ have the same Ljapunov exponent as the original time series. The reason is that Haar (and Daubechies) wavelets are finite impulse response filters and that the convolution is performed only once. Therefore we deal with a nonrecursive finite impulse filter. Using the time series given by the logistic map (21) we evaluate x_j^L, x_j^H using Eqs. (16), (18) and (19). We have $x_j^L = x_{j+1}^L$ for j even and $x_j^H = -x_{j+1}^H$ for j even. We use the series x_j^L for the calculation of the Ljapunov exponent. Notice that in the case of the logistic equation (21) the even output resulting from Haar wavelets, i.e., the details, contain nearly as much information as the odd output, i.e., the averages. We remove the duplication of each entry of x_j^L and calculate the Ljapunov exponent of this time series. Thus the size of the time series is $N/2$. An algorithm to find the Ljapunov exponent from time series is described by Steeb.[11] The Ljapunov exponent is much larger than $\ln(2)$ for this time series, i.e., the time series becomes more chaotic.

Using the coefficients for the Daubehies-4 and Daubechies-6 wavelet we find similar results, i.e., the Ljapunov exponent is much larger than $\ln(2)$. Furthermore, for other one-dimensional chaotic maps we also find similar results.

References

1. R. Badii, G. Broggi, B. Derighetti, and M. Ravani, *Phys. Rev. Lett.* **60**, 979 (1988).
2. F. Mitschke, M. Möller, and W. Lange, *Phys. Rev. A* **37**, 4518 (1988).
3. A. Chennaoui, K. Pawelzik, W. Liebert, H. G. Schuster, and G. Pfister, *Phys. Rev. A* **41**, 4151 (1990).
4. D. S. Broomhead, J. P. Huhe, and M. R. Muldoon, *J. R. Stat. Soc. B* **54**, 373 (1992).
5. M. T. Rosenstein and J. J. Collins, *Comput. Graphics* **18**, 587 (1994).
6. Y. Hardy and W.-H. Steeb, *Classical and Quantum Computing with C++ and Java Simulations* (Birkhäuser Publishing, Basel, 2001).
7. G. Strang, *Siam Rev.* **31**, 613 (1989).
8. G. Erlebacher, M. Y. Hussaini, and L. M. Jameson (eds.), *Wavelets: Theory and Applications* (Oxford University Press, Oxford, 1996).

9. J. J. Benedetto and M. W. Frazier (eds.), *Wavelets*: *Mathematics and Applications* (CRC Press, BocaRaton, 1994).
10. W.-H. Steeb, *Hilbert Spaces*, *Wavelets*, *Generalised Functions and Modern Quantum Mechanics* (Kluwer Academic Publishers, Dordrecht, 1998).
11. W.-H. Steeb, *The Nonlinear Workbook*: *Chaos*, *Fractals*, *Cellular Automata*, *Neural Networks*, *Genetic Algorithm*, *Fuzzy Logic with C++*, *Java*, *SymbolicC++*, *and Reduce Programs* (World Scientific, Singapore, 1999).