Quantum Information and Computation, Vol. 9, No. 5&6 (2009) 0414–0422

QUANTUM WAVELET TRANSFORMS OF ANY ORDER

FRANCISCO ARGÜELLO
Department of Electronic and Computing, University of Santiago de Compostela
15782, Santiago de Compostela, Spain

Received January 22, 2008
Revised December 15, 2008

Many classical algorithms are known to efficiently compute the wavelet transforms. However, those classical algorithms cannot be directly translated to quantum algorithms. Recently, efficient and complete quantum algorithms for two representative wavelet transforms (quantum Haar and quantum Daubechies of fourth order) have been proposed. In this paper, we generalize these algorithms in order to they can be applied to Daubechies wavelet kernels of any order. Specifically, we develop a method that efficiently factorize those kernels. The factorization is compatible with the existing pyramidal and packet quantum wavelet algorithms. All steps of the algorithm are unitary and easily implementable on a quantum computer.

Keywords: Quantum computing, Unitary transform, Wavelet
Communicated by: K Moelmer & B Kane

1 Introduction

A quantum computer [1] works with qubits where each qubit can be a zero, a one, or a quantum superposition of both, allowing for an infinite number of states. In other words, a qubit can exist simultaneously as both 0 and 1, with a numerical coefficient representing the probability for each state. An algorithm for a quantum computer must initialize those qubits in some specified state. In each step of the algorithm, the quantum computer operates by manipulating those qubits with quantum logic gates. The existence of superposition states in the qubit provides a new dimension to the concept of computing. Because of superposition states in qubits, a quantum processor can perform calculations using all possible input values simultaneously. So, with only one calculation, all the possible outputs corresponding to all the possible inputs (combinations of zeros and ones) can be computed. In other words, a quantum computer provides a massive parallelism achieved through superposition. To realize the full potential of quantum computers, new algorithms must be designed that exploit quantum parallelism fully.

The quantum computation is encoded into the unitary time evolution of a quantum mechanical state vector [2]. The unitary character of the process ensures it is invertible. Therefore, the paradigm of computation as a quantum process implies that all quantum computations have to be unitary. In order words, it has to be assured that all steps of a quantum algorithm are restricted to unitary transformations. This excludes all the classically computable operations that cannot be written as unitary transformations. The bad news is that

many computations that we would like to perform are not originally described in terms of unitary operations.

Algorithm design is a complicated task, and in quantum computing it becomes even more complicated. As each intermediate operation must be unitary, we might need to be quite creative in encoding a desired computation on a quantum computer. Fortunately, the unitary transforms such as Fourier, Walsh-Hadamard, cosine, Hartley, and wavelet transforms are describable naturally in terms of unitary operations [3, 4]. The kernel of these transformations can be expressed, in matrix form, as an unitary matrix. However, classical algorithms usually include non unitary operations and therefore they cannot be used in quantum computing. This is the case of the wavelet algorithms based on lifting steps [5], which are not unitary operations. Quantum algorithms for the Fourier [1], cosine [6, 7] and Hartley [8] transform have been proposed.

There is a class of unitary transforms, the wavelet transforms, which are as useful as the Fourier transform for some applications. Wavelet transforms are used to expose the multi-scale structure of a signal. A wavelet is a kind of mathematical function used to divide a given function or continuous-time signal into different frequency components and study each component with a resolution that matches its scale [9, 10]. The wavelet transform separates low and high frequencies, just as the Fourier transform. The advantage is that wavelets are localized in time, since they only are defined on part of the interval of the data, as opposed to the trigonometric functions used in Fourier analysis which are defined on the entire interval.

A few years after the discovery of the quantum Fourier algorithm, it has been shown that certain wavelet transforms can also be implemented on a quantum computer in a polynomial number of quantum gates. In fact, explicit quantum circuits were developed for the most popular discrete wavelet transforms, namely, the Haar and Daubechies of fourth order wavelets, both for pyramidal and packet algorithms [4, 11, 12, 13]. As it happens in classical signal analysis, it is natural to expect that quantum wavelet transforms will find important future applications for the treatment of quantum images, quantum databases, quantum data compression, and other applications [14].

Høyer [11] presented a general technique for developing fast quantum algorithms for computing unitary transforms. It is based on a routine which uses a generalized Kronecker product. This technique directly gives quantum networks for computing unitary transforms which are known to be expressible by generalized Kronecker products. Applications include re-development of the networks for computing the Walsh-Hadamard and the quantum Fourier transform. Also, new quantum networks implementing the kernel of two wavelet transforms are found this way. The wavelet kernels considered in that paper are the quantum Haar and quantum Daubechies of fourth order ($D^{(4)}$) kernels.

Fijany and Williams [4] developed fast algorithms and efficient circuits for quantum wavelet transforms. First, they analyze the feasibility and efficiency of the implementation of the packet and pyramid algorithms by using a given wavelet kernel. They also develop efficient and complete gate-level circuits for two representative wavelet kernels, the Haar and Daubechies of fourth order kernels. The approach used is to factor the operators for these transforms into direct sums, direct products and dot products of unitary matrices. Surprisingly, they found that the operations that are easy and inexpensive to implement classically are not always easy and inexpensive to implement quantum mechanically, and vice versa. In particular, the

computational cost of performing certain permutation matrices is ignored classically because they can be avoided explicitly. However, quantum mechanically, these permutation operations must be performed explicitly and hence their cost enters into the full complexity measure of the quantum transform.

In this paper we present a quantum algorithm for computing Daubechies wavelets of any order. It is based on a generalization of the factorization proposed by Høyer [11] for the Daubechies wavelet kernel $D^{(4)}$. Specifically, we develop a method that efficiently factorize any Daubechies wavelet kernel $D^{(k)}$. With this factorization, we can directly apply the quantum pyramidal and quantum packet algorithms developed by Fijany and Williams [4]. All steps of the algorithm are unitary and easily implementable on a quantum computer. The organization of the rest of the paper is as follows: in section 2, we describe the Høyer's factorization; the proposed generalized factorization is developed in section 3; and finally, in section 4, we present the conclusions.

2 Daubechies wavelet of fourth order

The Daubechies wavelet [9] of fourth order has four scaling function coefficients. Each step of the wavelet transform applies the wavelet function to the input data. The kernel of this wavelet for sequences of 2^n data elements can be written in matrix form (of size $2^n \times 2^n$) as

$$D_{2^n}^{(4)} = \begin{pmatrix} h_0 & h_1 & h_2 & h_3 & & & & & & & & \\ h_3 & -h_2 & h_1 & -h_0 & & & & & & & & \\ & & h_0 & h_1 & h_2 & h_3 & & & & & & \\ & & h_3 & -h_2 & h_1 & -h_0 & & & & & & \\ & & & & & & \ddots & & & & & \\ & & & & & & & h_0 & h_1 & h_2 & h_3 & \\ & & & & & & & h_3 & -h_2 & h_1 & -h_0 & \\ h_2 & h_3 & & & & & & & & h_0 & h_1 & \\ h_1 & -h_0 & & & & & & & & h_3 & -h_2 & \end{pmatrix}. \tag{1}$$

This form is suitable for a classical computation and because of its sparse structure, the application of $D_{2^n}^{(4)}$ can be realized with an optimal cost of $O(2^n)$. However, the matrix form of $D_{2^n}^{(4)}$ as given by (1) is not suitable for a quantum implementation. To achieve a feasible and efficient quantum implementation, a suitable factorization of $D_{2^n}^{(4)}$ need be developed [4]. Høyer [11] proposed a factorization of $D_{2^n}^{(4)}$ as

$$D_{2^n}^{(4)} = (I_{2^{n-1}} \otimes C_1) S_{2^n} (I_{2^{n-1}} \otimes C_0), \tag{2}$$

with

$$C_0 = 2 \begin{pmatrix} h_3 & -h_2 \\ h_2 & h_3 \end{pmatrix}, \quad C_1 = \frac{1}{2} \begin{pmatrix} \frac{h_0}{h_3} & 1 \\ 1 & \frac{h_1}{h_2} \end{pmatrix}, \tag{3}$$

where $h_0 = \frac{3+\sqrt{3}}{4\sqrt{2}}$, $h_1 = \frac{3-\sqrt{3}}{4\sqrt{2}}$, $h_2 = \frac{1-\sqrt{3}}{4\sqrt{2}}$, $h_3 = \frac{1+\sqrt{3}}{4\sqrt{2}}$ (it is not the standard coefficient ordering [9]), and

$$S_{2^n} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & & & & & & & & \ddots & & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \end{pmatrix}. \tag{4}$$

S_{2^n} is a permutation matrix with a quantum description given by

$$S_{2^n}|a\rangle = \begin{cases} |a\rangle & \text{if } a \text{ even}, \\ |(a-2)\mathrm{mod}2^n\rangle & \text{if } a \text{ odd}. \end{cases} \tag{5}$$

S_{2^n} can be implemented using a quantum circuit with a complexity of $O(n)$ [4, 11, 15]. Given the factorization of the wavelet kernel (2), its corresponding wavelet transform can be performed according to a pyramid algorithm or a packet algorithm. Fijany and Williams [4] proposed the factorization of these algorithms using permutation matrices and developed efficient quantum circuits that implement them.

3 Quantum wavelet of any order

The order of the wavelet refers to the number of coefficients. Daubechies orthogonal wavelets $D^{(2)}$ to $D^{(20)}$ (even orders only) are commonly used. Each wavelet has a number of vanishing moments equal to half the number of coefficients. For example, $D^{(2)}$ (the Haar wavelet) has two coefficients and one vanishing moment, $D^{(4)}$ has four coefficients and two vanishing moments, etc. Increasing the number of vanishing moments, the wavelet becomes smoother. In this section we will develop a method for factoring the kernel of the Daubechies wavelet of any order suitable for a quantum implementation. It is a generalization of the factorization given in Eq. (2) for the wavelet of fourth order.

Let $A_{2^n}^{(k)}$ (with k even) be a matrix with a similar form to the Daubechies wavelet kernel of kth order but with arbitrary coefficient values limited only by the unitary (orthogonality) condition. For example, the sixth order matrix $A_{2^n}^{(6)}$ can be written as

$$A_{2^n}^{(6)} = \begin{pmatrix} a_0 & a_1 & a_2 & a_3 & a_4 & a_5 & & & & & & \\ a_5 & -a_4 & a_3 & -a_2 & a_1 & -a_0 & & & & & & \\ & & a_0 & a_1 & a_2 & a_3 & a_4 & a_5 & & & & \\ & & a_5 & -a_4 & a_3 & -a_2 & a_1 & -a_0 & & & & \\ & & & & & & & & \ddots & & & \\ a_4 & a_5 & & & & & & & & a_0 & a_1 & a_2 & a_3 \\ a_1 & -a_0 & & & & & & & & a_5 & -a_4 & a_3 & -a_2 \\ a_2 & a_3 & a_4 & a_5 & & & & & & & & a_0 & a_1 \\ a_3 & -a_2 & a_1 & -a_0 & & & & & & & & a_5 & -a_4 \end{pmatrix}. \tag{6}$$

In the case that the coefficient values of $A_{2^n}^{(k)}$ are the same than those in the Daubechies matrix, we can write $A_{2^n}^{(k)} = D_{2^n}^{(k)}$ and $a_i = h_i$. When it is necessary to indicate in the coefficients the order of the matrix, we will write $a_i^{(k)}$ instead of a_i.

We can generalize the factorization (2) from the particular case of $D_{2^n}^{(4)}$ to the general case of $A_{2^n}^{(k)}$. Specifically, we can write a matrix of order k as the multiplication of a matrix of order $k-2$ and two auxiliary orthogonal matrices,

$$A_{2^n}^{(k)} = (I_{2^{n-1}} \otimes C_2) S_{2^n} A_{2^n}^{(k-2)}, \tag{7}$$

with S_{2^n} given by (4) and

$$C_2 = \begin{pmatrix} c_a & c_b \\ -c_b & c_a \end{pmatrix}. \tag{8}$$

Expression (7) implies a system of equations that relates the coefficients c_a, c_b and $a_i^{(k-2)}$ to the coefficients $a_i^{(k)}$. Solving these equations, we obtain the following expressions:

$$\begin{aligned}
\alpha &= \sqrt{(a_0^{(k)})^2 + (a_{k-1}^{(k)})^2}, && (9)\\
c_a &= \frac{1}{\alpha} a_0^{(k)}, && (10)\\
c_b &= -\frac{1}{\alpha} a_{k-1}^{(k)}, && (11)\\
a_0^{(k-2)} &= \alpha, && (12)\\
a_1^{(k-2)} &= -\alpha \frac{a_{k-2}^{(k)}}{a_{k-1}^{(k)}}, && (13)\\
a_i^{(k-2)} &= \frac{1}{\alpha}\left(a_0^{(k)} a_i^{(k)} + (-1)^i a_{k-1}^{(k)} a_{k-i-1}^{(k)}\right), \quad i = 2, \ldots, k-3. && (14)
\end{aligned}$$

This construction is linear in k, though for all practical purposes, k is a constant. By recursively applying Eq. (7) and expressions (9)-(14), the factorization of the Daubechies wavelet of any order k can be easily obtained as

$$D_{2^n}^{(k)} = (I_{2^{n-1}} \otimes C_{2,k}) S_{2^n} (I_{2^{n-1}} \otimes C_{2,k-2}) S_{2^n} \cdots (I_{2^{n-1}} \otimes C_{2,4}) S_{2^n} (I_{2^{n-1}} \otimes A_2^{(2)}), \tag{15}$$

where we have used a second subindex to distinguish matrices C_2 with different coefficient values. This equation is the generalization of (2) to any order k.

The factorization given by Eq. (15) has a efficient implementation on a quantum computer. All factors are unitary matrices and can be implemented with a complexity of $O(n)$. Figure 1 shows a block-level circuit implementation of this equation. The circuit can be used as kernel in the pyramid and packet quantum algorithms proposed in [4].

3.1 Example

As a example, we will develop the factorization of Daubechies wavelet kernel of sixth order, $D^{(6)}$. The kernel $D_{2^n}^{(6)}$ is given in matrix form by Eq. (6) with the followings coefficients [9]:

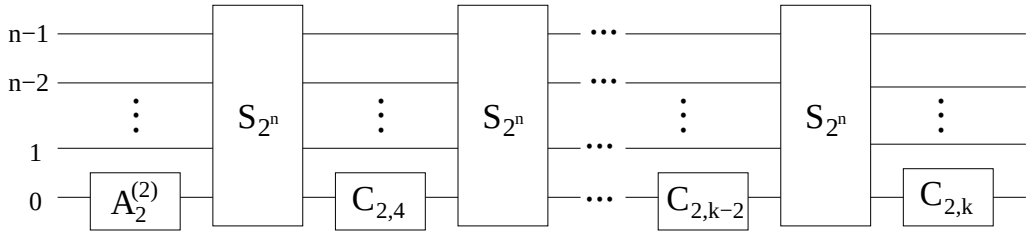

Fig. 1. A block-level circuit for implementation of $D_{2^n}^{(k)}$ based on Eq. (15).

$$\begin{aligned} h_0 &= 0.3326705529500825 \\ h_1 &= 0.8068915093110924 \\ h_2 &= 0.4598775021184914 \\ h_3 &= -0.1350110200102546 \\ h_4 &= -0.0854412738820267 \\ h_5 &= 0.0352262918857095 \end{aligned} \tag{16}$$

We apply recursively Eq. (7). In the first step, $D_{2^n}^{(6)} = (I_{2^{n-1}} \otimes C_{2,6}) S_{2^n} A_{2^n}^{(4)}$, and using expressions (9)-(14),

$$\alpha = \sqrt{h_0^2 + h_5^2}, \tag{17}$$

$$c_4 = \frac{1}{\alpha} h_0, \tag{18}$$

$$c_5 = -\frac{1}{\alpha} h_5, \tag{19}$$

$$a_0^{(4)} = \alpha, \tag{20}$$

$$a_1^{(4)} = -\alpha \frac{h_4}{h_5}, \tag{21}$$

$$a_2^{(4)} = \frac{1}{\alpha}(h_0 h_2 + h_5 h_3), \tag{22}$$

$$a_3^{(4)} = \frac{1}{\alpha}(h_0 h_3 - h_5 h_2). \tag{23}$$

In the second step, $A_{2^n}^{(4)} = (I_{2^{n-1}} \otimes C_{2,4}) S_{2^n} A_{2^n}^{(2)}$, and

$$\beta = \sqrt{(a_0^{(4)})^2 + (a_3^{(4)})^2}, \tag{24}$$

$$c_2 = \frac{1}{\beta} a_0^{(4)}, \tag{25}$$

$$c_3 = -\frac{1}{\beta} a_3^{(4)}, \tag{26}$$

$$a_0^{(2)} = \beta, \tag{27}$$

$$a_1^{(2)} = -\beta \frac{a_2^{(4)}}{a_3^{(4)}}. \tag{28}$$

As a result, we obtain the factorization of $D_{2^n}^{(6)}$ as

$$D_{2^n}^{(6)} = (I_{2^{n-1}} \otimes C_{2,6}) S_{2^n} (I_{2^{n-1}} \otimes C_{2,4}) S_{2^n} (I_{2^{n-1}} \otimes A_2^{(2)}), \tag{29}$$

with

$$A_2^{(2)} = \begin{pmatrix} a_0 & a_1 \\ a_1 & -a_0 \end{pmatrix}, \tag{30}$$

$$C_{2,4} = \begin{pmatrix} c_2 & c_3 \\ -c_3 & c_2 \end{pmatrix}, \tag{31}$$

$$C_{2,6} = \begin{pmatrix} c_4 & c_5 \\ -c_5 & c_4 \end{pmatrix}, \tag{32}$$

$$\alpha = \sqrt{h_0^2 + h_5^2}, \tag{33}$$

$$\beta = \alpha \sqrt{1 + \left(\frac{h_0 h_3 - h_5 h_2}{h_0^2 + h_5^2} \right)^2}, \tag{34}$$

$$a_0 = \beta, \tag{35}$$

$$a_1 = \beta \frac{h_0 h_2 + h_3 h_5}{h_2 h_5 - h_0 h_3}, \tag{36}$$

$$c_2 = \frac{\alpha}{\beta}, \tag{37}$$

$$c_3 = \frac{1}{\alpha\beta}(h_2 h_5 - h_0 h_3), \tag{38}$$

$$c_4 = \frac{1}{\alpha} h_0, \tag{39}$$

$$c_5 = -\frac{1}{\alpha} h_5. \tag{40}$$

Table 1 shows the coefficient values obtained with our method for some wavelet transforms.

4 Concluding remarks

In this paper we have proposed a fast quantum algorithm for wavelet transforms of any order. Specifically, we have developed a method for factorizing a generic Daubechies wavelet kernel. With this factorization, the wavelet kernel can be efficiency implemented with a complexity of $O(n)$. This implies the feasibility and efficiency of the quantum implementation of both the pyramid and packet algorithms by using our factorization for the wavelet kernel.

The quantum wavelet transform can be computed efficiently, but few applications are known. The most basic wavelet, the Haar transform, has been used in quantum searching, sorting, and element distinctness [16], and to replace the Fourier transform in the Grover algorithm [17]. This wavelet might also be applicable in other problems where the input function is guaranteed to be piecewise constant. For more smooth input functions, a Daubechies wavelet transform of higher order would probably do better [16]. We believe it is a very interesting question to study the use of the wavelet transforms and of other related transformations to replace the Fourier transform in many quantum applications.

Acknowledgements

This work was supported in part by the MCYT under contract TIN 2007-67537-C03-01 and by Xunta de Galicia under contract 08TIC001206PR.

References

1. M.A. Nielsen and I.L. Chuang (2000), *Quantum computation and quantum information*, Cambridge University Press.
2. G. Cybenko (1996), *Reducing quantum computations to elementary unitary operations*, IEEE Multimedia, Vol. 3, No. 2, pp. 27-32.
3. M. Rötteler, M. Püschel, and Thomas Beth (1999), *Fast signal transforms for quantum computers*, Proc. Workshop Physics and Computer Science, Werner Kluge (Ed.), pp. 31-43.
4. A. Fijany and C.P. Williams (1998), *Quantum wavelet transforms: fast algorithms and complete circuits*, Lecture Notes Comput. Sci., Vol. 1509: Selected papers from the First NASA Int. Conf. Quantum Computing and Quantum Communications, pp. 10-33.
5. I. Daubechies and W. Sweldens (1998), *Factoring wavelet transforms into lifting steps*, J. Fourier Anal. Appl., Vol. 4, No. 3, pp. 247-269.
6. C.-C. Tseng and T.-M. Hwang (2005), *Quantum circuit design of 8x8 discrete cosine transform using its fast computation flow graph*, Proc. IEEE Int. Symp. Circuits and Systems, Vol. 1, pp. 828-831.
7. A. Klappenecker and M. Rötteler (2001), *Discrete cosine transforms on quantum computers*, Proc. 2nd Int. Symp. Image and Signal Processing and Analysis (ISPA), pp. 464-468.
8. C.-C. Tseng and T.-M. Hwang (2004), *Quantum circuit design of 8x8 discrete Hartley transform*, Proc. IEEE Int. Symp. Circuits and Systems, Part 3, pp. III-397–III-400.
9. I. Daubechies (1992), *Ten lectures on wavelets*, SIAM.
10. Y. Meyer (1993), *Wavelets: algorithms and applications*, SIAM.
11. P. Høyer (1997), *Efficient quantum transforms*, quant-ph/9702028.
12. A. Klappenecker (1999), *Wavelets and wavelets packets on quantum computers*, Proc. Wavelet Applications in Signal and Image Processing VII, SPIE, pp. 703-713.
13. H. Ohnishi, H Matsueda, and L. Zheng (2005), *Quantum wavelet transform and matrix factorization*, Proc. Quantum Electronics Conf., pp. 1327-1328.
14. M. Terraneo and D.L. Shepelyansky (2003), *Imperfection effects for multiple applications of the quantum wavelet transform*, Phys. Rev. Lett., Vol. 90, No. 25, pp. 257902 (4).
15. V. Vedral, A. Barenco, and A. Ekert (1996), *Quantum networks for elementary arithmetic operations*, Phys. Rev. A, Vol. 54, No. 1, pp. 147-153.
16. P. Høyer, J. Neerbek, and Y. Shi (2002), *Quantum complexities of ordered searching, sorting, and element distinctness*, Algorithmica, Vol. 34, No. 4, pp. 429-448.
17. S. Parka, J. Baeb, and Y. Kwonc (2007), *Wavelet quantum search algorithm with partial information*, Chaos, Solitons & Fractals, Vol. 32, No. 4, pp. 1371-1374.

Table 1. Coefficient values obtained in the factorization of some wavelet transforms.

Daubechies $D^{(6)}$	Daubechies $D^{(12)}$
$a_0 = 0.3811623112547295$	$a_0 = 0.2199821557649779$
$a_1 = 0.9245081354314583$	$a_1 = 0.9755038960172864$
$c_2 = 0.8776586509122140$	$c_2 = 0.6072614608381266$
$c_3 = 0.4792862323069094$	$c_3 = 0.7945020567504809$
$c_4 = 0.9944404247714915$	$c_4 = 0.8614131902113027$
$c_5 = -0.1053007197520300$	$c_5 = -0.5079048294021096$
	$c_6 = 0.9717186051810308$
Daubechies $D^{(8)}$	$c_7 = 0.2361418055851860$
$a_0 = 0.3067400413031922$	$c_8 = 0.9975589756036319$
$a_1 = 0.9517933321214848$	$c_9 = -0.0698290068140210$
$c_2 = 0.7767099476830462$	$c_{10} = 0.9999533612666631$
$c_3 = 0.6298584421679204$	$c_{11} = 0.0096579134134792$
$c_4 = 0.9679887560642052$	
$c_5 = -0.2509935619359045$	Daubechies $D^{(14)}$
$c_6 = 0.9989436716376567$	$a_0 = 0.1926509552582037$
$c_7 = 0.0459515059065259$	$a_1 = 0.9812673486040248$
	$c_2 = 0.5429835256039748$
Daubechies $D^{(10)}$	$c_3 = 0.8397433482455683$
$a_0 = 0.2562893254436279$	$c_4 = 0.7999360902765198$
$a_1 = 0.9666001146615028$	$c_5 = -0.6000852035112311$
$c_2 = 0.6848711679863367$	$c_6 = 0.9403783419491656$
$c_3 = 0.7286641772867876$	$c_7 = 0.3401302309306516$
$c_4 = 0.9203874725141433$	$c_8 = 0.9900484148597817$
$c_5 = -0.3910075452456475$	$c_9 = -0.1407271694934331$
$c_6 = 0.9912472113188926$	$c_{10} = 0.9993149766026649$
$c_7 = 0.1320188094648594$	$c_{11} = 0.0370078037394183$
$c_8 = 0.9997830230904323$	$c_{12} = 0.9999896788726393$
$c_9 = -0.0208304282278630$	$c_{13} = -0.0045433630930678$