Proceedings of the International Conference on Computing Anticipatory Systems, pp. 286-295, 2001

A Quantum Analog to Basis Function Networks

Dan Ventura

Computer Science Department
Brigham Young University, Provo, UT USA 84602
phone: (801) 378-9075 fax: (801) 378-7775
email: ventura@cs.byu.edu web: axon.cs.byu.edu/Dan

Abstract. A Fourier-based quantum computational learning algorithm with similarities to classical basis function networks is developed. Instead of a Gaussian basis, the quantum algorithm uses a discrete Fourier basis with the output being a linear combination of the basis. A set of examples is considered as a quantum system that undergoes unitary transformations to produce learning. The main result of the work is a quantum computational learning algorithm that is unique among quantum algorithms as it does not assume *a priori* knowledge of a function *f*.

Keywords: Quantum Computing, Basis Function Networks, Fourier Analysis, Computational Learning

1 INTRODUCTION

The field of quantum computation investigates the power of the unique characteristics of quantum systems used as computational machines. Early quantum computational successes have been impressive, yielding results that are classically impossible [1] [2] [3] [4]. It is therefore natural also to suspect the existence of quantum computational *learning* algorithms that do things classically impossible. Classical Fourier analysis techniques have been used as an approach to computational learning, but the resulting algorithms must be bounded to avoid exponential time requirements. Since quantum computation can offer an exponential speedup in processing time and since quantum computation often contains Fourier transforms as a key element, it is natural to consider a quantum analog to classical Fourier-based computational learning.

Basis function networks [5] [6] [7] [8] represent functions as linear combinations of nonlinear functions. The most well-known employ a hidden layer of radial basis functions with a Gaussian response followed by an output layer of nodes with linear response. That is, the hidden nodes' response is Gaussian based on their distance from the input stimulus, and the output nodes' response is a linear combination of the hidden node responses. However, basis functions other than Gaussian can be used [9].

This paper considers one such approach, presenting a Fourier-based quantum learning algorithm. It introduces a unique approach to computational learning, extending it into the quantum realm by considering a set of examples as a quantum system that undergoes unitary transformations to produce learning. As a result, the paper demonstrates that quantum computational learning is a feasible and potentially fruitful area for research.

An outline of the paper’s approach is as follows. As an initial step, Section 2 presents an introduction to classical Fourier-based learning. An intermediary step, discussed in Section 3, reinterprets the Fourier analytical approach as a linear algebraic formulation. Section 4 reformulates the approach a final time as a quantum computational algorithm. Section 5 provides some discussion of the proposed algorithm and focuses particularly on the issues of noise and generalization, and Section 6 concludes the paper final remarks and directions for further research.

2 FOURIER-BASED LEARNING

Here is given a brief description of Fourier-based learning as it applies to our approach. The subject of discrete Fourier analysis is well developed and is treated only very briefly here. For a more in depth presentation see any book on Fourier analysis, for example [10].

A bipolar-valued binary function *f*: $\{0,1\}^n \rightarrow \{-1,1\}$ can be represented as a Fourier expansion (the expansion used here is actually a simplified Fourier expansion called a Walsh transform)

$$f(\vec{x}) = \sum_{\vec{a}\in\{0,1\}^n} \hat{f}(\vec{a})\chi_{\vec{a}}(\vec{x}) \tag{1}$$

where the Fourier basis functions $\chi_{\vec{a}}(\vec{x})$ are defined as

$$\chi_{\vec{a}}(\vec{x}) = (-1)^{\vec{a}^{\mathrm{T}}\vec{x}} \tag{2}$$

and the Fourier coefficients being given by

$$\hat{f}(\vec{a}) = \frac{1}{2^n}\sum_{\vec{b}\in\{0,1\}^n} f(\vec{b})\chi_{\vec{b}}(\vec{a}) \tag{3}$$

Actually, in the general case Eq. 3 should use χ^* (complex conjugate of χ); however, in the simplified case considered here (bipolar rather than complex output), $\chi^* = \chi$.

Classical machine learning includes methods that employ such Fourier analytical methods for learning *f* by approximating a polynomial number of large Fourier coefficients [11] [12] [13]. In order to determine the set A of large coefficients, the methods require access to a membership oracle. The large coefficients, $\hat{f}(\vec{a})$ (for $\vec{a} \in A$) are then approximated by

$$\tilde{\hat{f}}(\vec{a}) = \frac{1}{m}\sum_{\vec{x}\in T} f(\vec{x})\chi_{\vec{x}}(\vec{a}) \tag{4}$$

with T being a set of *m* carefully chosen examples of the form $\vec{x} \rightarrow f(\vec{x})$ with $\vec{x} \in \{0,1\}^n$ and $f(\vec{x}) \in \{-1,1\}$. Finally, using the fact that the function can be at least weakly approximated [14] with a polynomial number of large coefficients (the set A) and using Eq. 4 to approximate those coefficients, the function *f* may be approximated by

$$\tilde{f}(\vec{x}) = \sum_{\vec{a}\in A} \tilde{\hat{f}}(\vec{a})\chi_{\vec{a}}(\vec{x}) \tag{5}$$

We propose here a Fourier-based quantum learning algorithm that uses only an example oracle (i.e. a training set) rather than the membership oracle required by the classical machine learning techniques. In other words, the classical machine learning approach requires the ability to choose which examples will be used to learn the function (which in typical learning problems we can not do); on the other hand, the method that is proposed here makes no such requirement -- a standard training set suffices.

2.1 A Fourier Example

A simple example will help illustrate the concept of a Fourier expansion. Let *n* = 2 and

$$f = \begin{cases} 00 \rightarrow \ \ 1 \\ 01 \rightarrow \ \ 1 \\ 10 \rightarrow -1 \\ 11 \rightarrow \ \ 1 \end{cases}$$

To calculate the Fourier basis functions use Eq. 2 and for example,

$$\chi_{00}(00) = -1^{(00)\binom{0}{0}} = -1^{0} = 1$$

and

$$\chi_{01}(11) = -1^{(01)\binom{1}{1}} = -1^{1} = -1$$

The other fourteen values for the four Fourier functions are calculated similarly. Next, calculate the Fourier coefficients using Eq. 3. For example,

$$\hat{f}(11) = \frac{1}{4}\left(f(00)\chi_{00}(11) + f(01)\chi_{01}(11) + f(10)\chi_{10}(11) + f(11)\chi_{11}(11)\right)$$

That is,

$$\hat{f}(11) = \frac{1}{4}\left((1)(1) + (1)(-1) + (-1)(-1) + (1)(1)\right) = \frac{1}{2}$$

and the other coefficients are found in the same manner. Finally, the Fourier expansion of *f* can be written using Eq. 1,

$$f(\vec{x}) = \hat{f}(00)\chi_{00}(\vec{x}) + \hat{f}(01)\chi_{01}(\vec{x}) + \hat{f}(10)\chi_{10}(\vec{x}) + \hat{f}(11)\chi_{11}(\vec{x})$$

Which in this example evaluates to

$$f(\vec{x}) = \frac{1}{2}\chi_{00}(\vec{x}) - \frac{1}{2}\chi_{01}(\vec{x}) + \frac{1}{2}\chi_{10}(\vec{x}) + \frac{1}{2}\chi_{11}(\vec{x})$$

and solving for any of the values 00, 01, 10, or 11 will result in the appropriate output for *f*. Now if instead of knowing *f*, we only have a training set such as

$$T = \begin{cases} 00 \rightarrow \ \ 1 \\ 01 \rightarrow \ \ 1 \\ 10 \rightarrow -1 \end{cases}$$

then the coefficients are approximated using Eq. 4 instead of Eq. 3. For example now

$$\tilde{f}(11) = \frac{1}{3}\left(f(00)\chi_{00}(11) + f(01)\chi_{01}(11) + f(10)\chi_{10}(11)\right)$$

which simplifies to

$$\tilde{\hat{f}}(11) = \frac{1}{3}\left((1)(1) + (1)(-1) + (-1)(-1)\right) = \frac{1}{3}$$

The Fourier expansion is now approximated using Eq. 5 instead of Eq. 1

$$\tilde{f}(\vec{x}) = \tilde{\hat{f}}(00)\chi_{00}(\vec{x}) + \tilde{\hat{f}}(01)\chi_{01}(\vec{x}) + \tilde{\hat{f}}(10)\chi_{10}(\vec{x}) + \tilde{\hat{f}}(11)\chi_{11}(\vec{x})$$

which simplifies to

$$\tilde{f}(\vec{x}) = \frac{1}{3}\chi_{00}(\vec{x}) - \frac{1}{3}\chi_{01}(\vec{x}) + \frac{3}{3}\chi_{10}(\vec{x}) + \frac{1}{3}\chi_{11}(\vec{x})$$

Now solving for 00, 01, or 10 will give 4/3, 4/3, and -4/3 respectively. While these are not the correct values, they are the correct sign. On the other hand, solving for 11 results in 0, which is equivalent to “don’t know”. This will be addressed further in Section 5.

3 LINEAR ALGEBRAIC FORMULATION

The next step is to reformulate the Fourier equations in matrix form. Note that the $\chi_{\vec{a}}$ can be considered vectors in a 2^n-dimensional space indexed by $\vec{x} \in \{0,1\}^n$. These $\chi_{\vec{a}}$ form an orthonormal basis for the function space with inner product

$$\left\langle \chi_{\vec{a}}, \chi_{\vec{b}} \right\rangle = E_x\left[\chi_{\vec{a}}(\vec{x})\chi_{\vec{b}}(\vec{x})\right] = \begin{cases} 1 & \text{if } \vec{a} = \vec{b} \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

Again, in the general case one of the χ in Eq. 6 should be χ^*, but for the special case of bipolar outputs $\chi^* = \chi$. Now let $\boldsymbol{B}$ be the matrix formed by taking the $\chi_{\vec{a}}$ as the rows. Because of the orthonormality of the basis, the columns are also formed by the $\chi_{\vec{a}}$. Also, define *f* as a vector in an 2^n-dimensional space indexed by $\vec{x} \in \{0,1\}^n$ such that

$$f_{\vec{x}} = f(\vec{x}) \tag{7}$$

Then

$$\frac{1}{2^n}\boldsymbol{B}f = \hat{f} \tag{8}$$

gives the Fourier coefficients as a vector in a 2^n-dimensional space indexed by $\vec{a} \in \{0,1\}^n$ so that

$$\hat{f}_{\vec{a}} = \hat{f}(\vec{a}) \tag{9}$$

To evaluate $\chi_{\vec{a}}(\vec{x})$, define *y* as the 2^n-vector indexed by $\vec{b} \in \{0,1\}^n$ such that

$$y_{\vec{b}} = \begin{cases} 1 & \text{if } \vec{b} = \vec{x} \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

and calculate

$$\boldsymbol{B}y = \chi \tag{11}$$

Now χ is a vector in a 2^n-dimensional space indexed by $\vec{b} \in \{0,1\}^n$, the $\vec{a}$ th element of which is equal to $\chi_{\vec{a}}(\vec{x})$. Finally, the Fourier representation of *f* is given as

$$f(\vec{x}) = (\boldsymbol{B}y)^{\dagger}(\frac{1}{2^n}\boldsymbol{B}f) \tag{12}$$

As in Section 2, *f* may be approximated by approximating the Fourier coefficients using a set of *m* examples, T. Define $\widetilde{f}$ as a 2^n-vector indexed by $\vec{x} \in \{0,1\}^n$ such that

$$\widetilde{f}_{\vec{x}} = \begin{cases} f(\vec{x}) & \text{if } \vec{x} \in T \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

Then the Fourier representation of the approximate function is

$$\widetilde{f}(\vec{x}) = (\boldsymbol{B}y)^{\dagger}(\frac{1}{m}\boldsymbol{B}\widetilde{f}) \tag{14}$$

3.1 A Linear Algebra Example

For clarity, the example of Section 2.1 is repeated here using the matrix formulation. According to Eq. 7 the function *f* is now written as a vector,

$$f_{\vec{x}} = \begin{pmatrix} 1 \\ 1 \\ -1 \\ 1 \end{pmatrix}$$

The matrix ***B*** is

$$\boldsymbol{B} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}$$

and using Eq. 8 the Fourier expansion of *f* is now written

$$\frac{1}{4}\boldsymbol{B}f = \frac{1}{4}\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}\begin{pmatrix} 1 \\ 1 \\ -1 \\ 1 \end{pmatrix} = \frac{1}{2}\begin{pmatrix} 1 \\ -1 \\ 1 \\ 1 \end{pmatrix}$$

Now considering the case where the function is unknown but a training set is available, the approximate function represented by the training set is written as a vector using Eq. 13,

$$\tilde{f}_{\vec{x}} = \begin{pmatrix} 1 \\ 1 \\ -1 \\ 0 \end{pmatrix}$$

and the approximate Fourier transform is obtained from Eq. 14,

$$\frac{1}{3}\boldsymbol{B}\tilde{f} = \frac{1}{3}\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}\begin{pmatrix} 1 \\ 1 \\ -1 \\ 0 \end{pmatrix} = \frac{1}{3}\begin{pmatrix} 1 \\ -1 \\ 3 \\ 1 \end{pmatrix}$$

4 QUANTUM FORMULATION

The final step is to extend the vector representation into the quantum regime. Given a set of *n* (where *n* is the length of the binary input $\vec{x}$ to *f*) qubits whose basis states, $|\vec{x}\rangle$, correspond to all the different values for $\vec{x}$, define

$$|f\rangle = \frac{1}{\sqrt{2^n}} \sum_{\vec{x}\in\{0,1\}^n} c_{\vec{x}}|\vec{x}\rangle \tag{15}$$

where the amplitudes

$$c_{\vec{x}} = f(\vec{x}) = \{-1,1\} \tag{16}$$

as a quantum state of the *n* qubits that describes the function *f*. The domain of *f* is encoded in the state labels and the range of the function in the phases of the amplitudes. Similarly, given a set of *m* examples T, define

$$|\tilde{f}\rangle = \frac{1}{\sqrt{m}} \sum_{\vec{x}\in T} c_{\vec{x}}|\vec{x}\rangle \tag{17}$$

where again the amplitudes are defined as in Eq. 16. This quantum state of the *n* qubits describes the partial function $\tilde{f}$. Next, define the operator $\hat{B}$ as the matrix ***B*** multiplied by the normalization constant $1/\sqrt{2^n}$ and note that it acts on a quantum state, transforming it from the $|\vec{x}\rangle$ basis to the Fourier basis, $\chi_{\vec{a}}$. Now that transformation takes the form

$$\hat{B}|f\rangle = |\hat{f}\rangle \tag{18}$$

or

$$\hat{B}|\tilde{f}\rangle = |\tilde{\hat{f}}\rangle \tag{19}$$

slightly abusing the hat notation by using it to symbolize both an operator and the quantum state vector associated with the Fourier expansion of *f*. Similar to Eq. 11

$$\hat{B}\left|\vec{x}\right\rangle = \left|\chi\right\rangle \tag{20}$$

results in a quantum state that contains in its amplitudes the values of all Fourier basis functions for the input $\vec{x}$. Finally,

$$f(\vec{x}) = \left\langle \hat{B}\vec{x} \,\middle|\, \hat{B}f \right\rangle \tag{21}$$

represents the full function and

$$\tilde{f}(\vec{x}) = \left\langle \hat{B}\vec{x} \,\middle|\, \hat{B}\tilde{f} \right\rangle \tag{22}$$

represents an approximation of the function from T.

4.1 A Quantum Example

Finally, the example of Section 2.1 is repeated using the quantum formulation. According to Eq. 15 the function *f* is now written as a quantum superposition,

$$|f\langle = \frac{1}{2}|00\langle + \frac{1}{2}|01\langle - \frac{1}{2}|10\langle + \frac{1}{2}|11\langle$$

The operator $\hat{B}$ is

$$\hat{B} = \frac{1}{2}\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}$$

and using Eq. 18 the Fourier expansion of *f* is now written

$$\hat{B}\left|f\right\rangle = \frac{1}{2}\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}\frac{1}{2}\begin{pmatrix} 1 \\ 1 \\ -1 \\ 1 \end{pmatrix} = \frac{1}{2}\left|00\right\rangle - \frac{1}{2}\left|01\right\rangle + \frac{1}{2}\left|10\right\rangle + \frac{1}{2}\left|11\right\rangle$$

Now considering the case where the function is unknown but a training set is available, the approximate function represented by the training set is written as a quantum superposition using Eq. 17,

$$\left|f\right\rangle = \frac{1}{\sqrt{3}}\left|00\right\rangle + \frac{1}{\sqrt{3}}\left|01\right\rangle - \frac{1}{\sqrt{3}}\left|10\right\rangle$$

and the approximate Fourier transform is obtained from Eq. 19,

$$\hat{B}\left|\tilde{f}\right\rangle = \frac{1}{2}\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix} \frac{1}{\sqrt{3}} \begin{pmatrix} 1 \\ 1 \\ -1 \\ 0 \end{pmatrix} = \frac{1}{2\sqrt{3}}\left|00\right\rangle - \frac{1}{2\sqrt{3}}\left|01\right\rangle + \frac{3}{2\sqrt{3}}\left|10\right\rangle + \frac{1}{2\sqrt{3}}\left|11\right\rangle$$

Note that because of the unique statistical nature of quantum systems, the amplitudes are not the exact Fourier coefficients now, but are instead proportional to them.

4.2 A Quantum Fourier-based Learning Algorithm

As any quantum algorithm is effected by applying operators to a quantum system, to describe such an algorithm, we must identify and describe the operators necessary to produce the algorithm. Equations 17 and 22 describe a quantum algorithm for Fourier-based inductive learning, given a set of functional examples T and an appropriate set of qubits. The requisite operators are $\hat{T}$, $\hat{B}$, and $\hat{O}$, for preparing the state described in Eq. 17, for performing the Walsh transform, and for observing the system respectively. Since $\hat{B}$ is actually the well-known Hadamard transform used in many quantum algorithms and since observation is achieved by measuring some property of the system (and is also present in all quantum algorithms), we will not discuss them further here. Description of the $\hat{T}$ operator is non-trivial and is treated in detail in [15]. Here it suffices to say that it has been shown that the $\hat{T}$ operator can be implemented with a polynomial (in *m*) number of basic quantum operators all of which operate on one, two or three qubits.

Finally, we assume (as do all proposals for quantum algorithms) that preparation of the quantum system into specific basis states, such as $\left|\vec{0}\right\rangle$ and $\left|\vec{x}\right\rangle$ present no theoretical challenges (though technological obstacles may be significant).

5 GENERALIZATION

A crucial consideration for any learning algorithm is its generalization capability. Here we note that the algorithm as described in Section 4 has, in fact, no generalization capability and perfectly memorizes the training set. To see this, recall from Eq. 22 that

$$\tilde{f}(\vec{x}) = \left\langle \hat{B}\vec{x} \middle| \hat{B}\tilde{f} \right\rangle$$

Noting that operator $\hat{B}$ is unitary and using linear algebra gives

$$\tilde{f}(\vec{x}) = \left\langle \hat{B}\vec{x} \middle| \hat{B}\tilde{f} \right\rangle = \left\langle \vec{x} \middle| \hat{B}^{\dagger}\hat{B} \middle| \tilde{f} \right\rangle = \frac{1}{2^n}\left\langle \vec{x} \middle| \tilde{f} \right\rangle = \begin{cases} \frac{1}{\sqrt{m}} f(\vec{x}) & \text{if } \vec{x} \in T \\ 0 & \text{otherwise} \end{cases} \qquad (23)$$

Note that up to a normalization constant, this results holds for both the Fourier-analytical and linear algebraic descriptions as well if $|A| = 2^n$ (where A is the set of coefficients discussed in Section 2), that is, if all Fourier basis functions are included in the approximation of *f*. To illustrate, we continue the example of Section 4.1 (2.1 and 3.1 are analogous). Calculating the approximate value for 00 or 01 or 10 will demonstrate the non-zero case in Eq. 23; for 00, using Eq. 22 we have

$$\tilde{f}(00) = \left\langle \hat{B}\vec{x} \middle| \hat{B}\tilde{f} \right\rangle = \frac{1}{2}\begin{pmatrix} 1 & 1 & 1 & 1 \end{pmatrix} \frac{1}{2\sqrt{3}} \begin{pmatrix} 1 \\ -1 \\ 3 \\ 1 \end{pmatrix} = \frac{1}{\sqrt{3}}$$

As an example of the zero case, we calculate the approximate value for 11. Again, using Eq. 22

$$\tilde{f}(11) = \left\langle \hat{B}\vec{x} \,\middle|\, \hat{B}\tilde{f} \right\rangle = \frac{1}{2}\begin{pmatrix}1 & -1 & -1 & 1\end{pmatrix}\frac{1}{2\sqrt{3}}\begin{pmatrix}1\\ -1\\ 3\\ 1\end{pmatrix} = 0$$

Recall that *f*: $\{0,1\}^n \rightarrow \{-1,1\}$ and therefore $f(\vec{x})$ can never equal 0. Therefore, approximating *f* by approximating all 2^n Fourier coefficients results in memorization of the training set (up to the sign of the output) and no generalization whatsoever on new inputs.

Generally in the quantum system all basis states are present (corresponding to $|A| = 2^n$ for the non-quantum cases). The result is that equation Eq. 22 is not really *approximating* the Fourier coefficients of the original 2-valued function

f:$\{0,1\}^n \rightarrow \{-1,1\}$ but rather is *computing* the Fourier coefficients of a 3-valued function $\tilde{f}:\{0,1\}^n \rightarrow \{-1,0,1\}$, which has the same large coefficients as *f*, except that they are normalized by an exponential factor.

The end result is that given the proposed algorithm and a perfect quantum system no generalization is possible. However, it is well-known that quantum systems are noisy, easily perturbed and prone to some level of decoherence. If we can assume that such disturbances are random and have a more pronounced affect upon basis states with smaller amplitudes, then the fact that quantum systems are noisy becomes a boon rather than a detriment. This is because due to the nature of the Fourier transform, those basis states with the largest amplitudes are most correlated with the function *f*. Therefore, small random perturbations of the system serve to break the system symmetry around novel functional values, and the basis states most robust with respect to the perturbations, being highly correlated with *f*, provide a large probability of reasonable generalization on those novel inputs.

6 CONCLUSION

Classical Fourier-based learning methods can be extended into the quantum domain and one such possibility has been presented. While classical Fourier-based approaches are limited by their dependence on a membership oracle, the proposed algorithm works using only an example oracle, providing potentially significant advantage over its classical counterparts. Classical machine learning approaches that employ Fourier-analytical methods achieve an ability to generalize on novel data at least in part due to the fact that not all the Fourier coefficients are used in the approximation. However, in the general case for the quantum algorithm, all of the Fourier coefficients are used in the approximation (and indeed, this cannot be avoided). It is therefore proposed that the noise inherent in quantum systems may actually facilitate generalization for the quantum Fourier-based algorithm presented here. In order to investigate this possibility, ongoing work includes simulation of the algorithm on classical computers and analysis qualifying and quantifying system noise conducive to useful generalization.

7 REFERENCES

1. Shor, P., *SIAM Journal of Computing*, **26**, 1484-1509 (1997).
2. Grover, L., "A Fast Quantum Mechanical Algorithm for Database Search", in *Proceedings of the 28th Annual ACM Symposium on the Theory of Computing*, ACM, New York, 1996, pp. 212-19.
3. Hogg, T., *Journal of Artificial Intelligence Research*, **4**, 91-128 (1996).
4. Terhal, B.M. and Smolin, J.A., *Physical Review A* , **58**, 1822-1826 (1998).
5. Powell, M.J.D., "Radial Basis Functions for Multivariable Interpolation: A Review", in *Algorithms for Approximation*, edited by J. Mason and M. Cox, Clarendon Press, Oxford, 1987, pp. 143-167.
6. Broomhead, D.S. and Lowe, D., *Complex Systems*, **2**, 321-355 (1988).
7. Moody, J.E. and Darken, C.J., *Neural Computation*, **1**, 281-294 (1989).
8. Poggio, T. and Girosi, F., *Proceedings of the IEEE*, **78**, 1481-1497 (1990).

9. Burges, C.J.C., *Data Mining and Knowledge Discovery*, **2**, 121-167 (1998).
10. Morrison, N., *Introduction to Fourier Analysis*, Wiley, New York, 1994.
11. Kushilevitz, E. and Mansour, Y., *SIAM Journal of Computing*, **22**, 1331-48 (1993).
12. Blum, A., Furst, M., Jackson, J., Kearns, M., Mansour, Y. and Rudich, S., "Weakly Learning DNF and Characterizing Statistical Query Learning Using Fourier Analysis", in *Proceedings of the 26th Annual ACM Symposium on Theory of Computing*, ACM Press, 1994, pp. 253-62.
13. Bshouty, N.H. and Jackson, J., "Learning DNF Over the Uniform Distribution Using A Quantum Example Oracle", in *Proceedings of the 8th Annual Conference on Computational Learning Theory (COLT'95)*, Santa Cruz, California, 1995, pp. 118-127.
14. Shapire, R.E., *Machine Learning*, **5**, 197-227 (1990).
15. Ventura, D. and Martinez, T., *Foundations of Physics Letters*, **12**, 547-559(1999).