A Sequential Algorithm for Feed-forward Neural Networks with Optimal Coefficients and Interacting Frequencies

Enrique Romero and René Alquézar
Departament de Llenguatges i Sistemes Informàtics
Universitat Politècnica de Catalunya
Barcelona, Spain
{eromero,alquezar}@lsi.upc.edu

Abstract

An algorithm for *Sequential Approximation with Optimal Coefficients and Interacting Frequencies* (*SAOCIF*) for Feed-forward Neural Networks is presented. *SAOCIF* combines two key ideas. The first one is the optimization of the coefficients (the linear part of the approximation). The second one is the strategy to choose the frequencies (the non-linear weights), taking into account the interactions with the previously selected ones. The resulting method combines the locality of sequential approximations, where only one frequency is found at every step, with the globality of non-sequential methods, where every frequency interacts with the others. The idea behind *SAOCIF* can be theoretically extended to general Hilbert spaces. Experimental results show a very satisfactory performance.

1 Introduction

In terms of the Bias/Variance decomposition, as far as the number of hidden units of a Feed-forward Neural Network (FNN) grows, bias decreases and variance increases. This happens because the flexibility of the model also grows with the number of hidden units [Geman et al. 1992; Barron 1994]. Therefore, choosing an adequate architecture is a very important issue in order to obtain good generalization performance. We will focus on FNNs with one hidden layer of units (two layers of weights), including both Multi-Layer Perceptrons (MLPs) and Radial Basis Function Networks (RBFNs).

Sequential approximation algorithms for FNNs (also named incremental or constructive) allow to dynamically construct the network without setting *a priori* the number of hidden units. They can help to find a proper trade-off between bias and variance by controlling, among other things, the number of hidden units. These methods start with a small network (usually with no hidden units),

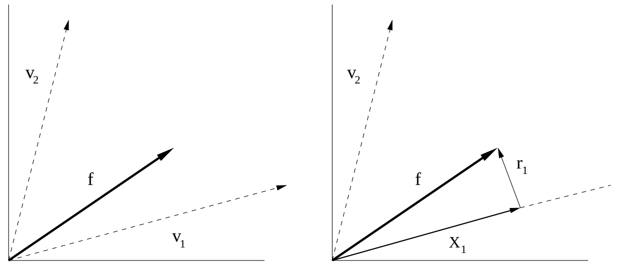

Figure 1: Sequence of the approximation of a vector f in $\mathbb{R}^2$ with v_1 and v_2 matching the previous residue without recalculating the coefficients. In the first step (left-middle), X_1 is obtained. In the second step (middle-right), r_1 is approximated with v_2. The resulting vector (X_2) is not the best approximation that can be achieved with v_1 and v_2. In this case, optimizing the coefficients allows to obtain f.

and sequentially add new terms (that is, hidden units, each associated with a frequency) until a satisfactory solution is found. We will refer to the weights in the first layer (the non-linear weights) as *frequencies* and to the weights in the second layer (the linear weights) as *coefficients*.

Most of the sequential models found in the literature keep the previously selected frequencies fixed. Among these sequential models, many of them choose the new term so that it matches the previous residue as best as possible (see section 2). It is well known that, although this strategy leads to approximations convergent towards the target function, it may be far from being the best strategy. This fact can be observed in the example in figure 1: when approximating the vector f with v_1 and v_2 we obtain X_2; clearly, this is not the best possible approximation, since v_1 and v_2 form a basis of $\mathbb{R}^2$. In this case, optimizing the coefficients of the previously added terms would lead to a much better approximation (exact, in fact) of the target vector. But recalculating the coefficients is not enough, as illustrated in the example in figure 2. Suppose that X_1 is a partial approximation of f, and h is the vector which best matches the residue r_1. Since h does not lie on the plane that contains X_1 and f, it is not necessarily the vector that, together with X_1, best approximates the target vector f. Any vector lying on the plane that contains X_1 and f (g, for example) allows an exact approximation of f. Regardless of the coefficients optimization, matching the residue does not take into account the interactions with the previously selected terms. Any vector lying on the plane that contains f and a vector of the subspace spanned by the previous terms allows an exact approximation of the target vector. The vector that best matches the residue does not necessarily satisfy this property.

Important exceptions to the idea of matching the residue are the Orthogonal Least Squares Learning (OLSL) algorithm [Chen et al. 1991] and the Kernel

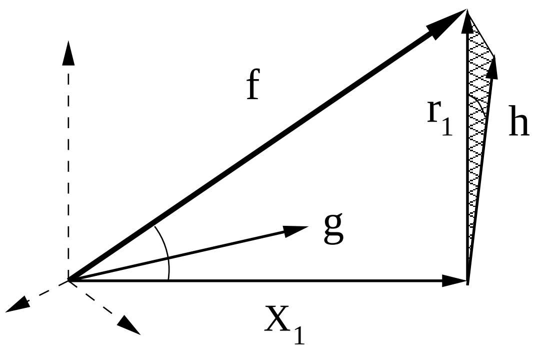

Figure 2: Approximation of a vector f in $\mathbb{R}^3$ matching the previous residue and recalculating the coefficients. Suppose that h is the vector that best matches the residue r_1. The vector g, which lies on the plane that contains f and X_1, allows an exact approximation to f when combined with X_1. The vector h (not on this plane) does not have this property. Optimizing the coefficients is not enough if h is selected.

Matching Pursuit (KMP) with *pre-fitting* algorithm [Vincent and Bengio 2002], where an (implicit or explicit) orthogonalization procedure is performed. In these algorithms, every point in the data set is considered as a candidate frequency. After calculating the optimal coefficients for every possible network, the best one (in terms of the minimum squared error) is selected.

In this work we present an algorithm for *Sequential Approximation with Optimal Coefficients and Interacting Frequencies* (*SAOCIF*) for FNNs, which combines two key ideas. On the one hand, it optimizes the coefficients, so that the best approximation with the selected vectors is always achieved, as in figure 1. On the other, the frequencies are selected at every step taking into account the interactions with the previously selected terms in a flexible manner. The interactions are discovered by means of their optimal coefficients. In the example in figure 2, *SAOCIF* would select g (instead of h), because it allows a better approximation of f when combined (interacts) with X_1. The contribution of the new frequency is measured in terms of its capability of approximation to the target vector together with the previously selected frequencies. There is no explicit intention to match the residue. That is the idea of *interacting frequencies*. Therefore, it may be able to obtain, with the same number of hidden units, better approximations than matching the residue as best as possible. In terms of the Bias/Variance decomposition, it will be possible to obtain simpler models with the same bias, since the same level of approximation may be achieved with less hidden units.

The proposed algorithm can be seen as an extension and generalization of the OLSL and KMP with *pre-fitting* algorithms in several ways. First, it is not restricted to select the candidate frequencies from the points in the data set. In this sense, a number of different heuristics can be used to select the frequencies. Second, it allows to choose the activation function for every hidden unit. Finally, it is possible to further tune the selected frequencies with any non-linear optimization technique.

The idea behind *SAOCIF* can be extended to approximation in Hilbert spaces, maintaining orthogonal-like properties. The theoretical results obtained prove that, under reasonable conditions, the residue of the approximation is (in the limit) the best one that can be obtained with any subset of the given set of vectors. The importance of the interacting frequencies lies in the hypothesis that, as it can be seen in figure 2, it seems more plausible to find better partial approximations selecting the new frequency taking into account the interactions with the previous frequencies than just matching the residue as best as possible. Therefore, the rate of convergence is expected to increase.

Several experiments were performed in order to test the algorithm. Experimental results show a very satisfactory performance when compared to other sequential approaches. In particular, *SAOCIF* works better than methods that select the new frequencies based on the idea of matching the residue, confirming the suitability of the interacting frequencies approach. When the number of hidden units is fixed *a priori*, better performance is obtained (with a moderate increase in the execution times). When the desired approximation accuracy is fixed *a priori*, *SAOCIF* allows to obtain models with less hidden units. These results are in agreement with the Bias/Variance decomposition.

The rest of the paper is organized as follows. An overview of some sequential approximations can be found in section 2. The definition of *SAOCIF* and the algorithm are presented in section 3. The extension to general Hilbert spaces is discussed in section 4. The experimental results are described in section 5. Finally, section 6 concludes and outlines some directions for further research.

2 Sequential Approximations

In this section, several sequential approximations for regression are described. As it will be seen, matching the residue is the underlying idea for most of the proposed schemes. Although some of them optimize the coefficients, the frequencies are selected without taking into account the interactions with the previously selected terms. Important exceptions are the OLSL and the KMP with *pre-fitting* algorithms, where an (implicit or explicit) orthogonalization is performed. An extensive review on sequential methods for regression can be found in [Kwok and Yeung 1997a].

2.1 Dynamic Node Creation

The Dynamic Node Creation method [Ash 1989] is a sequential method where, during the training, a new hidden node is added when the rate of decrease of the average squared error is less than a certain value. After a new node is added, the whole network is trained with standard Back-Propagation until the solution is satisfactory or another node is needed. Several variants of the Dynamic Node Creation procedure can be found in the literature [Bello 1992; Bartlett 1994; Wang et al. 1994; Shin and Ghosh 1995].

2.2 Resource-Allocating Network

The Resource-Allocating Network is presented in [Platt 1991]. When the network performs well on a presented pattern, the whole network is trained. Otherwise, a new Gaussian RBF hidden unit is added. Therefore, a memorization of training patterns is performed in some sense. The training of the whole network is performed with gradient descent. Several variations of this model can be found in [Kadirkamanathan and Niranjan 1993; Yingwei et al. 1997; Salmerón et al. 2001]. All of them are specific for RBFNs.

2.3 Projection Pursuit

Projection Pursuit is a family of optimization methods which can be found in the Statistics literature [Huber 1985]. Projection Pursuit Regression (PPR) [Friedman and Stuetzle 1981], as a particular case of function approximation, estimates the conditional expectation of a random variable $Y \in \mathbb{R}$ given $X \in \mathbb{R}^I$ by means of a sum of ridge functions $\sum_{j=1}^{N} g_j(a_j \cdot x)$ as follows (the a_j's act as the frequencies). Suppose that the first $N-1$ terms of the approximation have been determined. That is, the vectors a_j and the functions g_j $(1 \leqslant j \leqslant N-1)$ have been calculated. Let $r_{N-1}(x)$ be the residue at step $N-1$. Find a_N and g_N such that $\|r_{N-1}(x) - g_N(a_N \cdot x)\|$ is the minimum. This process is repeated until the residue is smaller than a user-defined threshold. This procedure may be improved by *back-fitting*: omit some of the earlier summands g_j, determine better replacements, and then iterate. In [Jones 1987] it is proved that, under mild smoothness conditions, a minimizing direction a_N exists at every step such that $\lim_{N\to\infty} E[r_N] = 0$. Later, it was proved that the convergence may be accelerated approximating by an optimal convex combination [Jones 1992]. The upper bounds for the rate of convergence of approximations with FNNs derived in [Barron 1993] are based on this result.

2.3.1 Projection Pursuit in Signal Processing

Some methods with the same underlying ideas as Projection Pursuit can be found in the Signal Processing area. In [Mallat and Zhang 1993] Matching Pursuit (MP) is described, an algorithm that decomposes any signal into a linear expansion of waveforms that are selected from a (possibly redundant) dictionary of functions F, with $\|g_\gamma\| = 1$ for every $g_\gamma \in F$. Similar to PPR, the MP algorithm works choosing at step N an element $g_{\gamma_N} \in F$ which closely matches the residue $R^N f$, that is $\left|\left\langle R^N f, g_{\gamma_N}\right\rangle\right| \geqslant \alpha \sup_{g_\gamma \in F} \left|\left\langle R^N f, g_\gamma\right\rangle\right|$, where $0 < \alpha \leqslant 1$. The convergence property of MP is proved in [Mallat and Zhang 1993], based on the results in [Jones 1987]. After N steps, a recalculation of the coefficients can be made, named *back-projection*, to approximate f at best with the selected vectors. This idea was formalized in the Orthogonal MP algorithm [Pati et al. 1993]. Similar results are obtained in [Qian and Chen 1994] with a particular set of functions: the normalized Gaussian functions with adjustable variance and time-frequency center.

2.3.2 Projection Pursuit with Neural Networks

The two layer architecture of a neural network is well suited to construct an approximation with PPR. The Projection Pursuit Learning Network (PPLN) [Hwang et al. 1994] is modeled as a one hidden layer MLP that learns unit by unit, and layer by layer cyclically after all the training patterns are presented. Weights are learnt while the other ones remain fixed, and no global optimization of the coefficients is performed.

The Incremental Linear Quasiparallel (ILQ) algorithm, a sequential algorithm for neural networks based on the ideas of PPR and MP, is presented in [Kůrková and Beliczyński 1995]. Every iteration consists of two steps. In the first one, the frequency of the new hidden unit is determined. In the second one, all output weights are recalculated. Given a the set of functions G, the frequencies are determined trying to find $g_N \in G$ so that nearly maximizes $|\langle f - f_{N-1}, g\rangle|$ over $g \in G$. Therefore, the resulting method is similar to Orthogonal MP, where there exists a *back-projection* procedure at every iteration. Based on the results in [Jones 1992], upper bounds for the rate of convergence are derived in [Kůrková and Beliczyński 1995].

With similar ideas, a number of objective functions to obtain the new frequency are explored in [Kwok and Yeung 1997b], all of them with the aim of matching the residue.

2.4 Cascade-Correlation

In the Neural Networks literature, the most used constructive method is, probably, Cascade-Correlation (CASCOR) [Fahlman and Lebiere 1990]. CASCOR combines two key ideas. The first one is the cascade architecture, in which the newly added hidden unit receives inputs from the input layer as well as from the previously added hidden neurons. The second one is the learning algorithm. For each new hidden unit, the algorithm tries to maximize the correlation (or, more precisely, the covariance) between the output of the new unit and the residual error signal of the network. In order to maximize this function, a gradient ascent is performed. The input weights of the hidden units are frozen at the time the new unit is added to the network, and only the output connections are trained. There exist variations of CASCOR where the architecture is not cascaded, and where the learning rule is changed to train directly for minimization of the output errors (see [Prechelt 1997] for details). Anyway, the frequencies obtained by these methods are the result of matching the residue at the previous step with only one term. There exist many variants of the original CASCOR algorithm in the literature (see, for example, [Littmann and Ritter 1992; Lahnajärvi et al. 1999; Lehtokangas 1999; Treadgold and Gedeon 1999; Islam and Murase 2001; Ma and Khorasani 2004]). A comparison of some of them can be found in [Lahnajärvi et al. 2002].

2.5 Orthogonal Sequential Neural Networks Methods

The Orthogonal Least Squares Learning (OLSL) algorithm is proposed in [Chen et al. 1991], a learning procedure for RBFNs based on the Orthogonal Least Squares method [Chen et al. 1989]. The procedure starts with a single Gaussian RBF hidden unit and it sequentially increases the number of hidden units, one at a time, until the model error is satisfactory. The frequency of the new hidden unit (the center) is selected among the points in the data set. The classical Gram-Schmidt orthogonalization method is used at each step to form a set of orthogonal vectors for the space spanned by the output vectors of the previously selected hidden units. For every point in the data set, the orthogonal component of its output vector to that space is obtained. The new information introduced by this unit is caused by that part of its output vector which is orthogonal to the space spanned by the output vectors of previously selected hidden units. In this context, an output vector is an element of $\mathbb{R}^T$, where T is the number of patterns, obtained by applying the Gaussian function to every point in the data set. After computing its optimal coefficient (with the squared error function), the point in the data set maximizing the error reduction ratio is selected. The procedure is terminated when a predetermined percentage of the total error is reduced. Some extensions of the original procedure can be found in [Chen et al. 1996, 1999].

Recently, the Kernel Matching Pursuit (KMP) algorithm was described [Vincent and Bengio 2002], an extension of MP that can be used to build kernel-based solutions to supervised Machine Learning problems. The emphasis of the KMP scheme is put on the building of an alternative to Support Vector Machines (SVMs) [Vapnik 1995] that controls the sparsity of the solution (that is, the number of support vectors). Whereas good generalization abilities of SVMs are related to margin maximization, KMP is designed to build sparse kernel-based solutions minimizing the squared error function. As in SVMs, the frequencies of the resulting network are a subset of the points in the data set. Given a data set D, the dictionary (needed in MP) is defined as the set of functions $F = \{K(x, x_i) : x_i \in D\}$, where K is a symmetric positive definite kernel function. Three versions of KMP are defined in [Vincent and Bengio 2002]: basic KMP (similar to basic MP), KMP with *back-fitting* at every step (similar to Orthogonal MP) and KMP with *pre-fitting* (similar to OLSL). The optimization problems posed can be solved exactly because a finite dictionary is used. Experimental comparisons between KMP with *pre-fitting* and SVMs for several classification problems show comparable results with typically much sparser models for KMP with *pre-fitting*. A very similar method, but particular for Gaussian processes, can be found in [Smola and Bartlett 2001]. The main difference lies in the loss function to be optimized.

3 Definition of *SAOCIF* and Algorithm

As previously mentionedd, most of the sequential models for FNNs found in the literature choose the new term so that it matches the previous residue as best as possible. This strategy can be far from being the best one, since it does not take into account the interactions with the previously selected terms. In this section *SAOCIF* is presented, a sequential scheme for FNNs where the contribution of the new frequency is measured in terms of its capability of approximation to the target vector together with the previously selected ones.

3.1 Definition

Definition. Let H be the Hilbert space $\mathbb{R}^T$, where T is the number of patterns in a data set $D = \{x_1, \cdots, x_T\}$, $f = (f_1, \cdots, f_T) \in H$ the target vector and Ω a space of frequencies. A *Sequential Approximation with Optimal Coefficients and Interacting Frequencies* (*SAOCIF*) for FNNs is a sequence of vectors $\{X_N\}_{N \geqslant 0}$ in H whose terms are defined as:

1. $X_0 = 0$.
2. $X_N = \sum_{k=1}^{N-1} \lambda_k^N v_{\omega_k} + \lambda_N^N v_{\omega_N}$, so that
   (a) The coefficients $\lambda_1^N, \cdots, \lambda_{N-1}^N, \lambda_N^N$ are optimal. That is, the vector X_N is the best approximation of f with vectors $v_{\omega_1}, \cdots, v_{\omega_{N-1}}, v_{\omega_N} \in H$ in terms of minimizing the squared error $\|f - X_N\|^2$ (that is, with the metric induced by the usual inner product in $\mathbb{R}^T$).
   (b) The frequency $\omega_N \in \Omega$ is selected taking into account the interactions of v_{ω_N} with $v_{\omega_1}, \cdots, v_{\omega_{N-1}}$ in order to minimize $\|f - X_N\|^2$.

Remarks.

1. In FNNs terminology, every frequency $\omega_k \in \Omega$ is associated with a hidden unit $\varphi_k(\omega_k, x)$, where φ_k is the activation function. The i-th component of v_{ω_k} is the value of the hidden unit $\varphi_k(\omega_k, x)$ at the i-th point in D. That is, $v_{\omega_k} = (\varphi_k(\omega_k, x_1), \cdots, \varphi_k(\omega_k, x_T))$. The output function of the network with N hidden units is $X_N(x) = \sum_{k=1}^{N} \lambda_k^N \varphi_k(\omega_k, x)$.
2. At step N, a new frequency (ω_N) is considered, the number of terms of the approximation is increased by one $(\lambda_N^N v_{\omega_N})$, and the coefficients $\lambda_1^N, \cdots, \lambda_{N-1}^N$ are optimized in order to obtain the best approximation of f with vectors $v_{\omega_1}, \cdots, v_{\omega_{N-1}}, v_{\omega_N}$. The frequencies $\omega_1, \cdots, \omega_{N-1}$ are kept fixed. The vectors $v_{\omega_1}, \cdots, v_{\omega_{N-1}}, v_{\omega_N}$ are not necessarily mutually orthogonal.
3. As it is well known [Achieser 1956], since X_N is the best approximation of f with $v_{\omega_1}, \cdots, v_{\omega_{N-1}}, v_{\omega_N}$, it holds that

$$\forall k : 1 \leqslant k \leqslant N \quad \langle f - X_N, v_{\omega_k} \rangle = 0, \tag{1}$$

where $\langle \cdot, \cdot \rangle$ is the inner product in H. That is, $f - X_N$ is orthogonal to the space spanned by $v_{\omega_1}, \cdots, v_{\omega_{N-1}}, v_{\omega_N}$. By definition of inner product, (1) is equivalent to the following linear equations system:

$$A_N \cdot \left(\lambda_1^N, \cdots, \lambda_{N-1}^N, \lambda_N^N\right)^t = \left(\langle f, v_{\omega_1} \rangle, \cdots, \langle f, v_{\omega_{N-1}} \rangle, \langle f, v_{\omega_N} \rangle\right)^t \quad (2)$$

where $A_N[i,j] = \langle v_{\omega_i}, v_{\omega_j} \rangle$ for $1 \leqslant i,j \leqslant N$. Thus, once the frequencies $\omega_1, \cdots, \omega_{N-1}, \omega_N \in \Omega$ have been selected, the optimal coefficients $\lambda_1^N, \cdots, \lambda_{N-1}^N, \lambda_N^N$ can be computed by solving (2).

4. Using (1), it is immediate to verify that

$$\|f - X_N\|^2 = \left\| f - \sum_{k=1, k \neq j}^{N} \lambda_k^N v_{\omega_k} \right\|^2 - \left|\lambda_j^N\right|^2 \left\|v_{\omega_j}\right\|^2 \quad (3)$$

$$\|f - X_N\|^2 = \|f\|^2 - \|X_N\|^2 \quad (4)$$

$$\|X_N\|^2 = \sum_{k=1}^{N} \lambda_k^N \langle f, v_{\omega_k} \rangle \quad (5)$$

As it can be observed, there is a great parallelism between these properties and those satisfied by an approximation with orthogonal vectors.

3.2 Algorithm

A sequential training algorithm for FNNs following the ideas of *SAOCIF* definition is presented in figure 3. Hidden units are added one at a time, choosing the frequencies in a flexible manner, so as to adjust the network until a satisfactory model is obtained. The algorithm works as follows. Suppose that we are at step N and we have a certain procedure to generate frequencies. For every candidate frequency, the optimal coefficients of the network are computed with that frequency installed in the new hidden unit, in order to test the real contribution of the frequency (together with the $N-1$ previously selected ones) to the approximation to the target vector. There is no explicit intention to match the residue. That is the idea of interacting frequencies. Note that, according to (4), maximizing $\|X_N\|^2$ is equivalent to minimizing $\|f - X_N\|^2$. When the frequency is satisfactory or there are no more candidate frequencies (criterion 1, see below), the selected frequency can be optionally tuned.

Concerning the architecture needed to construct the approximation, it must have the following characteristics:

1. It must be a feed-forward architecture with a hidden layer of units (including both MLPs and RBFNs).

Algorithm
  repeat
    Increase by 1 the number of hidden units N
    Pick an activation function for the new hidden unit
    repeat
      Assign a candidate frequency ω to the new hidden unit
      Compute the optimal coefficients $\{\lambda_k\}_{1 \leqslant k \leqslant N}$ by solving (2)
      Set $\omega_N := \omega$ if $\|f - X_N\|^2$ is minimized
    until the frequency ω_N is satisfactory or
        there are no more candidate frequencies (criterion 1)
    Optionally, tune the selected frequency ω_N
    Fix the frequency ω_N in the network
  until the network is satisfactory (criterion 2)
end Algorithm

Figure 3: An algorithm to construct an FNN following the ideas of *SAOCIF*.

2. There are no restrictions about the dimension of the input and the output. There will be as many as the target function have. If there are several outputs, the total inner products must be calculated as the summation of the individual inner products for every output.

3. There is no restriction about the biases in the hidden units, since they can be treated as part of the frequencies.

4. There is no restriction about the activation functions in the hidden units. They can be, for example, sigmoidal, Gaussian, sines, cosines, wavelets, etc. Recent works have shown that the use of non-sigmoidal activation functions for MLPs may lead to very promising results [Sopena et al. 1999]. Obviously, different units may have different activation functions. The output units must have a linear activation function.

As it can be seen, the only real restriction in the architecture is the linear activation function in the output units.

The resulting algorithm combines the locality of sequential approximations, where only one frequency is found at every step, with the globality of non-sequential methods, such as Back-Propagation [Rumelhart et al. 1986], where every frequency interacts with the others. The interactions are discovered by means of the optimal coefficients. The importance of the interacting frequencies lies in the hypothesis that they allow to find better partial approximations, with the same number of hidden units, than frequencies selected just to match the residue as best as possible. In terms of the Bias/Variance decomposition, it will be possible to obtain simpler models with the same bias, since the same level of approximation may be achieved with less hidden units.

3.3 Implementation

The strategy to select the candidate frequency is probably the most important part of the algorithm. In our experiments (see section 5) three strategies were introduced in order to test the algorithm:

1. Random strategy: The frequencies are randomly selected within a certain range.

2. Input strategy: The frequencies are selected from the points in the data set (as is often the case in RBFNs) in a deterministic manner: for every hidden unit to be added, every point in the training set is tested as a candidate frequency.

3. BGA strategy: The frequencies are selected with a more sophisticated strategy from the field of Evolutionary Algorithms, where a population of frequencies evolves driven by a Breeder Genetic Algorithm (BGA) [Mühlenbein and Schlierkamp-Voosen 1993] with the squared error as the fitness function.

The number of candidate frequencies of every strategy may be very different. Whereas for the Input one it is fixed (given the data set), the Random and BGA strategies can be parameterized so as to assign as many candidate frequencies as desired. This is clearly related to the computational cost, as explained below.

Regarding the criterion 1 in figure 3, every strategy has its own one. For the Random strategy, a fixed number of frequencies is selected. For the Input one, the number of points in the data set determines the number of frequencies to test. For the BGA strategy, a maximum number of generations is determined. Regarding the criterion 2 in figure 3, many stopping criteria can be used: percentage of learnt patterns, early stopping with a validation set, low relative rate of decrease of the error, etc.

SAOCIF satisfies a number of interesting properties to implement it in an efficient fashion, as explained next.

By (4) we have $\|f - X_N\|^2 = \|f\|^2 - \|X_N\|^2$, with $\|f\|^2$ constant. Therefore, the frequency that minimizes the error is such that maximizes $\|X_N\|^2$. By (5) we know that $\|X_N\|^2 = \sum_{k=1}^{N} \lambda_k^N \langle f, v_{\omega_k} \rangle$. The values of $\{\langle f, v_{\omega_k} \rangle\}_{1 \leqslant k \leqslant N}$ are the independent vector of the linear equations system (2) just solved to obtain $\{\lambda_k^N\}_{1 \leqslant k \leqslant N}$, which can be kept stored in memory. Therefore, once the coefficients have been obtained, $\|X_N\|^2$ can be computed with cost $O(N)$, avoiding one pass through the data set (the cost of directly computing $\|X_N\|^2$ or $\|f - X_N\|^2$ is $O(T \cdot N)$, where T is the number of examples). In our implementation, ω_N was selected so as to maximize $\|X_N\|^2$, computed with (5).

Since the frequencies $\omega_1, \cdots, \omega_{N-1}$ are kept fixed, the proposed linear equations system at step N is equal to the system solved at step $N-1$ for the selected frequency, but with a new row and a new column. Therefore, the system (2) posed at step N can be efficiently solved with bordered systems techniques

[Faddeeva 1959] as follows. Note that (2) can be stated as

$$A_N \cdot (\lambda_1^N, \cdots, \lambda_N^N)^t = \left(\begin{array}{c|c} A_{N-1} & a \\ \hline a^t & \gamma \end{array}\right) \cdot \left(\begin{array}{c} x \\ \hline \eta \end{array}\right) = \left(\begin{array}{c} b \\ \hline \beta \end{array}\right) = (\langle f, v_{\omega_1} \rangle, \cdots, \langle f, v_{\omega_N} \rangle)^t,$$

where x, a, b are vectors with $N-1$ components, and η, β, γ are scalars. It can be easily verified that

$$\eta = \frac{\beta - a^t \cdot A_{N-1}^{-1} \cdot b}{\gamma - a^t \cdot A_{N-1}^{-1} \cdot a} \quad \text{and} \quad x = A_{N-1}^{-1} \cdot (b - \eta \cdot a). \tag{6}$$

Since A_{N-1} and b are equal for every candidate frequency, matrix A_{N-1} was inverted only once at every step, prior to the selection of the first frequency, and kept stored in memory. For every candidate frequency, its associated linear equations system was solved using (6). Whereas the inversion of a matrix has cost $O(N^3)$, the computational cost of multiplying it by a vector is $O(N^2)$, where N is the matrix dimension (in our case, the number of hidden units).

Regarding the computational cost, let N_{SAOCIF} be the number of hidden units of the resulting network, F the number of candidate frequencies for every added hidden unit (assuming that it is equal for every one), T the number of patterns and I the input dimension. Using the aforementioned implementation properties, and assuming $N_{SAOCIF} \leqslant F$ and $N_{SAOCIF} \leqslant T$, the computational cost of the algorithm in figure 3 can be bounded by $C_1 \cdot N_{SAOCIF}^2 \cdot F \cdot h_1(T \cdot I)$, where C_1 is constant and $h_1(T \cdot I) \in O(T \cdot I)$. The conditions $N_{SAOCIF} \leqslant T$ and $N_{SAOCIF} \leqslant F$ are not real restrictions, since it makes no sense, for example, constructing a network with more hidden units than examples in the data set. Similarly, the number of candidate frequencies is usually larger than the number of hidden units. In addition, these conditions make that the quadratic factor N_{SAOCIF}^2 is not so relevant, in practice, as one could think (see section 5). The computational cost mainly comes from the construction of the linear equations system for every candidate frequency rather than from solving it.

3.4 Comparison with Other Sequential Schemes

The algorithm in figure 3 can be seen as an extension and generalization of the OLSL and KMP with *pre-fitting* algorithms (see section 2) in several ways. First, it is not restricted to select the candidate frequencies from the points in the data set. In this sense, a number of different heuristics can be used. Second, it allows to choose the activation function for every hidden unit. Finally, it is possible to further tune the selected frequency with any non-linear optimization technique. Note that the OLSL and KMP with *pre-fitting* algorithms can be simulated with the proposed algorithm by selecting the frequency with the Input strategy without tuning.

4 Extension to Hilbert Spaces

In this section, *SAOCIF* is extended to general Hilbert spaces. Consider the following approximation problem: "Let H be a Hilbert space with inner product $\langle , \rangle : H \times H \to \mathbb{C}$, a space of parameters Ω, and $f \in H$ a vector to approximate with vectors $v_\omega = v(\omega)$, $v : \Omega \to H$, $\omega \in \Omega$, such that $\forall \omega \in \Omega \quad \|v_\omega\| \neq 0$. Find $\omega_1, \omega_2, \cdots \in \Omega$ and $\lambda_1, \lambda_2, \cdots \in \mathbb{C}$ such that $\lim_{N\to\infty} \left\| f - \sum_{k=1}^{N} \lambda_k v_{\omega_k} \right\| = 0$". This definition is, in essence, the traditional one in approximation of vectors in Hilbert spaces [Achieser 1956]. The condition $\|v_\omega\| \neq 0$ is equivalent, by the inner product properties, to that of $v_\omega \neq 0$. Observe that every vector $v_\omega \in H$ depends on a parameter $\omega \in \Omega$.

In this setting, the same definition of *SAOCIF* stated in section 3.1 is also valid for a general Hilbert space H, with the metric induced by its inner product. As previously, the term *frequency* refers to every $\omega_1, \omega_2, \cdots \in \Omega$, and *coefficient* to every $\lambda_1, \lambda_2, \cdots \in \mathbb{C}$. The following results show that *SAOCIF* converges towards the target vector in this general setting under reasonable conditions.

Proposition 1. Let H be a Hilbert space, and $f \in H$. Any *SAOCIF* $\{X_N\}_{N\geqslant 0}$ satisfies the following properties:

(Pa) $\forall N \geqslant 0 \quad \|f - X_{N+1}\|^2 \leqslant \|f - X_N\|^2$.

(Pb) If $M \geqslant N$, then $\langle f - X_M, f - X_N \rangle = \|f - X_M\|^2$.

(Pc) Suppose that $\lambda_N^N \neq 0$. The vector v_{ω_N} is orthogonal to the space spanned by $\{v_{\omega_1}, \cdots, v_{\omega_{N-1}}\}$ if and only if the coefficients do not change between steps $N-1$ and N.

(Pd) $\{X_N\}_{N\geqslant 0}$ is convergent in H. That is, $\exists g \in H \quad \lim_{N\to\infty} \|g - X_N\| = 0$.

The proof can be found in Appendix A.

Observe that by (Pc), the only directions that guarantee that, without recalculating the coefficients, the approximation is optimal are the orthogonal directions. Therefore, if vectors are not mutually orthogonal, coefficients must be recalculated.

Theorem 1. Let H be a Hilbert space, $f \in H$ and a *SAOCIF* $\{X_N\}_{N\geqslant 0}$. Let g be such that $\lim_{N\to\infty} \|g - X_N\| = 0$ (Pd), and suppose that for every $\mu \in \mathbb{C}$ and every $\omega_0 \in \Omega$ we have

$$\|f - X_{N+1}\|^2 \leqslant \|f - (X_N + \mu v_{\omega_0})\|^2 + \alpha_N \tag{7}$$

for every $N \geqslant 0$. That is, the approximation of f with X_{N+1} is better (up to α_N) than the best approximation of the residue $f - X_N$ that one could achieve with only one vector $v_{\omega_0} \in v(\Omega)$. If $\limsup_{N\to\infty} \alpha_N \leqslant 0$, then:

(Ta) The vector g satisfies:

(Ta1) $\forall \omega_0 \in \Omega \quad \lim_{N\to\infty} \langle g - X_N, v_{\omega_0} \rangle = 0$.

(Ta2) $\forall \omega_0 \in \Omega \quad \langle f, v_{\omega_0} \rangle = \langle g, v_{\omega_0} \rangle$.

(Ta3) $\langle f - g, g \rangle = 0$.

(Ta4) There is no subset of vectors in $v(\Omega)$ that approximate f more than g. That is,

$$\|f - g\| = \inf_{\mu_k \in \mathbb{C},\ \psi_k \in \Omega} \left\| f - \sum_k \mu_k v_{\psi_k} \right\|.$$

(Tb) If there exists $A \subseteq \Omega$ such that the set of vectors $\{v_\psi : \psi \in A\}$ spans H, then $\{X_N\}_{N \geqslant 0}$ converges towards f.

The proof can be found in Appendix B.

Observe that these results are not very restrictive, since the universal approximation capability is a necessary condition for the convergence property. Hence, *SAOCIF* allows (by selecting Ω and $v(\Omega)$) to choose any (or some) of the multiple vector families satisfying this property. In the particular case of L^2, it can be used with approximations by algebraic polynomials, Fourier series, wavelets and FNNs (including both MLPs and RBFNs), among others.

The hypothesis about the tolerance α_N is, in essence, the same as in [Jones 1992; Barron 1993; Kůrková and Beliczyński 1995; Kůrková 1998]. Imposing conditions on α_N, as in [Barron 1993; Kůrková and Beliczyński 1995], the rate of convergence can be proved to be $O(1/\sqrt{N})$, which has the same order as the optimal ones that can be found in the literature. However, the actual rate of convergence is expected to increase, since it seems more plausible to find better approximations (or equivalently, smaller α_N) selecting the new frequency taking into account the interactions with the previously selected ones than, for example, matching the residue as best as possible.

5 Experiments

We now comment some generalities about the experiments performed in order to illustrate and test the algorithm for *SAOCIF* presented in section 3.

In the tables of results, *MPR* means "Matching the Previous Residue", and it is a sequential scheme where the new frequency is selected so as to match the previous residue as best as possible and the previous coefficients are not recalculated. *MPR* can be implemented with a slight change in the algorithm in figure 3: the selected frequency maximizes $\langle f - X_{N-1}, v_\omega \rangle^2 / \|v_\omega\|^2$ and the coefficient of the new hidden unit is computed as $\lambda_N = \langle f - X_{N-1}, v_\omega \rangle / \|v_\omega\|^2$. The idea behind *MPR* is exactly the same as in PPR, PPLN or MP without *back-projection* (see section 2). As previously stated (see section 3.3), let F be the number of candidate frequencies for every hidden unit, T the number of patterns and I the input dimension. In this setting, the computational cost of *MPR* can be bounded by $C_2 \cdot N_{MPR} \cdot F \cdot h_2(T \cdot I)$, where C_2 is constant, N_{MPR} is the number of hidden units of the network obtained with *MPR* and $h_2(T \cdot I) \in O(T \cdot I)$. When *back-projection* is present (that is, the coefficients are optimized after the selection of the frequency), we will refer to this method as "Optimal Coefficients after Matching the Previous Residue" (*OCMPR*), equivalent to Orthogonal MP or ILQ. In this case, and assuming that $N_{OCMPR} \leqslant F$, the computational cost of *OCMPR* can be bounded by $C_3 \cdot N_{OCMPR} \cdot F \cdot h_3(T \cdot I) + C_4 \cdot N^4_{OCMPR}$, where C_3 and C_4 are constant, N_{OCMPR} is the number of hidden units of the resulting network and $h_3(T \cdot I) \in O(T \cdot I)$. As it can be seen, the respective computational costs of *SAOCIF* (see section 3.3), *MPR* and *OCMPR* strongly depend on the number of hidden units of the final models (N_{SAOCIF}, N_{MPR} and N_{OCMPR}). This is a very important issue in order to assess the execution times found for each method.

Several activation functions ('AF' in the tables) have been tested, such as linear (lin), logistic (lgt), cosine (cos) in the MLP model and Gaussian (gau) in the RBFN model. Output units did not have biases, and in the hidden units they were only present for the logistic activation function. Regarding the strategies to select the frequencies, the column 'WR' indicates the range of weights to look for candidate frequencies. When the value 'Input' is present, it means that the candidate frequencies are selected from the points in the training set, as explained in section 3. Standard parameters were used for the BGA (see [Belanche 2000], for example). The average number of hidden units in the resulting networks is shown in column 'NH'. An 'NP' value means "Not Possible", indicating that the learning of the training set was unsatisfactory.

5.1 HEA Data Sets

The data sets described in [Hwang et al. 1994] were used to compare the approximation capabilities of *MPR*, *OCMPR* and *SAOCIF*. These data sets were constructed from five non-linear two-dimensional functions with different levels of complexity. Every training set contains 225 points generated from the uniform distribution $U[0,1]^2$ with their corresponding function value. The test set comprises $10{,}000$ points of a regularly spaced grid on $[0,1]^2$. No noise was added. We will refer to those data sets as HEAn, where n varies from 1 to 5. These data sets have been widely used in the literature (see, for example, [Kwok and Yeung 1997b; Ma and Khorasani 2004; Treadgold and Gedeon 1999]). Different from [Hwang et al. 1994], 10 independent training sets were constructed and trained for every function. In some preliminary experiments, we observed that overfitting was not present with these data sets. Results are shown in table 1 as the average, over 10 runs with the respective training sets for every function, of the normalized mean squared error

$$\text{NMSE} = \frac{\sum_{i=1}^{T} \left(\text{HEA}n(x_i) - X_N(x_i)\right)^2}{\sum_{i=1}^{T} \left(\text{HEA}n(x_i) - \overline{\text{HEA}n}\right)^2}$$

after the addition of 30 hidden units. For every model (*MPR*, *OCMPR* and *SAOCIF*), the frequencies were selected with the BGA strategy (100 generations of an initial random population of 100 individuals) without tuning. The activation function was also selected for every unit, in addition to the frequency

	Training			Test		
Data Set	*MPR*	*OCMPR*	*SAOCIF*	*MPR*	*OCMPR*	*SAOCIF*
HEA1	0.001937	0.000017	0.000000	0.002219	0.000073	0.000000
HEA2	0.002146	0.000175	0.000028	0.005095	0.000710	0.000133
HEA3	0.015259	0.006857	0.003222	0.059817	0.019668	0.011035
HEA4	0.002875	0.002993	0.000406	0.009441	0.008655	0.000994
HEA5	0.002725	0.001107	0.000506	0.007358	0.003459	0.001706

Table 1: NMSE in the training and test sets after the addition of 30 hidden units for the *MPR*, *OCMPR* and *SAOCIF* schemes. For every model, the frequencies were selected with the BGA strategy without tuning

as follows: for a predefined set of activation functions, the process of selecting the frequency is repeated for every activation function in the set. Different units may have different activation functions. The set of activation functions contained the hyperbolic tangent, Gaussian function, sine, cosine or the polynomial kernel $k(x) = (x+1)^2$, with MLP units.

As it can be observed in table 1, the approximation capability of *SAOCIF* is superior to that of *OCMPR*, which in turn compares favorably with *MPR*. Generalization results are also better for *SAOCIF* than *OCMPR* and *MPR*. Since overfitting was not observed during the learning process with these data sets, the best results are obtained by those models that are able to fit more accurately the data. Therefore, the suitability of the idea of interacting frequencies is confirmed. Taking the mean execution time for *MPR* as 1, the relative execution times for *OCMPR* and *SAOCIF* in these experiments were 1.01 and 1.25, respectively. Note that the relative improvement of *SAOCIF* with respect to *MPR* and *OCMPR* (measured as the ratio between the respective NMSE in the test set) is always greater than 1.78. Therefore, a better performance is obtained with a moderate increase in the execution time.

5.2 The *Two Spirals* Data Set

The well-known *Two spirals* problem consists in identifying the points of two interlocking spirals. Both training and test set comprise 194 points with balanced classes. This problem was tested with *MPR*, *OCMPR* and *SAOCIF* for the logistic activation function. The frequencies were selected randomly (200 attempts) within a certain range of weights. There was no further training to tune the frequency after a new hidden unit was added. The maximum number of hidden units added was 500, and no more hidden units were added when the whole training set was learnt.

Results are shown in table 2 as the average of 10 runs. The Cascade-Correlation algorithm (CASCOR, see section 2) was also tested. Column 'Train' indicates the percentage of the training set which has been learnt. Column 'Test' indicates the generalization performance obtained by an average-output committee of the resulting networks. As already known, this is a very hard

Method	AF	WR	NH	Train	Test
MPR	lgt	[-16,+16]	500	98.97%	99.48%
MPR	lgt	[-8,+8]	NP	-	-
OCMPR	lgt	[-8,+8]	105.2	100%	100%
SAOCIF	lgt	[-8,+8]	91.7	100%	100%
CASCOR	lgt	[-1,+1]	13.3	100%	99.48%

Table 2: Results for the *Two spirals* problem with the Random strategy (*MPR*, *OCMPR*, *SAOCIF*) and CASCOR. The reduced number of hidden units in CASCOR is due to its cascade architecture, but its average number of frequencies was 101.75

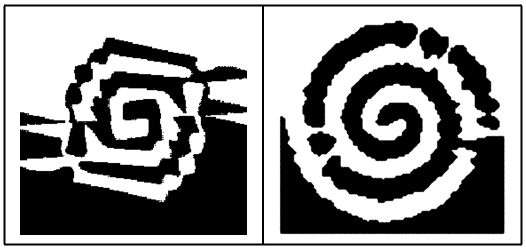

Figure 4: Generalization obtained by CASCOR (left) and *SAOCIF* (right) respectively with logistic functions for the *Two spirals* problem.

problem for the logistic function because of its intrinsic high non-linearity and radial symmetry, but it could be learnt with *SAOCIF* and an adequate (and very large) range of weights. *MPR* could not solve the problem with the same ranges, at least with 500 hidden units. Generalization in the obtained solutions was very good. Figure 4 compares the generalization obtained by the CASCOR and *SAOCIF* algorithms.

In these experiments, *OCMPR* took less execution time than *SAOCIF*, which in turn was faster than *MPR*. The execution time for *MPR* was very high because of the large number of needed hidden units. Taking the mean execution time for *OCMPR* as 1, the relative execution times for *SAOCIF* and *MPR* in these experiments were 1.52 and 10.98, respectively. However, as expected from its better approximation capability, the number of hidden units in the obtained solutions is smaller for *SAOCIF* than for *OCMPR* and *MPR*.

5.3 The *Pima Indians Diabetes* Data Set

Another comparison was performed with a widely used medical diagnosis data set, namely the "Pima Indians Diabetes" database produced in the Applied Physics Laboratory, Johns Hopkins University, 1988. The same data set as in [Prechelt 1997] was used, with input values in $[0, 1]$.

This data set was tested with *MPR*, *OCMPR* and *SAOCIF* for different activation functions. The Input and the BGA strategies were tested to select the candidate frequencies. The BGA constructed, for every frequency, 20 generations of an initial random population of 100 individuals. Every selected frequency was further tuned with Back-Propagation. After every epoch, the coefficients were computed as in the selection step. Learnings rates of the frequencies were automatically set following the *bold driver* technique proposed in [Vogl et al. 1988], and the momentum was set to 0. The tuning was stopped when the relative rate of decrease of the error was less than 0.0005. In some of the experiments, linear activation functions were combined with non-linear activation functions in the hidden layer ('lin-fun' in the tables). When linear activation functions were present, their optimal frequencies can be calculated analytically, solving a linear equations system similar to (2) and setting the coefficients to 1. For the *SAOCIF* scheme, the coefficients of the linear hidden units are kept fixed when new non-linear units are added, in order to approximate only the non-linear component of the function. Different from the *Two Spirals* data set, previous experience with this data set suggested that it would be necessary to control the complexity of the model in order to obtain a good performance. A double 5-4-fold Cross-Validation (CV) was performed as follows [Ripley 1995]. First, a 5-fold CV (the outer CV) was performed to obtain five folds (4 folds to "learn" and 1 fold to test). Then, the 4 folds of the "learning set" of the outer CV were used as follows: 3 folds to train and 1 fold to validate, as in a 4-fold CV (the inner CV). Therefore, the number of trained models in a double 5-4-fold CV is 20. For the BGA strategy, this procedure was repeated 5 times, in order to alleviate the effect of random initial populations. Previous to every CV, the examples in the data set were randomly shuffled. For every training, no more hidden units were added when the error on the validation set did not improve for 5 consecutive hidden units.

The results are shown in table 3, as the average in the network with minimum validation set error. The column 'Test' indicates the mean generalization performance obtained by the average-output committees of the resulting networks in the inner CV. Although these results seem very similar for the different parameter configurations, there are some regularities which can be observed for this problem:

1. Non-linear activation functions different from the classical sigmoidal and Gaussian (such as cosines) may be satisfactorily used. Linear hidden units have mostly a positive influence on the results, and selecting the frequencies from the points in the data set seems well suited not only for RBFNs, as commonly used, but also for MLPs.

2. Interestingly, the performance of the Input strategy was quite similar to that of the BGA strategy, with a much smaller computational cost. Although more experiments are needed, it may be an interesting idea to obtain good and cheap models for data sets of moderate size.

3. The number of hidden units of the solutions obtained with *SAOCIF* is

		Input			BGA		
Method	AF	WR	NH	Test	WR	NH	Test
MPR	lgt	Input	5.65	76.60%	[-0.5,0.5]	3.29	76.24%
MPR	lin-lgt	Input	14.25	77.12%	[-0.5,0.5]	11.16	77.12%
MPR	cos	Input	3.80	76.47%	[-1.5,1.5]	1.90	76.99%
MPR	lin-cos	Input	3.65	76.47%	[-1.5,1.5]	3.27	77.23%
MPR	gau	Input	9.10	76.73%	[0,1]	7.66	76.78%
MPR	lin-gau	Input	11.90	76.99%	[0,1]	6.65	76.97%
OCMPR	lgt	Input	5.10	76.47%	[-0.5,0.5]	3.13	76.18%
OCMPR	lin-lgt	Input	3.90	76.47%	[-0.5,0.5]	4.13	76.63%
OCMPR	cos	Input	2.85	76.99%	[-1.5,1.5]	1.97	77.02%
OCMPR	lin-cos	Input	4.90	76.73%	[-1.5,1.5]	3.93	76.86%
OCMPR	gau	Input	6.95	76.73%	[0,1]	4.52	76.05%
OCMPR	lin-gau	Input	7.75	76.60%	[0,1]	5.43	76.44%
SAOCIF	lgt	Input	3.00	76.47%	[-0.5,0.5]	2.30	76.05%
SAOCIF	lin-lgt	Input	4.30	77.25%	[-0.5,0.5]	4.90	77.07%
SAOCIF	cos	Input	2.40	76.34%	[-1.5,1.5]	2.47	77.18%
SAOCIF	lin-cos	Input	4.20	77.39%	[-1.5,1.5]	3.12	77.39%
SAOCIF	gau	Input	4.80	78.04%	[0,1]	5.42	77.41%
SAOCIF	lin-gau	Input	4.90	77.12%	[0,1]	4.70	77.36%

Table 3: Results for the *Diabetes* data set with the Input (left) and BGA (right) strategy for the *MPR*, *OCMPR* and *SAOCIF* schemes

usually less than the number of hidden units of the solutions obtained with *OCMPR* and *MPR*. The mean number of hidden units for *SAOCIF*, *OCMPR* and *MPR* in table 3 were 3.88, 4.55 and 6.86, respectively.

4. For every non-linear activation function tested, the best results are obtained with the *SAOCIF* scheme, when compared to *OCMPR* and *MPR*. This fact is related to the number of hidden units of the obtained solutions, since *SAOCIF* obtains simpler models (in terms of number of hidden units) with the same level of approximation.

5. In this case, the execution times were very similar for *SAOCIF*, *OCMPR* and *MPR*. The larger computational cost of every step for *SAOCIF* was compensated with the lower number of steps (number of hidden units) in the obtained solutions.

It is not easy to make a direct comparison of these results with previous studies on this benchmark. Since the results are very sensitive to the partitions made on the data set, a realistic comparison can only be made when similar experimental techniques are used. In table 4 our results are compared to several methods found in the literature that have been applied to this data set with similar resampling techniques. In the same aforementioned conditions, the CASCOR

[Freund and Schapire 1996]	Boosting	75.60%
[Wilson and Martinez 1996]	Heterogeneous RBFNs	76.30%
[Rätsch et al. 1998]	RBFNs	75.90%
[Rätsch et al. 1998]	AdaBoost	76.10%
[Rätsch et al. 1998]	Support Vector Machines	76.50%
[Vincent and Bengio 2002]	Kernel Matching Pursuit	76.10%
This work	Cascade-Correlation	76.76%
This work	*MPR* (BGA/lin-cos)	77.23%
This work	*OCMPR* (BGA/cos)	77.02%
This work	*SAOCIF* (Input/gau)	**78.04%**

Table 4: Comparison of results for the *Diabetes* data set

algorithm was tested for more than 70 different sets of parameters. Results are also shown in table 4.

6 Conclusions and Future Work

An algorithm for *Sequential Approximation with Optimal Coefficients and Interacting Frequencies* (*SAOCIF*) for FNNs has been presented. Coefficients are optimized for every candidate frequency, so that the approximations maintain orthogonal-like properties. The new frequency is not chosen in order to match the previous residue as best as possible. It is selected with global approximation of the target vector criteria (interacting frequencies). The algorithm can be seen as an extension and generalization of the OLSL [Chen et al. 1991] and KMP with *pre-fitting* [Vincent and Bengio 2002] algorithms in several ways. The idea behind *SAOCIF* can be extended to general Hilbert spaces. Theoretical results prove that, under reasonable conditions, the residue of the obtained approximation is (in the limit) the best one that can be obtained with any subset of the given set of vectors.

Experimental results show a very satisfactory performance for *SAOCIF* and several suggesting ideas for future experiments, such as the selection of the frequencies from the data set, as in OLSL and KMP, or the combination of linear and non-linear activation functions in the hidden units. The candidate frequencies can be selected with heuristics different from current strategies. In principle, a more intelligent selection could lead to better approximations. Likewise, the selection of the activation function for the new hidden unit admits any number of heuristics.

The theoretical results (see the proofs for details) are a consequence of the optimality of the coefficients in *SAOCIF*. The importance of the interacting frequencies lies in the hypothesis that it seems more plausible to find better partial approximations selecting the new frequency taking into account the interactions with the previous frequencies than, for example, matching the residue as best as possible. Therefore, the actual rate of convergence is expected to increase with this strategy. Experimental results confirmed this hypothesis. In order to theoretically prove this claim it would be necessary to formalize the concept of "interacting frequencies", a non-trivial task in our opinion, since it is not an existential property but a procedural one.

Acknowledgments

This work has been supported by the Consejo Interministerial de Ciencia y Tecnología (CICYT), under projects DPI2002-03225 and CGL2004-04702-C02-02.

Appendix A

(Pa) Evident, since the coefficients of X_{N+1} are optimal.

(Pb) Expressing $f - X_N$ as $f - X_M + X_M - X_N$ we have

$$\langle f-X_M, f-X_N\rangle = \|f-X_M\|^2 + \langle f - X_M, X_M-X_N\rangle .$$

By (1), $\langle f - X_M, X_M - X_N\rangle = 0$ holds.

(Pc) The necessity is clear by (1). To prove the sufficiency, suppose that $X_N = X_{N-1}+\lambda_N^N v_{\omega_N}$. By (1), for every $1 \leqslant j \leqslant N-1$ we have $\left\langle f - X_N, v_{\omega_j}\right\rangle = 0$ and $\left\langle f - X_{N-1}, v_{\omega_j}\right\rangle = 0$. Therefore, for every $1 \leqslant j \leqslant N-1$ we have

$$0 = \left\langle f - X_N, v_{\omega_j}\right\rangle = \left\langle f - X_{N-1} - \lambda_N^N v_{\omega_N}, v_{\omega_j}\right\rangle = \left\langle -\lambda_N^N v_{\omega_N}, v_{\omega_j}\right\rangle ,$$

which finishes the proof, since $\lambda_N^N \neq 0$.

(Pd) Since H is complete, it suffices to prove that $\lim_{N,M\to\infty} \|X_M - X_N\|^2 = 0$. Suppose that $M > N$. Expressing $X_M - X_N$ as $(X_M - f) + (f - X_N)$, and using (Pb) we easily obtain

$$\|X_M - X_N\|^2 = \|f - X_N\|^2 - \|f - X_M\|^2.$$

Since the sequence $\{\|f - X_N\|^2\}_{N\geqslant 0}$ is decreasing and positive (Pa), it is convergent. Hence,

$$\lim_{N,M\to\infty} \left(\|f - X_N\|^2 - \|f - X_M\|^2\right) = 0.$$

$\square$

Appendix B

(Ta1) From Schwartz inequality, for every $\omega_0 \in \Omega$ we have

$$|\langle g - X_N, v_{\omega_0}\rangle| \leqslant \|g - X_N\| \, \|v_{\omega_0}\| .$$

The proof finishes using that $\lim_{N\to\infty} \|g - X_N\| = 0$.

(Ta2) Let $\omega_0 \in \Omega$. By hypothesis, for every $N \geqslant 0$ and every $\mu \in \mathbb{C}$

$$\begin{aligned}\|f - X_{N+1}\|^2 &\leqslant \|f - (X_N + \mu v_{\omega_0})\|^2 + \alpha_N \\ &= \|f - X_N\|^2 - 2\mathrm{Re}\,(\langle f - X_N, \mu v_{\omega_0}\rangle) + |\mu|^2 \|v_{\omega_0}\|^2 + \alpha_N.\end{aligned}$$

Expressing $f - X_N$ as $f - g + g - X_N$ we have

$$\begin{aligned}&\|f - X_{N+1}\|^2 - \|f - X_N\|^2 \leqslant \\ &\qquad |\mu|^2 \|v_{\omega_0}\|^2 - 2\mathrm{Re}\,(\langle f - g, \mu v_{\omega_0}\rangle) - 2\mathrm{Re}\,(\langle g - X_N, \mu v_{\omega_0}\rangle) + \alpha_N.\end{aligned}$$

Hence, renaming $D_N = \|f - X_N\|^2 - \|f - X_{N+1}\|^2$,

$$\begin{aligned}2\mathrm{Re}\,(\langle f - g, \mu v_{\omega_0}\rangle) - |\mu|^2 \|v_{\omega_0}\|^2 &\leqslant D_N - 2\mathrm{Re}\,(\langle g - X_N, \mu v_{\omega_0}\rangle) + \alpha_N \\ &\leqslant D_N + 2\,|\langle g - X_N, \mu v_{\omega_0}\rangle| + \alpha_N \\ &= D_N + 2\,|\mu|\,|\langle g - X_N, v_{\omega_0}\rangle| + \alpha_N.\end{aligned}$$

for every $\mu \in \mathbb{C}$. Let $\mu_0 = \frac{\langle f-g, v_{\omega_0}\rangle}{\|v_{\omega_0}\|^2}$, and $\varepsilon > 0$.
Using (Pa), (Ta1), and the hypothesis, there exists N_0 such that for every $N \geqslant N_0$,

$$\|f - X_N\|^2 - \|f - X_{N+1}\|^2 \leqslant \varepsilon/3,$$

$$2\,|\mu_0|\,|\langle g - X_N, v_{\omega_0}\rangle| \leqslant \varepsilon/3,$$

$$\alpha_N \leqslant \varepsilon/3.$$

Thus we have $2\mathrm{Re}\,(\langle f - g, \mu_0 v_{\omega_0}\rangle) - |\mu_0|^2 \|v_{\omega_0}\|^2 \leqslant \varepsilon$. Since

$$\langle f - g, \mu_0 v_{\omega_0}\rangle = \overline{\mu_0}\,\langle f - g, v_{\omega_0}\rangle = |\mu_0|^2 \|v_{\omega_0}\|^2,$$

$2\mathrm{Re}\,(\langle f - g, \mu_0 v_{\omega_0}\rangle) = 2|\mu_0|^2 \|v_{\omega_0}\|^2$ holds. Therefore, for every $\varepsilon \geqslant 0$

$$|\mu_0|^2 \|v_{\omega_0}\|^2 = 2\mathrm{Re}\,(\langle f - g, \mu_0 v_{\omega_0}\rangle) - |\mu_0|^2 \|v_{\omega_0}\|^2 \leqslant \varepsilon.$$

Hence, $|\mu_0|^2 \|v_{\omega_0}\|^2 = 0$. Since $\|v_{\omega_0}\|^2 \neq 0$, we derive $\mu_0 = 0$. Therefore, $\langle f - g, v_{\omega_0}\rangle = 0$ for every $\omega_0 \in \Omega$.

(Ta3) Using (Ta2) and Schwartz inequality we have

$$|\langle f - g, g\rangle| = |\langle f - g, g - X_N\rangle| \leqslant \|f - g\|\,\|g - X_N\|\,.$$

The proof finishes using that $\lim_{N\to\infty} \|g - X_N\| = 0$.

(Ta4) According to (Ta2), any vector combination $\sum_k \mu_k v_{\psi_k}$ in $v(\Omega)$ satisfies $\langle f - g, \sum_k \mu_k v_{\psi_k}\rangle = 0$. Hence we have

$$\|f - g\|^2 = \left\langle f - g, f - \sum_k \mu_k v_{\psi_k} \right\rangle - \langle f - g, g\rangle\,.$$

Using (Ta3) and Schwartz inequality we have

$$\|f-g\|^2 \leqslant \|f-g\| \left\| f - \sum_k \mu_k v_{\psi_k} \right\| .$$

Therefore, $\|f-g\| \leqslant \|f - \sum_k \mu_k v_{\psi_k}\|$ for any vector combination. The other inequality is clear, since for every $N \geqslant 0$

$$\inf_{\mu_k \in \mathbb{C},\ \psi_k \in \Omega} \left\| f - \sum_k \mu_k v_{\psi_k} \right\| \leqslant \|f - X_N\| \leqslant \|f-g\| + \|g - X_N\| .$$

The proof finishes using that $\lim_{N\to\infty} \|g - X_N\| = 0$.

(Tb) It is immediately derived from (Ta4).

$\square$

References

Achieser, N. I. (1956). *Theory of Approximation.* Frederick Ungar Pub. Co., New York.

Ash, T. (1989). Dynamic Node Creation in Backpropagation Networks. *Connection Science*, 1(4):365–375.

Barron, A. R. (1993). Universal Approximation Bounds for Superposition of a Sigmoidal Function. *IEEE Transactions on Information Theory*, 39(3):930–945.

Barron, A. R. (1994). Approximation and Estimation Bounds for Artificial Neural Networks. *Machine Learning*, 14(1):115–133.

Bartlett, E. B. (1994). Dynamic Node Architecture Learning: An Information Theoretic Approach. *Neural Networks*, 7(1):129–140.

Belanche, L. (2000). A Case Study in Neural Network Training with the Breeder Genetic Algorithm. Technical Report LSI-00-7-R, Departament de Llenguatges i Sistemes Informàtics, Universitat Politècnica de Catalunya, Spain.

Bello, M. G. (1992). Enhanced Training Algorithms, and Integrated Training/Architecture Selection for Multilayer Perceptron Networks. *IEEE Transactions on Neural Networks*, 3(6):864–875.

Bishop, C. M. (1995). *Neural Networks for Pattern Recognition.* Oxford University Press Inc., New York.

Chen, S., Billings, S. A., and Luo, W. (1989). Orthogonal Least Squares Methods and their Applications to Non-linear System Identification. *International Journal of Control*, 50(5):1873–1896.

Chen, S., Chng, E. S., and Alkadhimi, K. (1996). Regularized Orthogonal Least Squares Algorithm for Constructing Radial Basis Function Networks. *International Journal of Control*, 64(5):829–837.

Chen, S., Cowan, C. F. N., and Grant, P. M. (1991). Orthogonal Least Squares Learning Algorithm for Radial Basis Function Networks. *IEEE Transactions on Neural Networks*, 2(2):302–309.

Chen, S., Wu, Y., and Luk, B. L. (1999). Combined Genetic Algorithm and Regularized Orthogonal Least Squares Learning for Radial Basis Function Networks. *IEEE Transactions on Neural Networks*, 10(5):1239–1243.

Faddeeva, V. N. (1959). *Computational Methods of Linear Algebra*. Dover Publications Inc., New York.

Fahlman, S. E. and Lebiere, C. (1990). The Cascade-Correlation Learning Architecture. In *Advances in Neural Information Processing Systems*, volume 2, pages 524–532. Morgan Kaufmann.

Freund, Y. and Schapire, R. E. (1996). Experiments with a New Boosting Algorithm. In *13th International Conference on Machine Learning*, pages 148–156.

Friedman, J. H. and Stuetzle, W. (1981). Projection Pursuit Regression. *Journal of the American Statistical Association*, 76:817–823.

Geman, S., Bienenstock, E., and Doursat, R. (1992). Neural Networks and the Bias/Variance Dilemma. *Neural Computation*, 4(1):1–58.

Huber, P. J. (1985). Projection Pursuit. *The Annals of Statistics*, 13(2):435–475.

Hwang, J. N., Ray, S. R., Maechler, M., Martin, D., and Schimert, J. (1994). Regression Modelling in Back-Propagation and Projection Pursuit Learning. *IEEE Transactions on Neural Networks*, 5(3):342–353.

Islam, M. M. and Murase, K. (2001). A New Algorithm to Design Compact Two-hidden-layer Artificial Neural Networks. *Neural Networks*, 14(9):1265–1278.

Jones, L. K. (1987). On a Conjecture of Huber Concerning the Convergence of Projection Pursuit Regression. *The Annals of Statistics*, 15(2):880–882.

Jones, L. K. (1992). A Simple Lemma on Greedy Approximation in Hilbert Space and Convergence Rates for Projection Pursuit Regression and Neural Network Training. *The Annals of Statistics*, 20(1):608–613.

Kadirkamanathan, V. and Niranjan, M. (1993). A Function Estimation Approach to Sequential Learning with Neural Networks. *Neural Computation*, 5(6):954–975.

Kůrková, V. (1998). Incremental Approximation by Neural Networks. In Karny, M., Warwick, K., and Kůrková, V., editors, *Dealing With Complexity: A Neural Network Approach*, pages 177–188. Springer-Verlag, London.

Kůrková, V. and Beliczyński, B. (1995). Incremental Approximation by One-Hidden-Layer Neural Networks. In *International Conference on Artificial Neural Networks*, volume 1, pages 505–510.

Kwok, T. Y. and Yeung, D. Y. (1997a). Constructive Algorithms for Structure Learning in Feedforward Neural Networks for Regression Problems. *IEEE Transactions on Neural Networks*, 8(3):630–645.

Kwok, T. Y. and Yeung, D. Y. (1997b). Objective Functions for Training New Hidden Units in Constructive Neural Networks. *IEEE Transactions on Neural Networks*, 8(5):1131–1148.

Lahnajärvi, J. J. T., Lehtokangas, M. I., and Saarinen, J. P. P. (1999). Fixed Cascade Error - A Novel Constructive Neural Network for Structure Learning. In *International Conference on Artificial Neural Networks in Engineering*, pages 25–30.

Lahnajärvi, J. J. T., Lehtokangas, M. I., and Saarinen, J. P. P. (2002). Evaluation of Constructive Neural Networks with Cascaded Architectures. *Neurocomputing*, 48(1-4):573–607.

Lehtokangas, M. I. (1999). Modelling with Constructive Backpropagation. *Neural Networks*, 12(4-5):707–716.

Littmann, E. and Ritter, H. (1992). Cascade LLM Networks. In *International Conference on Artificial Neural Networks*, volume 1, pages 253–257.

Ma, L. and Khorasani, K. (2004). New Training Strategies for Constructive Neural Networks with Application to Regression Problems. *Neural Networks*, 17(4):589–609.

Mallat, S. G. and Zhang, Z. (1993). Matching Pursuits with Time-Frequency Dictionaries. *IEEE Transactions on Signal Processing*, 41(12):3397–3415.

Mühlenbein, H. and Schlierkamp-Voosen, D. (1993). Predictive Models for the Breeder Genetic Algorithm I. Continuous Parameter Optimization. *Evolutionary Computation*, 1(1):25–49.

Pati, Y. C., Rezaiifar, R., and Krishnaprasad, P. S. (1993). Orthogonal Matching Pursuit: Recursive Function Approximation with Application to Wavelet Decomposition. In *27th Asilomar Conference on Signals, Systems and Computers*, volume 1, pages 40–44.

Platt, J. (1991). A Resource-Allocating Network for Function Interpolation. *Neural Computation*, 3(2):213–225.

Prechelt, L. (1997). Investigation of the CasCor Family of Learning Algorithms. *Neural Networks*, 10(5):885–896.

Qian, S. and Chen, D. (1994). Signal Representation using Adaptive Normalized Gaussian Functions. *Signal Processing*, 36(1):1–11.

Rätsch, G., Onoda, T., and Müller, K. R. (1998). An Improvement of AdaBoost to Avoid Overfitting. In *International Conference on Neural Information Processing*, pages 506–509.

Ripley, B. D. (1995). Statistical Ideas for Selecting Network Architectures. In Kappen, B. and Gielen, S., editors, *Neural Networks: Artificial Intelligence and Industrial Applications*, pages 183–190. Springer-Verlag, London.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning Internal Representations by Error Propagation. In Rumelhart, D. E. and McClelland, J. L., editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition (vol. 1)*, pages 318–362. MIT Press.

Salmerón, M., Ortega, J., Puntonet, C. G., and Prieto, A. (2001). Improved RAN Sequential Prediction using Orthogonal Techniques. *Neurocomputing*, 41(1-4):153–172.

Shin, Y. and Ghosh, J. (1995). Ridge Polynomial Networks. *IEEE Transactions on Neural Networks*, 6(3):610–622.

Smola, A. J. and Bartlett, P. (2001). Sparse Greedy Gaussian Process Regression. In *Advances in Neural Information Processing Systems*, volume 13, pages 619–625. MIT Press.

Sopena, J. M., Romero, E., and Alquézar, R. (1999). Neural Networks with Periodic and Monotonic Activation Functions: A Comparative Study in Classification Problems. In *9th International Conference on Artificial Neural Networks*, volume 1, pages 323–328.

Treadgold, N. K. and Gedeon, T. D. (1999). Exploring Constructive Cascade Networks. *IEEE Transactions on Neural Networks*, 10(6):1335–1350.

Vapnik, V. N. (1995). *The Nature of Statistical Learning Theory.* Springer-Verlag, NY.

Vincent, P. and Bengio, Y. (2002). Kernel Matching Pursuit. *Machine Learning*, 48(1-3):165–187. Special Issue on New Methods for Model Combination and Model Selection.

Vogl, T. P., Mangis, J. K., Rigler, A. K., Zink, W. T., and Alkon, D. L. (1988). Accelerating the Convergence of the Back-Propagation Method. *Biological Cybernetics*, 59:257–263.

Wang, Z., Di Massimo, C., Tham, M. T., and Morris, J. (1994). A Procedure for Determining the Topology of Multilayer Feedforward Neural Networks. *Neural Networks*, 7(2):291–300.

Wilson, D. R. and Martinez, T. R. (1996). Heterogeneous Radial Basis Function Networks. In *International Conference on Neural Networks*, volume 2, pages 1263–1267.

Yingwei, L., Sundararajan, N., and Saratchandran, P. (1997). A Sequential Learning Scheme for Function Approximation using Minimal Radial Basis Function Neural Networks. *Neural Computation*, 9(2):461–478.

Yosida, K. (1965). *Functional Analysis.* Springer-Verlag, New York.