# A FUNCTION ESTIMATION APPROACH TO SEQUENTIAL LEARNING WITH NEURAL NETWORKS

Visakan Kadirkamanathan & Mahesan Niranjan

**CUED/F-INFENG/TR.111**

13 Septmeber 1992

Cambridge University Engineering Department
Trumpington Street
Cambridge CB2 1PZ
England

Email: visakan@eng.cam.ac.uk, niranjan@eng.cam.ac.uk

**Abstract**

In this paper, we investigate the problem of optimal sequential learning, viewed as a problem of estimating an underlying function sequentially rather than estimating a set of parameters of the neural network.

Firstly, we arrive at a sub-optimal solution to the sequential estimate which can be mapped by a growing Gaussian radial basis function (GaRBF) network. This network adds hidden units for each observation. The function space approach in which the estimates are represented as vectors in a function space, is used in developing a growth criteria to limit its growth. A simplification of the criterion leads to two joint criteria on the distance of the present pattern and the existing unit centres in the input space and on the approximation error of the network for the given observation to be satisfied together. This network is similar to the resource allocating network (RAN) [12] and hence RAN can be interpreted from a function space approach to sequential learning.

Secondly, we present an enhancement to the RAN. The RAN either allocates a new unit based on the novelty of an observation or adapt the network parameters by the LMS algorithm. The function space interpretation of the RAN lends itself to an enhancement of the RAN in which the extended Kalman filter (EKF) algorithm is used in place of the LMS algorithm. The performance of the RAN and the enhanced network are compared in the experimental tasks of function approximation and time-series prediction demonstrating the superior performance of the enhanced network with fewer number of hidden units. The approach adopted here has led us towards the minimal network required for a sequential learning problem.

# 1 Introduction

Artificial neural networks (ANNs) provide an input – output mapping and hence their output can be written as a function of the inputs and of the parameters or weights in the network. Learning in neural networks amount to the approximation of an underlying function which in turn reduces to the estimation of the parameters (or weights) that is optimal in some sense such as least squared approximation error. The conventional approaches to sequential learning view the problem in the parameter space, *ie.* as estimation of a set of parameters. Such an approach requires the complexity or size of the ANN to be specified *a priori*. We have adopted an alternative approach of estimating a function, in which we view the problem in a function space (infinite dimensional space of square integrable real functions) where different complexity ANN mappings can be represented. As we shall see later, the real advantage is demonstrated by the development of a growing network from this approach.

The function estimation approach to sequential learning, in which the task is to estimate an underlying function sequentially, led to the development of the *principle of $\mathcal{F}$-Projection* [5], [6], [9]. The principle is used in deriving a sequential estimate that is mapped by a Gaussian radial basis function (GaRBF) network that adds a hidden unit for each observation. This method of estimation may be looked upon as a sequential method of kernel based nonparametric estimation such as the Parzen window density estimation [3]. It differs from the statistical approach adopted by White [15] who specifies the rate of growth of the number of hidden units in the ANN based only on the number of observations.

We also demonstrate how the function space approach, where the function estimates are analysed in the space of all square integrable functions, leads to a criterion to limit the growth of the network. The resulting network is a GaRBF network that adds a hidden unit subject to the present observation satisfying some growth criteria. It is equivalent to the resource allocating network (RAN) [12] but for the manner in which it is derived.

Platt [12] describes the RAN as a single hidden layer network of locally tuned hidden units whose responses are linearly combined to form an output response. It is essentially a GaRBF network. However, the RAN starts with no hidden units and grows by allocating hidden units based on the 'novelty' of an observation. Since the novelty of each observation is tested, it is ideally suited for sequential learning problems such as on-line prediction and control. The objective behind its development is to gradually approach the appropriate complexity of the network that is sufficient to provide an approximation to an underlying mapping that is consistent with the observations being received. The RAN may be viewed as an extension of the restricted Coulomb energy (RCE) model [14] of classification to solving the function interpolation problem.

When the novelty or the growth criterion is not satisfied, the existing RAN parameters are adapted by the LMS algorithm. While the growth criterion and the allocation of a new hidden unit can be explained from the function space approach, the adaptation by LMS seem to be a weak step in an otherwise optimal procedure. The RAN can be enhanced by adopting the function space approach in the adaptation stage as well. The enhancement we suggest is to use the extended Kalman filter (EKF) algorithm in place of the LMS algorithm. The performances of the RAN and the enhanced RAN are compared in two experiments.

The organisation of the paper is as follows: The next section provides a brief description of the preliminary concepts and the notations used in the paper. Section 3 introduces the function space approach to sequential learning. Section 4 develops the growth criterion for the network and in section 5, Platt's description of the RAN is given along with a discussion on its equivalence to the network derived from the function space approach. Section 6 contains the description of the enhanced RAN. The experimental results are given in section 7 followed by conclusion in section 8.

## 2 Preliminaries and Notations

The ANN learns the mapping from a set of data in the form of input – output observation pairs $(\mathbf{x}_n, y_n)$, where $\mathbf{x}_n$ is an $M$-dimensional input vector and $y_n$ is an output scalar. The input lies in a subset $\mathcal{D}$ of the space of all real valued $M$-dimensional vectors $\Re^M$. The $n^{th}$ observation can then be described as:

$$\mathcal{I}^{(n)} = \left\{(\mathbf{x}_n, y_n) : \mathbf{x}_n \in \mathcal{D} \subseteq \Re^M ; y_n \in \Re\right\} \tag{1}$$

The observations $\mathcal{I}^{(n)}$, $n = 1, \ldots, N$ are assumed to be free of noise and be consistent with an underlying function $f_*$, *viz.*,

$$f_*(\mathbf{x}_n) = y_n \qquad \text{for} \qquad n = 1, \ldots, N \tag{2}$$

The mapping described by the ANN is denoted by $f(\mathbf{x})$ with a shorthand description of $f$. Hence, $f : \mathbf{x} \mapsto y(\mathcal{D} \mapsto \Re)$. The closeness between the ANN mapping and the underlying function is measured by some distance metric $D(f, f_*)$. A common and popular metric used with ANNs is the $L^2$-norm given by,

$$D(f,f_*) = \|f - f_*\| \tag{3}$$

where $\|\cdot\|$ denotes the $L^2$-norm. The squared $L^2$-norm[1] is given by,

$$\|f\|^2 = \int_{\mathbf{x}\in\mathcal{D}} |f(\mathbf{x})|^2 d\mathbf{x} \tag{4}$$

$|\cdot|$ gives the absolute value of its argument. Note that the $L^2$-norm of an $M$-dimensional vector $\mathbf{a}$ is also denoted by $\|\mathbf{a}\|$.

The $L^2$-norm describes a function space which contains all the square integrable real valued functions. Since an inner product can also be defined in this space, it is a *Hilbert space*, denoted by,

$$\mathcal{H} = \{f : \|f\| < \infty\} \tag{5}$$

[1]In general a weighting function $w(\mathbf{x})$ is used inside the integral. If the distribution of the past observations are not known a priori, $w(\mathbf{x})$ is taken to be uniform in $\mathcal{D}$ giving the expression in equation (4).

The mapping described by an ANN satisfies the above requirement in general and hence all possible functions an ANN can describe lie in the function space $\mathcal{H}$. The inner product between two functions $f, g \in \mathcal{H}$ is given by,

$$< f, g > = \int_{\mathbf{x} \in \mathcal{D}} f(\mathbf{x}) g(\mathbf{x}) d\mathbf{x} \tag{6}$$

The concepts from geometry can be applied in a Hilbert space, which leads us to the notion of angle between the two functions $f$ and $g$, given by,

$$\Omega = \cos^{-1} \left\{ \frac{< f, g >}{\|f\| \ \|g\|} \right\} \tag{7}$$

When the output of the network is not limited to the interval $(0, 1)$, the hidden – output layer transformation is linear. Then the single hidden layer ANN linearly combines the output of the hidden units. Each of the hidden units construct a mapping and hence these mappings can be viewed as the basis functions $\phi_k \in \mathcal{H}$, $k = 1, \ldots, K$, the total number of hidden units being $K$. The output is thus represented as,

$$f(\mathbf{x}) = \sum_{k=1}^{K} \alpha_k \phi_k(\mathbf{x}) \tag{8}$$

In sequential estimation, an estimate is required at each time instant $(n)$. The ANN mapping after it has learned from the $n^{th}$ observation $\mathcal{I}^{(n)}$ is denoted by $f^{(n)}$. This is known as the *posterior* estimate of the underlying function and $f^{(n-1)}$ as the *prior*.

## 3 Sequential Function Estimation

The sequential function estimation problem can be stated as follows: Given the prior estimate $f^{(n-1)}$ and the new observation $\mathcal{I}^{(n)}$, how do we combine these two information in obtaining the posterior estimate $f^{(n)}$?

Given only the information above and the assumption that the observations are free of noise, one approach to sequential estimation is to choose an optimal estimate at each step. Any improvement over this solution would require additional memory such as the probability density of the past observations. The step-wise optimal estimate is given by the *principle of $\mathcal{F}$-Projection* [6], which states,

$$f^{(n)} = \arg \min_{f \in \mathcal{H}} \|f - f^{(n-1)}\| \qquad \text{s.t.} \qquad f^{(n)} \in \mathcal{H}_n \tag{9}$$

where $\mathcal{H}_n$ is the set consisting of all the functions in $\mathcal{H}$ that satisfy the constraint $f(\mathbf{x}_n) = y_n$. The posterior is a projection of the prior onto the space $\mathcal{H}_n$. The principle is an analogue of the projection algorithm for linear models [4], where the prior parameter vector is projected onto the constraint hyperplane in the parameter space.

The equality constraint $f(\mathbf{x}_n) = y_n$ can be rewritten as an inner product in the function space,

$$< f, \delta_n > = y_n \tag{10}$$

where $\delta_n = \delta(\mathbf{x} - \mathbf{x}_n)$ is the impulse function[2]. The constrained minimisation can be solved exactly to give,

$$f^{(n)} = f^{(n-1)} + e_n h_n \tag{11}$$

where $e_n$ is the prediction error, given by,

$$e_n = y_n - f^{(n-1)}(\mathbf{x}_n) \tag{12}$$

and

$$h_n = \frac{\delta_n}{\|\delta_n\|^2} \tag{13}$$

This solution amounts to adding a spike at the point $\mathbf{x}_n$ to $f^{(n-1)}(\mathbf{x})$ such that $f^{(n)}(\mathbf{x})$ goes through the point $(\mathbf{x}_n, y_n)$. Such a solution discounts the fact that the underlying function is smooth and an observation has a bearing on its neighbourhood in the input space $\mathcal{D}$. Smoothness constraints must then be added to obtain a posterior estimate[3].

The smoothness constraint must be imposed on $h_n$ which has the following property: $h_n(\mathbf{x}_n) = 1$ and $h_n(\mathbf{x}_n + \mathbf{a}) = 0$ for any $\|\mathbf{a}\| \neq 0$. Smoothing this impulse like function subject to the constraint that $f^{(n)}(\mathbf{x}_n) = y_n$ yields the Gaussian RBF $\phi_n$, given by,

$$\phi_n(\mathbf{x}) = \exp\left\{-\frac{1}{\sigma_n^2}\|\mathbf{x} - \mathbf{u}_n\|^2\right\} \tag{14}$$

with $\mathbf{u}_n = \mathbf{x}_n$ and $\sigma_n$ representing the required smoothness[4]. Now the properties of $\phi_n$ are: $\phi_n(\mathbf{x}_n) = 1$ and $\phi_n(\mathbf{x}_n + \mathbf{a}) \to 0$ as $\|\mathbf{a}\| \to \infty$. The parameter $\sigma_n$ is the spread of the GaRBF representing its span around $\mathbf{x}_n$ in the input space. This view is similar to the *method of potential functions* [3] where each observation in the input space contributes to its neighbourhood via the potential of a charge placed on the observation, the span signifying the region of influence of the charge.

Hence, from the principle of $\mathcal{F}$-Projection and smoothing its solution, we have arrived at the posterior function estimate $f^{(n)}$, given by,

$$f^{(n)}(\mathbf{x}) = f^{(n-1)}(\mathbf{x}) + e_n \phi_n(\mathbf{x}) \tag{15}$$

Let us use the GaRBF network to map the function estimate $f^{(n-1)}$. Assume there are $K$ hidden units (basis functions) in the network that maps $f^{(n-1)}$. Then the posterior is given by,

$$\begin{aligned} f^{(n)}(\mathbf{x}) &= \sum_{k=1}^{K} \alpha_k \phi_k(\mathbf{x}) + e_n \phi_n(\mathbf{x}) && (16) \\ &= \sum_{k=1}^{K+1} \alpha_k \phi_k(\mathbf{x}) && (17) \end{aligned}$$

[2]The $\delta_n$ does not have a finite $L^2$-norm and hence $\delta_n \notin \mathcal{H}$. However, a function such as a rectangular function that approaches the impulse function in the limit can be used.

[3]In the case of networks with finite number of hidden units, this exact solution cannot be reached. In fact, the smoothness constraint is implicit in the selection of the basis functions, which in turn ensures that the posterior estimate obtained from the algorithm based on the principle of $\mathcal{F}$-Projection is sufficiently smooth [9].

[4]The maximum curvature given by $\sup |\phi''(\mathbf{x})| = \frac{1}{\sigma^2}$ where $''$ denotes the second derivative with respect to $\mathbf{x}$.

The posterior estimate is mapped by the same GaRBF network with a new hidden unit added and the parameters associated with it are assigned as follows:

$$\alpha_{K+1} = e_n \tag{18}$$

$$\mathbf{u}_{K+1} = \mathbf{x}_n \tag{19}$$

$$\sigma_{K+1} = \sigma_n \tag{20}$$

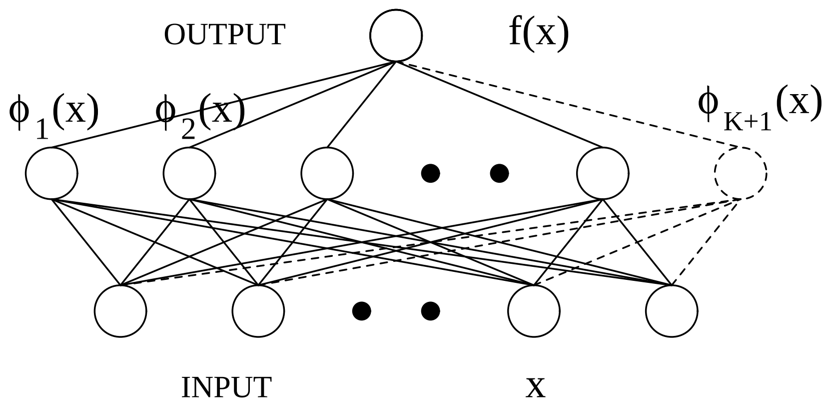



Figure 1: The network architecture of the growing GaRBF network. The dotted lines show the new links formed by the addition of a hidden unit.

Figure 1 shows the architecture of the network in which a hidden unit is added to map $f^{(n)}$. For the moment, we shall assume that we are given the value of $\sigma_n$. In the next secton we shall see how a reasonable value can be assigned.

The network we have arrived at grows with each new observation. The observations $\mathbf{x}_n$ are implicitly stored as the centres of the Gaussian hidden units and $e_n$ (hence $y_n$) are implicit in their coefficients. This estimate is similar in spirit to the Parzen window density estimation procedure where the number of kernels are the same as the number of observations and are centred on the input observations [3]. The difficulty with using this network for estimation is that the network grows indefinitely as the observations are continually received.

## 4 A Geometric Growth Criterion

In problems where the data is received sequentially, the network may have approximated the underlying function to a sufficient accuracy and may go on to add hidden units that contribute little to the final estimate. Not only does the complexity of the network increases unnecessarily, it adds to the computational burden significantly. Furthermore, if the data were noisy, good estimation of the network parameters require its number to be much smaller than the data from which they were estimated. This leads us to the question of how the network growth must be limited.

Consider the Hilbert space $\mathcal{H}$. The network solutions are points in this infinite dimensional space. If the network consists of $K$ hidden units (and hence $K$ basis functions), assuming that the parameters of the basis functions are not adapted, the network solutions lie in a $K$-dimensional subspace $\mathcal{H}_K$ formed by the $K$ basis functions. Figure 2 gives a 3-dimensional illustration of the network solutions.

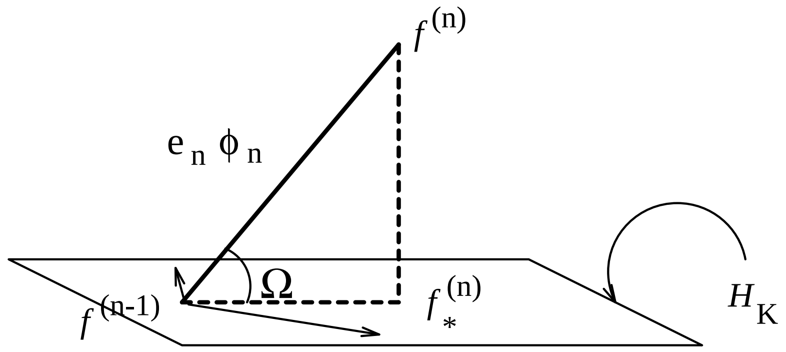



Figure 2: 3-D illustration of the prior and posterior network solutions in the Hilbert space.

The prior estimate $f^{(n-1)}$ and the posterior estimate obtained with the existing $K$ basis functions $f_*^{(n)}$, both lie in $\mathcal{H}_K$. The posterior estimate $f^{(n)}$ (given in equation (15)) is obtained by adding a new basis function $\phi_n$. Note that $f_*^{(n)}$ is the projection of $f^{(n)}$ onto $\mathcal{H}_K$ and hence is the closest point to $f^{(n)}$ in $\mathcal{H}_K$. The distance between $f^{(n)}$ and $f_*^{(n)}$ is given by $\|f^{(n)} - f_*^{(n)}\|$. This distance represents a measure of how bad our approximation will be if we do not add a new basis function. Hence, the decision to add a hidden unit can be based on this distance exceeding a threshold. From the geometry of the network solutions shown in figure 2, the criterion is stated as:

$$\|f^{(n)} - f_*^{(n)}\| = |e_n|.\|\phi_n\| \sin(\Omega) > \xi \tag{21}$$

where $\xi$ is a threshold and $\Omega$ is the angle formed by the new basis function $\phi_n$ to the subspace $\mathcal{H}_K$ defined by the $K$ basis functions in $f^{(n-1)}$. The norm of the basis function $\phi_n$ depends only on the width $\sigma_n$. The angle lies between 0 and $\frac{\pi}{2}$ and therefore $0 \leq \sin(\Omega) \leq 1$. Note that such an approach can also be adopted for block learning problems in choosing the form of the next basis function to be added.

The distance $\|f^{(n)} - f_*^{(n)}\|$ may be evaluated directly, but it is computationally intensive. Hence, the geometric criterion is simplified further as follows:

$$e_n > e_{\min} \tag{22}$$
$$\Omega > \Omega_{\min} \tag{23}$$

assuming that $\sigma_n$ is pre-determined in which case the growth criterion depends only on $e_n$ and $\Omega$.

These criteria are referred to as the *prediction error criterion* and the *angle criterion* respectively. The prediction error criterion checks for the interpolation of the present observation by the network. The angle criterion attempts to assign basis functions that are nearly orthogonal to all other existing basis functions[5].

The angle $\Omega$ is difficult to evaluate in general and an approximation of the angle criterion is for the smallest angle between the new basis function and all other existing basis functions to exceed a threshold. The angle between the two GaRBFs $\phi_k$ and $\phi_n$ with the same width $\sigma_k = \sigma_n = \sigma_0$ is given by (in [5]),

$$\Omega_k = \cos^{-1}\left(\exp\left\{-\frac{1}{2\sigma_0^2}\|\mathbf{x}_n - \mathbf{u}_k\|^2\right\}\right) \tag{24}$$

The angle criterion then reduces to,

$$\sup_k \quad \phi_k(\mathbf{x}_n) \leq \cos^2(\Omega_{\min}) \tag{25}$$

a threshold on the output of the basis functions to the input $\mathbf{x}_n$. This can equivalently be expressed as,

$$\inf_k \quad \|\mathbf{x}_n - \mathbf{u}_k\| \geq \epsilon_n \tag{26}$$

a threshold on the distance between the input $\mathbf{x}_n$ and the nearest GaRBF unit centre $\mathbf{u}_k$ with,

$$\epsilon_n = \sigma_0\sqrt{2\log(1/\cos^2\Omega_{\min})} \tag{27}$$

Even when the widths $\sigma_n$ are not equal, a similar criterion can be arrived at [5].

From equation (24) it is clear that the angle can be increased by lowering $\sigma_n$. However, lowering $\sigma_n$ increases the curvature of $\phi_n(\mathbf{x})$ which in turn gives a less smooth posterior estimate. A good choice for $\sigma_n$ then is for it to be as large as possible but satisfy the angle criterion. From equation (25) this turns out to be,

$$\sigma_n = \kappa\|\mathbf{x}_n - \mathbf{u}_{nr}\| \tag{28}$$

where

$$\mathbf{u}_{nr} = \arg\min_{\mathbf{u}_k} ||\mathbf{x}_n - \mathbf{u}_k|| \tag{29}$$

is the nearest GaRBF unit centre to $\mathbf{x}_n$ in $\mathcal{D}$, and

$$\kappa = \frac{1}{\sqrt{2\log(1/\cos^2\Omega_{\min})}} \tag{30}$$

If $\Omega_{\min}$ is decreased, allowing more overlap between the two basis functions, $\kappa$ is increased. The addition of the basis function centred on the input pattern has an analogy to placing marbles inside a restricted space such as a cube. The minimum distance criterion ensures that the marbles are of a particular radius. Irrespective of the distribution of the input patterns there is a limit on the number of such marbles that can be placed inside a finite volume cube.

The network now adds a hidden unit only if the prediction error criterion (equation (22)) and the *distance criterion* (equation (26)) are both satisfied. These growth criteria are the same as those for the RAN. What is different is how it is arrived at in this paper, where a function space approach is adopted.

[5]No two GaRBFs are completely orthogonal to each other, except in the limit of the widths $\sigma_k$ approaching 0 or $\infty$. No single GaRBF with unique parameters is a linear combination of any other GaRBFs, except in the limit of infinite number of GaRBFs.

# 5 The Resource Allocating Network

The resource allocating network (RAN) was developed as a means to overcome the problem of NP-completeness in learning with fixed size networks [12]. Its motivation was the fact that by allocating new resources, learning could be achieved in polynomial time. Platt views the task of the RAN as combining memorization with adaptation [13], in which memorization is achieved by storing the input – output observation such as in Parzen window and k-nearest neighbour methods. He improves upon these methods by storing fewer observations which grow sublinearly and eventually saturate. The RAN finds an appropriate network (or size) for interpolating the given data, whereas in using a fixed size network either a smaller network that does not interpolate well or a larger network that overfits and generalise poorly could be encountered.

The RAN is a single hidden layer network whose output response to an input pattern is a linear combination of the hidden unit responses, given by,

$$f(\mathbf{x}) = \alpha_0 + \sum_{k=1}^{K} \alpha_k \phi_k(\mathbf{x}) \tag{31}$$

where $\phi_k(\mathbf{x})$ are the responses of the hidden units to an input $\mathbf{x}$. The coefficients $\ldots, \alpha_k, \ldots$ are the weights of the hidden to output layer and $\alpha_0$ is the bias term. The RAN hidden unit responses are given by,

$$\phi_k(\mathbf{x}) = \exp\left\{ -\frac{1}{\sigma_k^2} \|\mathbf{x} - \mathbf{u}_k\|^2 \right\} \tag{32}$$

where $\mathbf{u}_k$ is the unit centre or mean of the Gaussian and $\sigma_k$ is the spread of the neighbourhood or width of the Gaussian. The network is essentially a GaRBF network, except for the term $\alpha_0$. Platt describes the operation of a hidden unit as storing a local region in the input space – the neighbourhood of $\mathbf{u}_k$. Hence, the $\mathbf{u}_k$ are viewed as stored patterns. The weights of the hidden – output layer, coefficients $\alpha_k$, define the contribution of each hidden unit to a particular output.

The network begins with no hidden units. The first observation $(\mathbf{x}_0, y_0)$, where $y_0$ is the target output, is used in intialising the coefficient $\alpha_0 = y_0$. As observations are received the network grows by storing some of them by adding new hidden units. The decision to store an observation $(\mathbf{x}_n, y_n)$ depends on its novelty, for which the following two conditions must be met:

$$\|\mathbf{x}_n - \mathbf{u}_{nr}\| \quad > \quad \epsilon_n \tag{33}$$

$$e_n = y_n - f(\mathbf{x}_n) \quad > \quad e_{\min} \tag{34}$$

where $\mathbf{u}_{nr}$ is the nearest stored pattern to $\mathbf{x}_n$ in the input space and $\epsilon_n$, $e_{\min}$ are thresholds. The first criterion says that the input must be far away from stored patterns and the second criterion says that the error in the network output to that of the target must be significant. The value $e_{\min}$ is chosen to represent the desired accuracy of the network output. The distance $\epsilon_n$ represents the scale of resolution in the input space.

When a new hidden unit is added to the network, the parameters or weights associated with this unit are assigned as follows:

$$\alpha_{K+1} = e_n \tag{35}$$

$$\mathbf{u}_{K+1} = \mathbf{x}_n \tag{36}$$

$$\sigma_{K+1} = \kappa \|\mathbf{x}_n - \mathbf{u}_{nr}\| \tag{37}$$

where $\kappa$ is an overlap factor which determines the overlap of the responses of the hidden units in the input space. The value for the width $\sigma_{K+1}$ is based on a nearest neighbour heuristic.

When the observation $(\mathbf{x}_n, y_n)$ does not satisfy the novelty criteria, the LMS algorithm is used to adapt the network parameters $\mathbf{w} = [\alpha_0, \ldots, \alpha_K, \mathbf{u}_1^T, \ldots, \mathbf{u}_K^T]^T$, given by,

$$\mathbf{w}^{(n)} = \mathbf{w}^{(n-1)} + \eta e_n \mathbf{a}_n \tag{38}$$

where $\eta$ is the adaptation step size and $\mathbf{a}_n = \nabla_w f(\mathbf{x}_n)$ is the gradient of the function $f(\mathbf{x})$ with respect to the parameter vector $\mathbf{w}$ evaluated at $\mathbf{w}^{(n-1)}$. Hence,

$$\mathbf{a}_n = \left[1, \phi_1(\mathbf{x}_n), \ldots, \phi_K(\mathbf{x}_n), \phi_1(\mathbf{x}_n)\frac{2\alpha_1}{\sigma_1^2}(\mathbf{x}_n - \mathbf{u}_1)^T, \ldots, \phi_K(\mathbf{x}_n)\frac{2\alpha_K}{\sigma_K^2}(\mathbf{x}_n - \mathbf{u}_K)^T\right]^T \tag{39}$$

The RAN begins with $\epsilon_n = \epsilon_{\max}$, the largest scale of interest, typically the size of the entire input space of non-zero probability density. The distance $\epsilon_n$ is decayed exponentially as,

$$\epsilon_n = \max\{\epsilon_{\max}\gamma^n, \epsilon_{\min}\} \tag{40}$$

where $0 < \gamma < 1$ is a decay constant. The value for $\epsilon_n$ is decayed until it reaches $\epsilon_{\min}$.

Platt showed that the complexity of the RAN was smaller than that of fixed size networks in achieving a given degree of approximation [12]. The advantages of the RAN are that it learns quickly, accurately and forms a compact representation. However, the growth pattern of the RAN depends critically on $\gamma$ which influences the rate of growth and on $e_{\min}$ which determines the final size of the network together with $\epsilon_{\min}$. These parameters have to be chosen *a priori* and hence the performance of the RAN depends crucially on their appropriate selection. The effect of using the LMS algorithm for adaptation is likely to result in slow convergence than if, say, an algorithm that attempts to obtain an optimal sequential estimate is used.

The RAN is described mathematically as follows:

- The network output $f(\mathbf{x})$ to the input $\mathbf{x}$ is given by,

$$f(\mathbf{x}) = \alpha_0 + \sum_{k=1}^{K} \alpha_k \phi_k(\mathbf{x})$$

$$\phi_k(\mathbf{x}) = \exp\left\{-\frac{1}{\sigma_k^2}\|\mathbf{x} - \mathbf{u}_k\|^2\right\}$$

- $\epsilon_n = \epsilon_{\max} \qquad \alpha_0 = y_0$.
- For each observation $(\mathbf{x}_n, y_n)$,
  - $\epsilon_n = \max\{\epsilon_{\max}\gamma^n, \epsilon_{\min}\}$
  - $e_n = y_n - f(\mathbf{x}_n)$
  - If

$$e_n > e_{\min} \qquad \& \qquad \|\mathbf{x}_n - \mathbf{u}_{nr}\| > \epsilon_n$$

    * Allocate a new hidden unit with,

$$\begin{aligned} \alpha_{K+1} &= e_n \\ \mathbf{u}_{K+1} &= \mathbf{x}_n \\ \sigma_{K+1} &= \kappa\|\mathbf{x}_n - \mathbf{u}_{nr}\| \end{aligned}$$

  - Else
    * $\mathbf{w}^{(n)} = \mathbf{w}^{(n-1)} + \eta e_n \mathbf{a}_n$

The description of the RAN, shown above, has mostly been derived from the function space approach in sections 3 and 4. There are differences however, between this derivation and the RAN as specified by Platt. Firstly, the term $\alpha_0$ does not appear in our solution. This difference may be neglected in view of the universal approximation properties of the ANN.

Secondly and importantly, the threshold on the distance criterion of the RAN, $\epsilon_n$ is reduced gradually until it reaches a minimum allowed value. The distance criterion $\epsilon_n$ provides a lower bound on the width $\sigma_n$ (from equations (26) and (28)), $\sigma_n > \kappa\epsilon_n$. The lower bound $\epsilon_{\min}$ on $\epsilon_n$ then gives,

$$\sigma_k > \kappa\ \epsilon_{\min} \tag{41}$$

a lower bound on the width of all the basis functions $\phi_k$. This ensures a limit on the smoothness of the basis functions preventing a noisy fit to the data.

The exponential decaying of the distance criterion allows fewer basis functions with large widths (smoother basis functions) initially and with increasing number of observations, more basis functions with smaller widths are allocated to fine tune the approximation. Since $\kappa$ is fixed, the minimum angle $\Omega_{\min}$ also remains unchanged and hence the near orthogonality condition is maintained. However, the lowering of $\epsilon_n$ is possible only with the simultaneous lowering of $\sigma_n$ achieved by equation (28).

Finally, we have not discussed how to adapt the parameters of the network when a hidden unit is not added. The RAN adapts the coefficients $\alpha_k$ and the hidden unit centres $\mathbf{u}_k$ when it decides not to add a hidden unit. The adaptation of the parameters $\mathbf{u}_k$ amounts to the rotation of the subspace $\mathcal{H}_K$. The function space interpretation given to the architecture and the growth criteria of the RAN suggests the use of an algorithm based on the same approach, the $\mathcal{F}$-Projections algorithm [5], [9]. We shall see in the next section how this leads to an enhancement of the RAN.

## 6 An Enhanced RAN

For the sequential learning problem, the principle of $\mathcal{F}$-Projection gives the optimal posterior estimate of an underlying function, given its prior estimate and a new observation [5]. An extension of the $\mathcal{F}$-Projections algorithm is the recursive nonlinear least squares (RNLS) algorithm [5], [7] in which the distribution of the previous input patterns is also recursively estimated. The RNLS estimate is obtained by minimising the cost function

$$\int_{\mathbf{x}\in\mathcal{D}} |f^{(n)}(\mathbf{x}) - f^{(n-1)}(\mathbf{x})|^2 p^{(n-1)}(\mathbf{x}) d\mathbf{x} \quad + \quad \frac{1}{n-1}|y_n - f^{(n)}(\mathbf{x}_n)|^2 \tag{42}$$

where $p^{(n-1)}(\mathbf{x})$ is the probability distribution of the past $(n-1)$ input observations. Solving the above minimisation is numerically intensive, but it can be approximated to give the well-known extended Kalman filter (EKF)[6] algorithm [5]. Having shown that the principle of $\mathcal{F}$-Projection formed the basis for the RAN, the enhancement we suggest here is to use the EKF algorithm in place of the LMS algorithm. This enhancement, first proposed in [5] and used in [10], improves the rate of convergence of the RAN and results in a network with smaller complexity. We shall refer to this enhanced network as RAN-EKF. A similar approach was (independently) proposed in [1] where they use the RLS algorithm for the growing multilayer perceptron (or back-propagation) network.

Given a parameter vector $\mathbf{w}$, the EKF algorithm obtains the posterior estimate $\mathbf{w}^{(n)}$ from its prior estimate $\mathbf{w}^{(n-1)}$ and its prior error covariance estimate $\mathbf{P}_{n-1}$ as follows (see [2]):

$$\mathbf{w}^{(n)} = \mathbf{w}^{(n-1)} + e_n \mathbf{k}_n \tag{43}$$

where $\mathbf{k}_n$ is the Kalman gain vector given by,

$$\mathbf{k}_n = \left[R_n + \mathbf{a}_n^T \mathbf{P}_{n-1} \mathbf{a}_n\right]^{-1} \mathbf{P}_{n-1}\mathbf{a}_n \tag{44}$$

where $\mathbf{a}_n$ is the gradient vector and $R_n$ is the variance of the measurement noise. The error covariance matrix is updated by,

$$\mathbf{P}_n = \left[\mathbf{I} - \mathbf{k}_n \mathbf{a}_n^T\right] \mathbf{P}_{n-1} \tag{45}$$

$\mathbf{I}$ being the identity matrix. Note that we are now adapting the parameters $\sigma_1, \ldots, \sigma_K$ as well and hence are included in the parameter vector $\mathbf{w}$.

The rapid convergence of the EKF algorithm may prevent the model from adapting to future data. To avoid this problem, a random walk model is often used [16] where the covariance matrix update becomes

$$\mathbf{P}_n = \left[\mathbf{I} - \mathbf{k}_n \mathbf{a}_n^T\right] \mathbf{P}_{n-1} + Q_0 \mathbf{I} \tag{46}$$

The parameter $Q_0$ is a scalar which determines the allowed random step in the direction of the gradient vector. The error covariance matrix $\mathbf{P}_n$ is a $P \times P$ positive definite symmetric matrix, where $P$ is the number of parameters being adapted. Whenever a new hidden unit

[6] For linear stationary models, EKF algorithm is equivalent to the weighted recursive least squares (RLS) algorithm [2].

is allocated the dimensionality of $\mathbf{P}_n$ increases and hence, the new rows and columns must be initialised. Since $\mathbf{P}_n$ is an estimate of the error covariance of the parameters, we choose,

$$\mathbf{P}_n = \begin{pmatrix} \mathbf{P}_{n-1} & 0 \\ 0 & P_0\mathbf{I} \end{pmatrix} \tag{47}$$

where $P_0$ is an estimate of the uncertainty in the initial values assigned to the parameters, which in our case is also the variance of the observations $\mathbf{x}_n$ and $y_n$. The dimension of the identity matrix $\mathbf{I}$ is equal to the number of new parameters introduced by the addition of a new hidden unit. The above equation combined with the EKF algorithm is used in place of the LMS algorithm in the RAN-EKF.

The RAN, with its LMS adaptation is considerably faster to implement than the relatively more complex (computationally) RAN-EKF. However, the EKF algorithm can be implemented as a fast transversal filter algorithm [1] to increase the speed and hence its capability to learn on-line in real-time.

## 7 Experimental Results

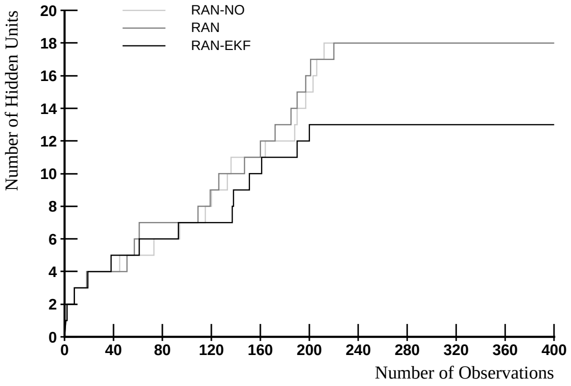
The RAN was developed as a model for solving sequential function interpolation problems. It is also suitable for off-line (or block) learning, where all the data are available en-bloc and presented one by one. Two types of problems are presented to the networks. The first is a synthetic function approximation problem where the data are a randomly sampled data consistent with an underlying function and presented one by one cyclically. The second is a problem of on-line prediction of a chaotic time-series where an underlying function does not exist. The variation of the performance measures are shown with increasing time in order to illustrate the effect of the evolving networks. The RAN used in these experiments did not have the $\alpha_0$ coefficient and hence differed only in the algorithm used for adaptation.

The underlying function that needs to be approximated in the first experiment is the Hermite polynomial (used in [11]),

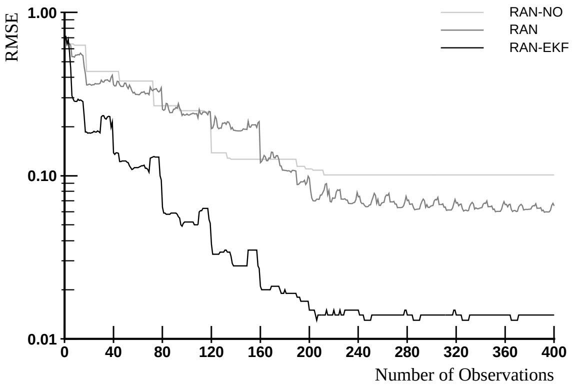
$$f_*(x) = 1.1(1 - x + 2x^2)\exp\left\{-\tfrac{1}{2}x^2\right\} \tag{48}$$

where $x \in \Re$. A random sampling of the interval $[-4, +4]$ is used in obtaining the 40 input – output data for the training set. The approximation error is calculated from 200 uniformly sampled data in the same interval.

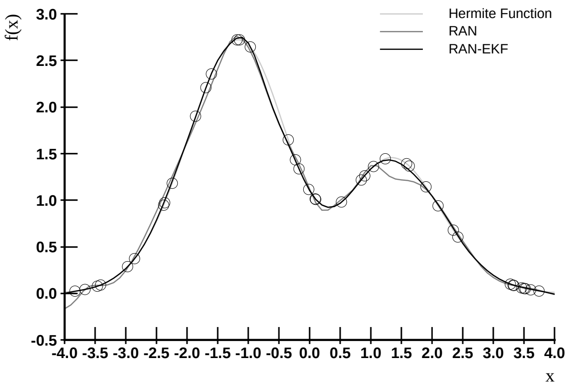
The parameter values used in this experiment are as follows: $\epsilon_{\max} = 2.0$, $\epsilon_{\min} = 0.2$, $\gamma = 0.977$, $e_{\min} = 0.02$, $\kappa = 0.87$, $\eta = 0.05$, $P_0 = R_n = 1.0$ and $Q_0 = 0.02$. The performance of the different RANs are shown in figure 3. The RAN-EKF is the enhanced network and RAN-NO is a network that adds hidden units according to the growth criteria but does not adapt the parameters when a unit is not added.




(a) Growth Pattern



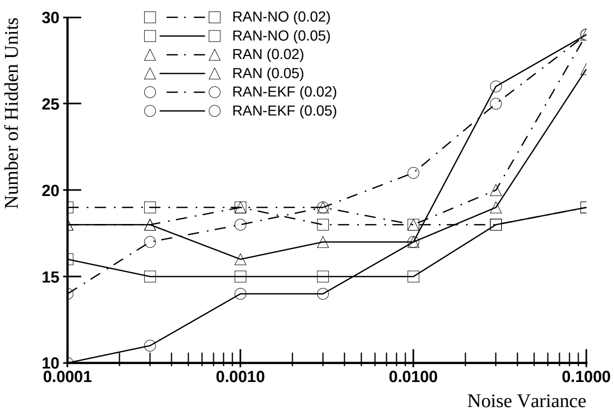

(b) RMSE -- Approximation Error



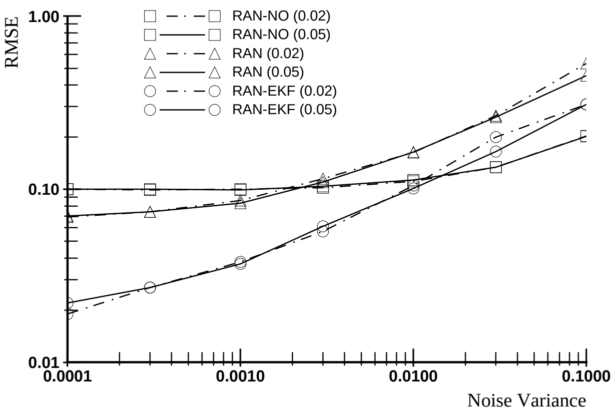

(c) The hermite function and the RAN approximations

Figure 3: The performance of the RAN versions on the Hermite function approximation problem. The circles represent the 40 input – output observations.

The growth pattern of all three networks were similar in the initial stages, suggesting that the rate of growth is independent of the adaptation procedure and depends only on the decay rate of $\epsilon_n$. In the latter stages though, the faster convergence of the RAN-EKF prevents further addition of hidden units by reducing the interpolation errors below the critical value $e_{\min}$. The approximation error, measured by the root mean squared error (RMSE) (in figure 1(B)) clearly illustrates the faster convergence achieved by the enhanced RAN. The final RMSE value of around 0.07 for the RAN was achieved in just 80 iterations by the RAN-EKF with only 6 hidden units compared to the 18 in the RAN. The enhanced network not only was least complex, but was also accurate by nearly an order below that achieved by the RAN. Note also that the performance of RAN-NO was comparable to the RAN, demonstrating the power behind the appropriate allocation of hidden units. The approximation shown in figure 3(c) shows that the RAN does not interpolate as smoothly as the RAN-EKF does, also observed in [5].




(a) Network size (hidden units) Vs Noise Variance




(b) RMS Approximation Error Vs Noise Variance

Figure 4: The effect of noise in observations for the approximation problem. The numbers inside the brackets in the key are the values of $e_{\min}$ used.

The effects of noise were analysed by adding varying levels of Gaussian noise to the training patterns. The size of the network and the approximation error for different noise levels for the RAN versions are shown in figure 4. It is apparent from the figure that the value of $e_{\min}$ at the lower ranges such as 0.02 and 0.05, chosen to represent the desired accuaracy, does not affect the goodness of approximation for any level of noise. At low levels of noise, a higher threshold $e_{\min}$ results in a smaller size network, the difference disappearing with increasing levels of noise. The RAN-EKF was able to consistently form a compact size than RAN while achieving better performance. At high noise levels, adapting the means $\mathbf{u}_k$ seemed to not only increase the size of the network but also resulted in poor approximation, demonstrated by the performance of RAN-NO.

In the second experiment, the chaotic time-series being predicted is the Mackey-Glass series which is governed by the following differential delay equation:

$$\frac{ds(t)}{dt} = -bs(t) + a\frac{s(t-\tau)}{1+s(t-\tau)^{10}} \tag{49}$$

with $a = 0.2$, $b = 0.1$ and $\tau = 17$. A sampled version of the above series is obtained by integrating the equation, smoothing and sampling it. The training data is obtained from a series of 5000 samples and the test data of 1000 samples from the same series but into the future.

The series is predicted $\nu = 50$ sample steps ahead using four past samples, namely $s_n, s_{n-6}, s_{n-12}, s_{n-18}$. Hence, the $n^{th}$ input – output data for the network to learn is

$$\begin{aligned} \mathbf{x}_n &= [s_{n-\nu}, s_{n-\nu-6}, s_{n-\nu-12}, s_{n-\nu-18}]^T & (50)\\ y_n &= s_n & (51) \end{aligned}$$

whereas the step ahead predicted value at time $(n)$ is given by,

$$z_{n+\nu} = f^{(n)}(\mathbf{x}_{n+\nu}) \tag{52}$$

where $f^{(n)}$ is the network at time $(n)$. The step ahead prediction error is given by,

$$\varepsilon_{n+\nu} = s_{n+\nu} - z_{n+\nu} \tag{53}$$

The error in the predicted value of $s_{n+\nu}$ becomes available to the network only after $\nu$ steps. The data is received sequentially and hence requires the storage of all samples from $s_{n-\nu-18}$ to $s_n$ and from $z_n$ to $z_{n+\nu}$.

The usual performance measure given for prediction has been the normalised RMSE on the test data [12]. For this problem, we have used the normalised RMSE as the approximation performance measure, where the normalisation is with respect to the variance of the time-series. This measures the error in an underlying mapping that may exist between $\mathbf{x}$ and $y$ or the average prediction error if the network does not adapt to future data. If such a mapping does not exist, this measure does not represent the prediction performance since the network continually adapts to incoming data and has the capacity to modify its mapping appropriately. A suitable measure is the exponentially weighted prediction error (WPE) which at time $(n)$ can be recursively computed as,

$$\mathrm{WPE}^{(n)} = \lambda\ \mathrm{WPE}^{(n-1)} + (1-\lambda)\,|\varepsilon_n|^2 \tag{54}$$

for some $0 < \lambda < 1$. In the results shown $\lambda = 0.95$.

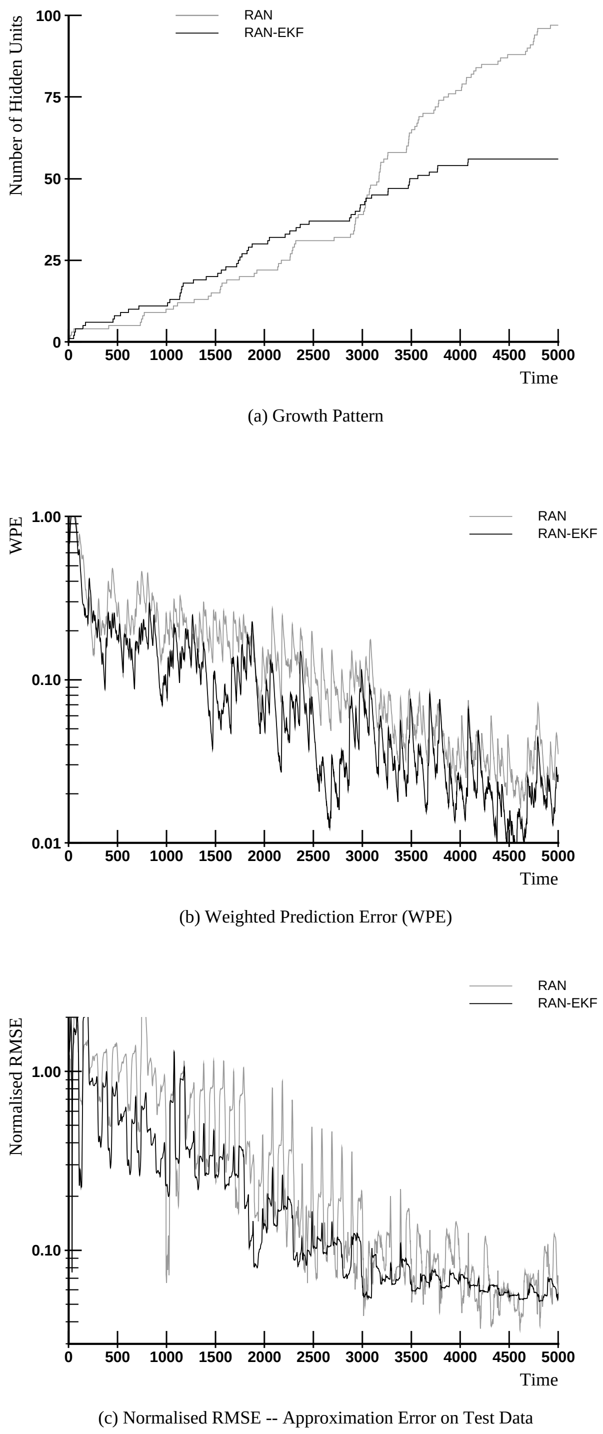



Figure 5: The performance of the RAN versions on the Mackey-Glass time series prediction problem.

The parameter values used in this experiment are as follows: $\epsilon_{\max} = 0.7$, $\epsilon_{\min} = 0.07$, $\gamma = 0.999$, $e_{\min} = 0.05$, $\kappa = 0.87$, $\eta = 0.02$, $P_0 = R_n = 1.0$ and $Q_0 = 0.0002$. The performance of the RAN and the enhanced network are shown in figure 5.

The growth pattern of the RAN and the RAN-EKF showed similar patterns to those observed in the first experiment. The growth rates were initially similar and the RAN-EKF levelled off quickly in the latter stages. The on-line prediction performance measured by the WPE shows that the RAN-EKF is consistently better than the RAN. On average, the WPE for the RAN-EKF was a factor 2 smaller than that of the RAN. The final value of WPE for RAN and RAN-EKF are 0.035 and 0.024 respectively. The normalised RMSE which measures the average prediction performance on a test data also shows the superiority of the RAN-EKF even when its complexity is smaller than that of the RAN. The final value for the normalised RMSE of 0.056 for RAN-EKF with 56 hidden units compares well with 0.063 for the RAN and Platt's results [12] for the RAN of 0.054 achieved with 143 hidden units. However, such a number can be misleading since it depends on choosing the appropriate time in the oscillatory performance measure as shown in figure 5. What the results demonstrate is that the enhanced network performed better in both the WPE and the RMSE, than the RAN with fewer hidden units. Platt had already shown that the RAN used fewer hidden units than fixed size networks while achiving better performance [12]. The enhancement suggested here has led us towards the minimal network that would be required in a sequential learning problem.

## 8 Conclusion

We have adopted the function estimation approach to optimal sequential learning with neural networks — a natural framework for approximation with ANN. The sub-optimal solution that resulted from this approach was shown to be equivalent to that of the RAN. Thus this work also provides an alternative interpretation to the RAN in contrast to that of Platt whose description was based on the RAN's similarity to the Parzen window and nearest neighbour methods. We showed that the new basis functions (hidden units) are assigned only if they form a significant angle in the function space to the existing basis functions. This ensures the efficient use of the basis functions and their allocation only according to the complexity of the underlying function to be mapped.

We have also proposed an enhancement to the RAN as a result of the function space interpretation given to its architecture and the growth criteria. This enhanced network uses the EKF algorithm to adapt the parameters when a hidden unit is not added. The superior performance of the enhanced network over the RAN was demonstrated in function approximation and time-series prediction tasks. The resulting network complexity is smaller while the approximation and prediction accuracy is higher. The initial convergence to the solution was also faster with the enhanced network. Results on a multi-class classification problem are given in [8] which further reinforces the above conclusion. This achievement is at the expense of added computational complexity which may be compensated either with systolic array or fast transversal filter implementations. The optimal sequential learning approach has taken us further towards attaining the minimal network that would be required for a given problem.

## References


[1] M. R. Azimi-Sadjadi and S. Sheedvash. Recursive node creation in back-propagation neural networks using orthogonal projection method. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, Toronto, 1991.

[2] J. V. Candy. *Signal processing: The model-based approach.* McGraw-Hill, New York, 1986.

[3] R. O. Duda and P. E. Hart. *Pattern classification and scene analysis.* John Wiley, New York, 1973.

[4] Goodwin and K. S. Sin. *Adaptive filtering prediction and control.* Prentice-Hall, New Jersey, 1984.

[5] V. Kadirkamanathan. *Sequential learning in artificial neural networks.* PhD Thesis, Cambridge University Engineering Department, 1991.

[6] V. Kadirkamanathan and F. Fallside. $\mathcal{F}$-Projection: A nonlinear recursive estimation algorithm for neural networks. Technical Report CUED/F-INFENG/TR.53, Cambridge University Engineering Department, 1990.

[7] V. Kadirkamanathan and M. Niranjan. Nonlinear adaptive filtering in nonstationary environments. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, Toronto, 1991.

[8] V. Kadirkamanathan and M. Niranjan. Application of an architecturally dynamic network for speech pattern classification. *Proceedings of the Institute of Acoustics*, Volume 14, 1992.

[9] V. Kadirkamanathan, M. Niranjan and F. Fallside. Sequential adaptation of radial basis function neural networks and its application to time-series prediction. In R. P. Lippmann, J. E. Moody and D. S. Touretzky, eds., *Neural Information Processing Systems 3.* Morgan Kaufmann, 1991.

[10] V. Kadirkamanathan, M. Niranjan and F. Fallside. Models of dynamic complexity for time-series prediction. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, San Francisco, 1992.

[11] D. J. C. MacKay. Bayesian interpolation. *Neural Computation*, 4(3), 415-447, 1992.

[12] J. C. Platt. A resource allocating network for function interpolation. *Neural Computation*, 3(2), 213-225, 1991.

[13] J. C. Platt. Learning by combining memorization and gradient descent. In R. P. Lippmann, J. E. Moody and D. S. Touretzky, eds., *Neural Information Processing Systems 3.* Morgan Kaufmann, 1991.

[14] D. L. Reilly, L. N. Cooper and C. Elbaum. A neural model for category learning *Biological Cybernetics*, 45:35–41, 1982.

[15] H. White. Connectionist nonparametric regression: Multilayer feedforward networks can learn arbitrary mappings. *Neural Networks*, 3, 535-544, 1990.

[16] P. C. Young. *Recursive estimation and time-series analysis.* Springer-Verlag, Berlin, 1984.