Recursively Adapted Radial Basis Function Networks and its Relationship to Resource Allocating Networks and Online Kernel Learning

Wei-Feng Liu, Puskal P. Pokharel, and Jose C. Principe, Fellow, IEEE

Abstract—This paper proposes a recursively adapted radial basis function network and provides additional insights into several well-known techniques such as radial basis function networks, resource allocating networks and stochastic gradient descent in reproducing kernel Hilbert spaces. Through this perspective, resource allocating networks are investigated in a more principled way so that issues of convergence and generalization can be mathematically analyzed in the least mean square framework.

I. INTRODUCTION

Resource allocating networks (RAN) are an intuitive, very appealing idea introduced by Platt [1] but its convergence and generalization have not been fully addressed mathematically.

Recently, Kivinen, Smola and Williamson [2] introduced an online kernel learning method (OKL) which can be simply viewed as a stochastic gradient descent method applied on the regularized cost function of a radial basis function network (RBFN). Interestingly, OKL bears much resemblance with RAN in terms of learning strategy and network architecture. However, there is one fundamental difference. OKL employs a forgetting factor in its training, which is the legacy of the regularization term in the RBFN cost function, to impose stability in learning and thus ensure generalization. Further statistical analysis concludes that this forgetting factor appears essential for its well-posedness [3]. Therefore, this reasoning raises the question if RAN is well-posed.

Generalization has been substantially studied in the framework of regularization theories [4], [5]. Regularization as a remedy for ill-posedness became widely known due to the work of Tikhonov [6]. In solving RBFN like problems, the Tikhonov regularization is essentially a trade-off between fitting training data and reducing solution norms. Consequently it reduces the sensitivity of the solution to small changes in the training data and imposes stability on the ill-posed problem.

Our work improves on related research in several ways:
1. We propose a recursively adapted RBFN and provide the convergence conditions.
2. A generalized recursively adapted RBFN learning framework is presented.
3. We introduce a least mean square (LMS) formulation for a very special case of RA-RBFN and a simplified RAN—called kernel least mean square (KLMS).
4. We address the well-posedness of KLMS mathematically based on the small-step-size theory and H∞ stable theorem.
5. Comparisons of the KLMS, RAN and OKL are provided thus additional understanding of RAN and OKL is obtained through the analysis of the KLMS.

II. RECURSIVELY ADAPTED RBF NETWORK

A. Learning Problem

Suppose the goal is to learn a function \( f: U \rightarrow \mathbb{R} \) based on a sequence \((u_i, y_i)\) of input-output examples \((u_i, y_i) \in Z = U \times Y\). \( Y \) is a compact subset of \( \mathbb{R} \) and \( U \) a compact subspace of \( \mathbb{R}^m \). A quadratic loss is chosen to measure the deviation of \( f \) from the observations. The problem statement is to find a function \( f \) in a hypothesis space \( H \), such that the following empirical risk is minimized

\[
R_{emp}(f) = \sum_{i=1}^{N} (y_i - f(u_i))^2
\]

B. Standard RBF networks

We will focus on the standard RBFN to ease the analysis. The regularization theory says if the hypothesis space is too large, instead of solving the optimization (1) a more meaningful \( f \) can be found by minimizing the following regularized empirical risk:

\[
\min_{f} R_{reg}(f) = \sum_{i=1}^{N} (y_i - f(u_i))^2 + \lambda \| f \|_H
\]

where \( \lambda \) is the regularization parameter. The well-known solution to (2) is [7]

\[
f_{reg} = \sum_{i=1}^{N} \alpha_i \kappa(u_i)\]

where the coefficients \( \alpha = (\alpha_1, \ldots, \alpha_N) \) satisfy the following linear equation

\[
(G + \lambda I) \alpha = \tilde{y}
\]

\( G \) is the Gram matrix and \( \tilde{y} = (y_1, \ldots, y_N) \).

The kernel \( \kappa: U \times U \rightarrow \mathbb{R} \) is a continuous, symmetric, positive-definite function [8]. The commonly used kernel
and also the only one used in this paper is the Gaussian kernel (where \( a \) is a kernel size related parameter)

\[
\kappa(u_i, u_j) = \exp(-a \|u_i - u_j\|^2).
\]

(5)

It can be seen that the RBF network boils down to a matrix inversion problem.

C. Recursively Adapted RBFN

Intuitively learning should be a step-by-step improvement process. This intuition motivates us to investigate if the RBFN has other interpretations than a matrix inversion.

**Theorem 1**: The RBFN solution (3) with nonzero \( \lambda \) can be expressed in the following way

\[
f_{\text{RBF}} = \frac{1}{\lambda} \sum_{i=1}^{N} e^i(i)\kappa(u_i, \cdot)
\]

(6)

where \( e^i(i) = y_i - f_{\text{RBF}}(u_i) \) is the final 'optimal' error on the \( i^{th} \) training example.

**Proof**: Taking the derivative of (2) with respect to \( f \) in the reproducing kernel Hilbert space (RKHS) yields directly [2]

\[
\frac{\partial}{\partial f} R_x[f, z^N] = \sum_{i=1}^{N} -2(y_i - f(u_i))\kappa(u_i, \cdot) + 2\lambda f
\]

(7)

Equating this to zero, we have

\[
\lambda f = \sum_{i=1}^{N}(y_i - f(u_i))\kappa(u_i, \cdot)
\]

(8)

Since (3) is a minimizer on the open set, (3) should be a stationary point, i.e., satisfies (8). Therefore

\[
\lambda f_{\text{RBF}} = \sum_{i=1}^{N} e^i(i)\kappa(u_i, \cdot)
\]

(9)

If \( \lambda \neq 0 \), we obtain

\[
f_{\text{RBF}} = \frac{1}{\lambda} \sum_{i=1}^{N} e^i(i)\kappa(u_i, \cdot)
\]

(10)

On the other hand, we have to show that solution (3) is the only one satisfying (8).

Denote \( \tilde{f} = [f(u_1), f(u_2), \ldots, f(u_N)]^T \). By (8), we have

\[
\lambda \tilde{f} = G(\tilde{y} - \tilde{f})
\]

(11)

or equivalently,

\[
\tilde{f} = (G + \lambda \mathbb{I})^{-1}G\tilde{y}
\]

(12)

Notice that \((G + \lambda \mathbb{I})^{-1}\) is interchangeable with \( G \), so by (3) and (4) we have

\[
\tilde{f} = G(G + \lambda \mathbb{I})^{-1}\tilde{y} = G\tilde{a} = f_{\text{RBF}}
\]

(13)

That means if any function satisfies (8), it equals \( f_{\text{RBF}} \) as defined in (3) at every training point \( u_i \). Therefore by (10)

\[
f = \lambda^{-1}\sum_{i=1}^{N}(y_i - f(u_i))\kappa(u_i, \cdot) = \lambda^{-1}\sum_{i=1}^{N}(y_i - f_{\text{RBF}}(u_i))\kappa(u_i, \cdot)
\]

(14)

\[
= \lambda^{-1}\sum_{i=1}^{N} e^i(i)\kappa(u_i, \cdot) = f_{\text{RBF}}
\]

To appreciate this result more, let’s take the linear kernel, i.e.

\[
\kappa(u_i, u_j) = u_i^T u_j
\]

If \( \lambda = 0 \) here, (8) becomes

\[
\sum_{i=1}^{N} e^i(i)\kappa(u_i, \cdot) = \sum_{i=1}^{N} e^i(i)u_i = 0
\]

(16)

which is the famous ‘principle of orthogonality’. If \( \lambda \neq 0 \), the principle of orthogonality does not hold and generally we have

\[
\sum_{i=1}^{N} e^i(i)u_i = \lambda f
\]

(17)

Theorem 1 reveals something interesting—that with fixed \( \lambda \) the coefficients of the optimal solution on the kernel basis are directly proportional to the corresponding errors and inversely proportional to the regularization parameter.

Furthermore it gives us hope that it may be possible to iteratively estimate \( f \) by the following fixed point algorithm:

\[
f^t = \eta \sum_{i=1}^{N} (y_i - f^{t-1}(u_i))\kappa(u_i, \cdot)
\]

(18)

**Algorithm 1** Recursively Adapted RBF Networks

**Input**: data \((u, y)\), size \( N \)

**Initialization**

\[
f^0 = 0, \quad \eta: \text{learning step}, \quad a: \text{kernel width parameter}
\]

**loop over convergence**

\[
\begin{array}{l}
\text{evaluate output of network at every point } f^{t-1}(u_i) \\
\text{computer error at every point } e^t(i) = y_i - f^{t-1}(u_i) \\
\text{update } f^t = \eta \sum_{i=1}^{N} e^t(i)\kappa(u_i, \cdot)
\end{array}
\]

An important question follows immediately: under what condition does this recursively adapted RBFN (RA-RBFN) converge.

**Theorem 2**: The RA-RBFN converges uniquely to (3) with \( \lambda = \eta^{-1} \) under the following conditions:

\[
\sup \{ |\kappa(u_i, u_j)|, 1 \leq i \leq N, \forall u \in U \} < \infty
\]

\[
\eta < 1/NK_0
\]

(19)

\[
K_0 := \max \{ |\kappa(u_i, u_j)|, 1 \leq i, j \leq N \}
\]

**Proof**: The proof is divided into two steps. First we show that \( f \) converges at data points \( u_j \). By (18), for any \( j \), we have

\[
f^t(u_j) = \eta \sum_{i=1}^{N} (y_i - f^{t-1}(u_i))\kappa(u_i, u_j)
\]

(20)

\[
f^{t-1}(u_j) = \eta \sum_{i=1}^{N} (y_i - f^{t-2}(u_i))\kappa(u_i, u_j)
\]

(21)

Subtracting (20) from (21), we get
\[ f'(u) = f^{-1}(u) = \eta \sum_{i=1}^{N} (f^{-2}(u) - f^{-3}(u)) \kappa(u, u_r) \quad (22) \]

Taking the absolute value on both sides and using the property of the absolute value yield

\[ |f'(u) - f^{-1}(u)| \leq \eta \sum_{i=1}^{N} |f^{-2}(u) - f^{-3}(u)| \| \kappa(u, u_r) \| \]

\[ \leq \eta K_0 \sum_{i=1}^{N} |f^{-2}(u) - f^{-3}(u)| \]

\[ \leq \sum_{i=1}^{N} |f^{-2}(u) - f^{-3}(u)| / N \]

Summing index \( j \) from 1 to \( N \) in (23), we obtain,

\[ \sum_{i=1}^{N} |f^{-2}(u) - f^{-3}(u)| < \sum_{i=1}^{N} |f^{-2}(u) - f^{-3}(u)| \]

(24)

It shows that the vectors \( [f'(u), \ldots, f'(u_N)] \) make a Cauchy sequence under the L1 metric. It is known that as long as the metric space is complete, every Cauchy sequence converges uniquely. And all the Hilbert spaces including \( R^N \) in this case are complete [9]. Thus \( [f'(u), \ldots, f'(u_N)] \) converges in the L1 norm sense as \( t \) goes to infinity, which implies every component in the vector converges.

Next, for any \( u \in U \), we have

\[ |f'(u) - f^{-1}(u)| \leq \eta \sum_{i=1}^{N} |f^{-2}(u) - f^{-3}(u)| \| \kappa(u, u_r) \| \]

(25)

Since \( \eta \) and \( |\kappa(u, u_r)| \) are bounded, \( |f^{-2}(u) - f^{-3}(u)| \) goes to zero as \( t \) goes to infinity, it follows \( f'(u) \) converges.

Let \( t \) go to infinity in (18),

\[ f^{-} = \eta \sum_{i=1}^{N} (y_i - f^{-}(u_i)) \kappa(u, u_r) \]

(26)

That means \( f^{-} \) satisfies (8) with \( \lambda = \eta^{-1} \). By theorem 1, \( f^{-} \) is the minimizer (3).

The condition (19) is sufficient but may not be necessary. The next theorem gives a sufficient and necessary condition.

**Theorem 3**: The sufficient and necessary condition for the RA-RBFN to converge uniquely to (3) is: \( \eta < 1/\sigma_1 \), where \( \sigma_1 \) is the largest eigenvalue of \( G \).

Proof: By the same argument in the proof of Theorem 2, it is enough to show that this is the sufficient and necessary condition for the convergence at the training data points.

Denote \( \tilde{f} = [f'(u_1), \ldots, f'(u_N)] \). Then (18) becomes

\[ \tilde{f} = \eta G (\tilde{y} - \tilde{f}^{-1}) \]

(27)

Denoting \( J = -\eta G \), we have

\[ \tilde{f}' = J \tilde{f}^{-1} - \tilde{y} \]

(28)

and further

\[ \tilde{f}' = J' \tilde{y} - (J' + J^{-1} + \ldots + J) \tilde{y} \]

(29)

For \( \tilde{f}' \) to converge, it requires that \( J' \) goes to 0 as \( t \) approaches infinity such that \( \tilde{f}' \) does not depend on the initial guess \( \tilde{f}^0 \) and the summation of geometric series converges. To achieve that, the absolute values of all the eigenvalues of \( J \) should not exceed unit (sufficient and necessary).

Assume the eigenvalues of \( G \) are \( \sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_N \geq 0 \), since \( G \) is positive semi-definite. By the discussion above, it requires

\[ |\eta \sigma_i| < 1 \]

(30)

Equivalently

\[ \eta < 1/\sigma_1 \]

(31)

Theorem 3 is stronger than Theorem 2 since \( NK_0 \) is greater than the trace of \( G \) thus the maximal eigenvalue.

The computational complexity of the RA-RBFN is \( O(N^3) \) which is very appealing comparing to the standard RBFN. However, the condition on the learning step constrains the choice of the regularization parameter which may impose a big limitation on its practical use.

But the learning process in the RA-RBFN is very interesting. It starts from an initial estimation \( f^{-1} \), then test this estimation on the training data for some deviation measures, after that the learning process improves its estimation as \( f' \) combining the previous estimation and the deviation just obtained. Actually improvement can be made not only by adapting the coefficients as in the RA-RBFN, we also can adapt the network structure as inspired by the RAN. Therefore, a more general version of algorithm 1 is the GRA-RBFN as follows:

**Algorithm 2** Generalized RA-RBFN (GRA-RBFN)

```
loop over input-output pairs (u, y)
{
    if the new data is novel
    {
        allocate a new unit
    }
    else
    {
        adjust the coefficients and parameters of the present network
    }
}
```

An input-output example is considered novel [1] if the input is far away from existing centers and if the difference between the desired output and the output of the network is large.

This learning framework is very general, and especially includes RAN as a special case. However, with simultaneous adaptation of the coefficients and the network structure it becomes mathematically intractable as exemplified by RAN. Another special case of the GRA-
RBFN is a network that grows but the coefficients are fixed one at a time.

III. KERNEL LEAST MEAN SQUARE

A. KLMS Learning Algorithm

For any kernel, such as the Gaussian kernel in (5), the Mercer’s theorem [8] states there exists a mapping \( \Phi \) such that

\[
\kappa(u_i, u_j) = \Phi(u_i)^\top \Phi(u_j), \quad \text{for } \forall u_i, u_j \in U. \tag{32}
\]

Usually \( \Phi \) is treated as a nonlinear mapping and \( \Phi(u) \) is the transformed feature vector lying in the feature space \( F \) (which is a Hilbert space). Equation (32) can be simply interpreted as the usual dot product in the vector space.

Denote the weight vector in \( F \) by \( \Omega \). Then the kernel least mean square (KLMS) is nothing but the least mean square (LMS) algorithm [10] performed on the transformed data sequence \( \{(\Phi(u_1), y_1), \ldots, (\Phi(u_N), y_N)\} \):

\[
\Omega_0 = 0
\]

\[
e_i^* = y_i - \Omega_{i-1}^\top \Phi(u_i)
\]

\[
\Omega_i = \Omega_{i-1} + \eta e_i^* \Phi(u_i)
\]

where \( \eta \) is the learning step and \( e_i^* \) is the apriori error since it is obtained only from the previous inputs. Normally, it is difficult to have direct access to the weight and the transformed data in the feature space and then the famed ‘kernel trick’ comes to rescue.

The repeated application of the weight-update equation yields

\[
\Omega_n = \eta \sum_{i=0}^n e_i^* \Phi(u_i). \tag{34}
\]

Therefore, after \( n \)-step training, the weight is expressed as the linear combination of the previous and present input data weighted by the apriori errors. More importantly, the input-output of this learning system can be solely expressed in terms of inner product, i.e.

\[
\Omega_n^\top \Phi(\tilde{u}) = \eta \sum_{i=1}^n e_i^* \kappa(u_i, \tilde{u}) \tag{35}
\]

and

\[
e_i^* = y_i - \eta \sum_{i=1}^n e_i^* \kappa(u_i, u_i) \tag{36}
\]

Denoting

\[
f^t_{\text{KLMS}}(\tilde{u}) = \eta \sum_{i=1}^t e_i^* \kappa(u_i, \tilde{u}) \tag{37}
\]

we may go on to express

\[
e_i^* = y_i - f^{t-1}_{\text{KLMS}}(u_i) \tag{38}
\]

\[
f^{t}_{\text{KLMS}}(\tilde{u}) = \eta e_i^* \kappa(u_i, \tilde{u}) + f^{t-1}_{\text{KLMS}}(\tilde{u}) \tag{39}
\]

Clearly it is seen that the KLMS allocates a new unit when a new training data comes in with the input \( u_i \) as the center and the difference between the output of the network and the desired output as the coefficient. After training, the final network architecture is

\[
f_{\text{KLMS}}(\tilde{u}) = \eta \sum_{i=0}^n e_i^* \kappa(u_i, \tilde{u}) \tag{40}
\]

The RAN, OKL and KLMS methods share the same learning strategy and same network architecture. But some differences should not be neglected. The KLMS and OKL have a learning step size while the RAN does not have. The RAN and OKL do not adapt the kernel parameters once it is initially computed whereas the RAN does. A fundamental difference between OKL with the other two is that the OKL has a forgetting factor which has been regarded as the key to the well-posedness and generalization of the OKL. However, we will point out in the next section that the KLMS algorithm is also well-posed even without a regularization parameter.

Algorithm 3 Kernel Least Mean Square

Input: data \( (u, y) \), size \( N \)

Initialization

\( f^0 = 0, \ \eta \) : learning step, \( a \) : kernel width parameter,

Novelty conditions

\( \varepsilon \) : error threshold, \( \delta \) distance threshold

Variables used in the loop

\( c_j, h_j \) : kernel centers and coefficients in \( f^{-1} \),

\( c_{\text{new}}, h_{\text{new}} \) : center of the new unit,

\( h_{\text{new}} \) : coefficient of the new unit

Loop over input-output pairs \( (u, y) \)

\[
\begin{cases}
\text{evaluate output of network } f^{-1}(u) \\
\text{computer error } e_i^* = y - f^{-1}(u) \\
\text{find distance to nearest center } d = \min_j \| c_j - u \|
\end{cases}
\]

if \( \| e_i^* \| > \varepsilon \) and \( d > \delta \)

\[
\begin{cases}
\text{allocate new unit, } c_{\text{new}} = u, \ h_{\text{new}} = \eta e_i^* \\
\text{ } \text{ } \quad f^t = h_{\text{new}} \kappa(u, c_{\text{new}}) + f^{t-1}
\end{cases}
\]

else

\( f^t = f^{t-1} \)

B. KLMS Learning Properties

In this section, we will set aside the novelty condition and treat the KLMS purely as the LMS method in the feature space to exploit a wealth of literature on LMS. Another important reason of doing so is that the LMS serves as the main common learning strategy among KLMS, RAN and OKL. By analyzing the KLMS, we gain additional understandings about RAN and OKL.

Assume the transformed data in the feature space satisfy the following multiple linear regression model

\[
y_n = (\Omega^\top \Phi(u_n) + \nu(n) \tag{41}
\]
where \( \Omega \) is the hypothesized model and \( vn \) is the model uncertainty.

Here we show that the norm of the solution in (40) is well upper-bounded.

**Theorem 4**: (\( H^\infty \) stable) Given training data \( \{\Phi(u), v\}_{i=1}^{N} \) that satisfy the linear regression model (41) for any unknown vector \( \Omega \) and finite energy noise sequence \( \{v(i)\} \) without any statistical assumption. The KLMS algorithm satisfies the following robustness condition

\[
\delta T_{jj} \leq \sum_{i=1}^{N} \left| e_i^* \right|^2 < \Omega + \sum_{j=1}^{N} \left| v(j) \right|^2
\]

if and only if the matrices \( \{\eta^{-1}I - \Phi(u_i)\Phi(u_i)^T\} \) are positive-definite for \( i = 1, 2, ..., N \). [10]

**Theorem 5**: Under the \( H^\infty \) stable condition, the norm of the apriori errors in the KLMS and further the norm of the KLMS solution in (35) are well upper-bounded.

Proof: First notice that

\[
e_i^* - v(i) = \Omega_i \sum_{j=1}^{N} e_j - v(j) < \left( \Omega^* \right)^T \Phi(u_i) - \Omega^* \Phi(u_i)
\]

Substituting (42) into (44), we have

\[
\delta T_{jj} \leq \sum_{j=1}^{N} \left| e_j^* - v(j) \right|^2 < \left( \Omega^* \right)^T + \sum_{j=1}^{N} \left| v(j) \right|^2
\]

or equivalently,

\[
\delta T_{jj} < \left( \Omega^* \right)^T + \sum_{j=1}^{N} \left| v(j) \right|^2
\]

By the triangle inequality

\[
\delta T_{jj} < \left( \Omega^* \right)^T + \sum_{j=1}^{N} \left| v(j) \right|^2
\]

In terms of norm,

\[
\| e_i^* \| < \delta \| \Omega^* \| + 2 \| v \|
\]

Further,

\[
\| f_{KLMS} \|^2 = \delta^2 \left( \tilde{e} \right) G^2 \leq \sigma_i \delta \| \tilde{e} \|^2
\]

\[
< \sigma_i \| \Omega^* \|^2 + 2\sigma_i \delta \| v \|^2
\]

where \( \sigma_i \) is the largest eigen-value of the Gram matrix \( G \).

Since all the terms on the right hand side are finite, the norm of the KLMS solution is well upper-bounded, which soundly demonstrates the stability of the KLMS in light of the conventional regularization theory in machine learning.

The significance of an upper bound for the solution norm is well studied by Poggio and Girosi in the context of regularization network theory [14]. A constraint on the solution norm like (48) ensures well-posedness of the problem through the resulting compactness of the effective hypothesis space. In Poggio’s words, compactness of the hypothesis space is sufficient for consistency of empirical error minimization, forcing smoothness and stability [14].

**IV. SIMULATIONS**

We use a simple numerical simulation to demonstrate the feasibility and efficiency of the newly derived algorithms. The example is the short-term prediction of the Mackey-Glass chaotic time series with parameter \( \tau = 30 \) and the sampling period 6s. We compare the performance of the LMS, kernel LMS, OKL, RA-RBFN and standard RBFN.

The time embedding is 10 for all the systems (i.e. using past 10 points to predict the present one) and a segment of 500 samples is used as the training data and another 100 as the test data. All the data is corrupted by the Gaussian noise with zero mean and 0.1 variance. A Gaussian kernel with kernel width 1 is chosen for all algorithms. One hundred Monte Carlo simulations are run with different realizations of noise. The results are summarized in Table 1. Fig. 1 is the learning curves for the linear LMS, KLMS and OKL (learning rate 0.1 for all and forgetting factor of OKL is 0.995).

Further,

\[
\| f_{KLMS} \|^2 = \delta^2 \left( \tilde{e} \right) G^2 \leq \sigma_i \delta \| \tilde{e} \|^2
\]

\[
< \sigma_i \| \Omega^* \|^2 + 2\sigma_i \delta \| v \|^2
\]

where \( \sigma_i \) is the largest eigen-value of the Gram matrix \( G \).

Since all the terms on the right hand side are finite, the norm of the KLMS solution is well upper-bounded, which soundly demonstrates the stability of the KLMS in light of the conventional regularization theory in machine learning.

The significance of an upper bound for the solution norm is well studied by Poggio and Girosi in the context of regularization network theory [14]. A constraint on the solution norm like (48) ensures well-posedness of the problem through the resulting compactness of the effective hypothesis space. In Poggio’s words, compactness of the hypothesis space is sufficient for consistency of empirical error minimization, forcing smoothness and stability [14].

**Fig. 1.** The learning curves of the LMS, KLMS and OKL (learning rate 0.1 for all and forgetting factor of OKL is 0.995).

**Fig. 2.** The learning curve of the RA-RBFN (learning rate is 0.007 set by the largest eigenvalue of G).
In RA-RBFN is required. Secondly, the learning rate past information to get the present error thus in a causal way. The RA-RBFN learns from errors just as the KLMS, but the RA-RBFN uses the global information to get the error at every point whereas the KLMS only uses some deviation measures, after that the learning process improves on the previous estimation using the deviation just obtained. This learning paradigm is actually very general and reminiscent of the way animals learn.

In this paper we present one learning paradigm and analyze two simplest special cases, namely RA-RBFN and KLMS. The RA-RBFN learns from errors just as the KLMS does, but the RA-RBFN uses the global information to get the error at every point whereas the KLMS only uses the past information to get the present error thus in a causal way. Secondly, the learning rate $\eta$ in RA-RBFN is required to meet the similar constraints to the KLMS for convergence as shown in Theorem 3. Additionally the reciprocal relationship of the learning step size and the regularization parameter is also interesting, which also reveals the role of the learning step size in the regularization mechanism of the KLMS as shown in (48). The analysis of the KLMS gives us some important clues about the RAN regarding its convergence and generalization. Furthermore, the results obtained in Theorem 5 relax significantly the well-posedness condition of the OKL.

The real world we live in is very rich and unexpected, which can be characterized as high dimensional. A mathematical theory of learning in high dimensions should naturally restrict the dimensionality of the space to the dimensionality of the data. Current algorithms do not do this, and the adopted methodology is to use regularization. Our recent results show that the class of algorithms based on the KLMS naturally (i.e. without regularization) provide solutions always in the data manifold, and can therefore have a tremendous impact in machine learning for high dimensional problems.

### References