Temporal Radial Basis Function For Spatio-Temporal Series

L. MESBAHI[1], A. BENYETTOU[1], F. HENDEL[2]

[1]Signal IMage PArole Laboratory, Dept. Computer Science, Science Faculty
[2] Dept. of Electronic, Genie-Electric Faculty
University of Science and Technology Oran- USTOMB,
BP1505, Oran El-Mnaouer 31000, Algeria.
Tel:(213) 41420608 - Fax: (213) 41420608

mesbahi_99@yahoo.com aek.benyettou@email.com

***Abstract* - In this paper, we present an extended form of the Radial Basis Function network called Temporal-RBF (T-RBF) network. This extended network is used in decision rules and classification in Spatio-Temporal domain applications like speech recognition, economic fluctuations, seismic measurements and robotics applications. We found that such a network complies, with a relative ease, to constraints such as capacity of universal approximation, sensibility of node, local generalisation in receptive field, etc ... For an optimal solution based on a probabilistic approach with a minimum of complexity, we propose two T-RBF models (1 and 2). Application to the problem of Mackey-Glass time series has revealed that T-RBF models are very promising, compared to traditional networks.**

***Keywords*- Temporal RBF, Classification, Spatio-Temporal, Speech recognition, Robotics applications.**

I. INTRODUCTION

One limitation of static network is their inability to respond to temporal pattern in the hidden node outputs. To respond to these patterns, the networks must have delay elements within each of its layers. The benefits of internal delays were demonstrated in [1] which is the dynamic approach, especially in signal prediction, where in the input to the network is a time varying signal and the desired output is a prediction of the signal at a fixed lag. Other examples are speech recognition and signal production, when the output autonomously follows a desired trajectory [1].

First, Day and Davenport [1] have introduced back-propagation through time (BPTT) in chaotic signal prediction tasks based on the Mackey-Glass differential delay equation. Lin, Ligomenides & Dayhoff [2] have proposed ATDNN [3], which consists of changing the delay time during learning and which gave good results in problems of eight and zero form. In problems which require information about the next event, it combines the recurrent neural network (RNN) firstly and adds time delay connections progressively [4]. Another method named Long Short Term Memory (LSTM) applied to time series benchmark problems does not even require RNN at all, because all relevant information about the next event is in fact conveyed by a few recent events contained within a small time window [5].

Our method integrates the time aspect in the RBF neural network. The novelty is that, compared to other methods described above, this approach is not a black box, so we can alter the kernels in sub-neural networks. In addition, we can apply a Bayesian classifier which introduces the prior-probability, cost of punishment in the case of rejection of misclassification. It also incorporates basic aspects of static RBF in approximation, denseness, uniqueness of interpolation and convergence rate [6]. Other extensions such as moving centres, weighted norm and different types of basis function and multiple scales were also considered. These criteria provide a useful theoretical framework for investigating radial basis function networks and learning algorithms [6]. A variety of approaches for training radial basis function networks have been developed, most of which can be divided into two stages: (i) learning the centres and field receptor in the hidden layer, (ii) learning the connection weights from the hidden layer to the output layer [7]. In this sense, we have proposed our model which integrates the time parameter in the network, in object to resolve some forgotten features in standard model, like memory state, dynamic measures, recalling phases...etc.

We therefore propose two models: *i*) first we introduce the delay time only to input neurones, ii) second we insert delay time in both input and hidden neurones. In sections II and III we describe the TRBF neural network and the different models characterising this approach, mainly the occurrence of time in hidden and input layers disjointedly. In last section, we compare our approach with standard temporal neural networks especially in application to the famous Mackey-Glass chaotic time series.

II. T-RBF APPROACH

A. Definition

The standard RBF can be trained to accomplish pattern recognition tasks with complex non linear boundaries, but are limited to processing static patterns – patterns that are fixed rather than time-varying in nature [2]. The Temporal RBF, like ATDNN, LSTM..., have been proposed to overcome this limitation. Networks with this capability can play an important role in applications that are naturally time-varying and dynamic.

Also, like classical RBF, their goal is to approximate a desired approximation by a collection of functions, named kernels [8], [9]. A kernel is characterised by a centre C_i and receptive field r, and can be chosen by k-means clustering or vector quantification.

All these parameters can be taken in account to introduce the Bayesian probabilistic classifier and prior knowledge about the problem. Moreover, we can combine this approach with other techniques like hidden Markovian models "HMM" by using the generating probabilities. In general the temporal discrimination function of class K is written in the following form [2]:

$$y_k(t_n)=\sum_{j=1}^{m1} w_{jk}\varphi\Big(\sum_{l=0}^{pl} w_j(l)x(t_n-l)+b_j\Big)+b_0 \qquad (1)$$

Next, we describe the network architecture and learning algorithm.

B. Network architecture

We propose the following architecture; see Figure 1 & Figure 2:

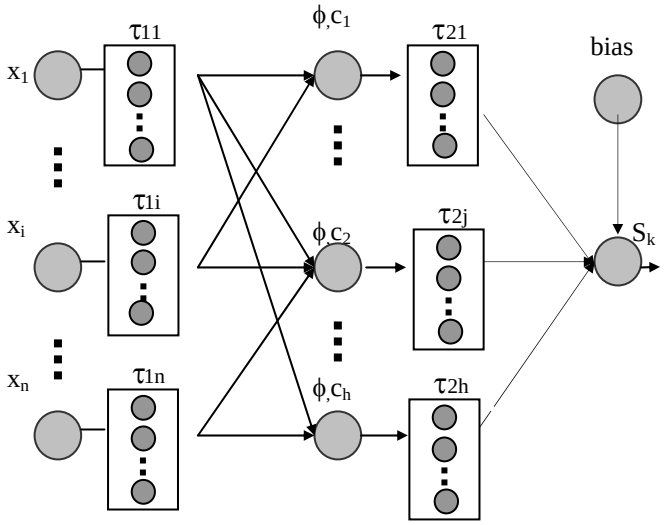

Fig.1 -General model for T-RBF

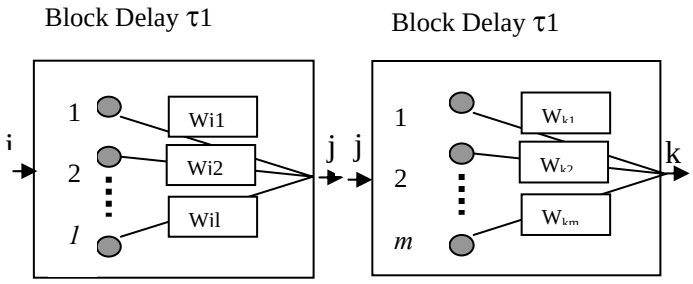

Fig.2 -Block delay(τ1, τ2) representation

Now we define:

$$S[k]=\varphi[W_{ok}+\sum_{j=1}^{m}\sum_{i=1}^{h} Yh[i][j]W_{kij}] \qquad (2)$$

Where φ: is an activation function (Linear or Sigmoid...etc.) and W_{0k} is bias.
The kernel function is:

$$Yh[i][j]=\phi_{j,\sigma i}(\|Ci-X\|) \qquad (3)$$

Where i=1..h, h: is number of centres .
j=1..m, m: is size of hidden time delay (τ2).
Dimension of Ci = dimension of X= n x l.
n= number of features of input vector.
l= is time delay size of input vector (τ1).
ϕ_j= the kernel function characterised by time delay j.

C. Unfolding network

We use this technique to eliminate the second block delay in object to reduce the complexity of computation. For this we must go through two phases:
Phase A
Time deletion by unfolding: we apply the Network Unfolding Algorithm (NUA) [10], which requires the four steps:

- Step1: Unfold inputs by creating new input nodes correspondingly and retain the weights and time delay on each connection.
- Step2: Re-adjust input time lag correspondingly for each hidden node j.
- Step3: -Unfold hidden nodes by creating new hidden nodes correspondingly and retain the weights and time delay on each connection.
  - For each newly created node do in parallel copy the whole branch which associates with the original hidden node in step2, and then connect to that new node as its branch and retain the weights
- Step4: Re-adjust input time lag by:
  - Removing the associated time delays between hidden and outputs units
  - Re-adjust the input node time lag such that each the input node takes the temporal difference.

For example if we have the following scheme, we obtain Figure 4 from Figure 3 after applying NUA.

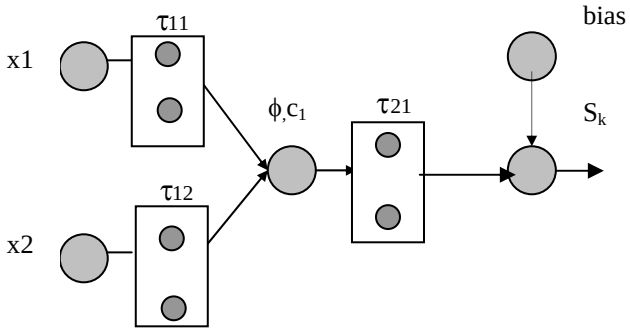

Fig.3. Initial Network with two blocks (τ1, τ2)

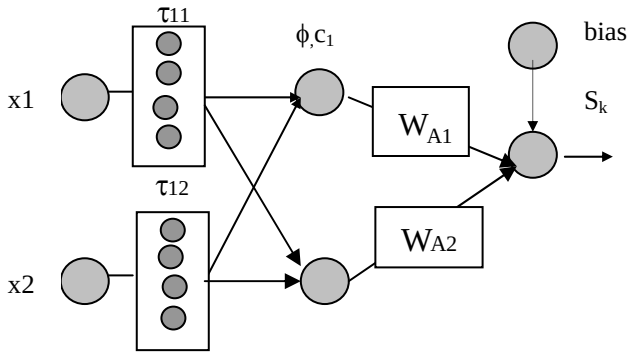

Fig.4. Deletion of Block delay τ2 by unfolding

Phase B

The aim of fusion of two hidden neurones in one hidden neurone is to represent only one centre in each iteration through the OLS learning algorithm [6] (see Figure 5). This last step has an incremental learning: in each iteration it creates a new node representing a new cluster, until obtaining a sufficient number of kernels or hidden nodes as the error reaches the stopping threshold.

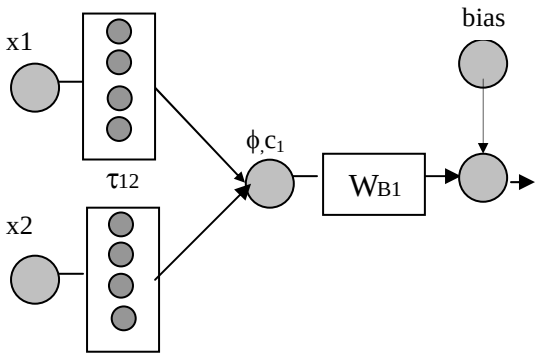

Fig.5. Resulting Network after fusion

Phases A and B yields the following tasks:

- The connection between the hidden layer and the output layer is based only on the weights, without time delay.
- The connection between the input layer and the hidden layer is based only on time delay, without weights.

C. Learning algorithm

The Orthogonal Least Square "OLS" algorithm has been applied. We suppose that the kernel function ϕ is fixed and have the same form in all hidden neurones, the initial set must be fixed. Therefore this algorithm allows us to do an incremental learning [8]. First it makes a linear separation between the input layer and the hidden layer; it creates the hidden neurones automatically [11], with application of Gram-Schmidt orthogonalisation and eliminates redundant information. Second, synaptic weights between hidden and output layers are calculated by the mean square method.

We consider the RBF network as a particular case of a linear regression model [6], defined by:

$$d(t)=\sum_{i=1} P_i\theta_i+\varepsilon(t) \qquad (4)$$

where $d(t)$: the desired output at t time;

θ_i : are the search parameters.

$\varepsilon(t)$: approximation error's of $d(t)$; $P_i(t)= P_i(x(t))$ the fixed functions of $x(t)$

The calculated function by the RBF network is the same described in Equation (4), the analogy is:

$$d=P.\theta+E \qquad (5)$$

We will consider the following notations:

d : The desired output vector's: $d = [d(1) \ldots\ldots d(N)]^T$.

N: The example number's of learning base; M : The initial number of centres.

P: The matrix of the hidden layer outputs: $P =[P_1.. P_M]$.

P_i :The vector of i^{ième} hidden cell outputs: $P_i =[P_i(1).. P_i(N)]^T$.

θ: The vector of output layer weights: $\theta = [\theta_1.... \theta_M]$.

E : The vector of errors between the calculated and desired outputs: $E = [\varepsilon(1).. \varepsilon(N)]^T$.

The resolution of equation system's (5) is a trivial problem. The solution vector θ can be defined by the mean square method.

The original idea of OLS method resides in transformation of P matrix to matrix with orthogonal columns one by one. The Orthogonalisation of columns P_i can be obtained by the decomposition of P matrix in two matrices W and A :

$$P=W.A \; . \qquad (6)$$

Where W: of size $N \times M$, is the orthogonal image of P matrix. A: of size $M \times M$, is superior triangular matrix, it contains the Orthogonalisation coefficients.

The A matrix is defined in the following form:

$$A= \begin{bmatrix} 1 & \alpha_{1,2} \ldots\ldots\ldots\ldots & \alpha_{1,M} \\ 0 & 1 \ \ldots\ldots\ldots\ldots\ldots & \alpha_{2,M} \\ & \ldots\ldots\ldots\ldots\ldots & \\ 0 & 0 \ \ldots 1 \quad \alpha_{M-2,M-1} & \alpha_{M-2,M} \\ 0 & 0 \ \ldots\ldots\ldots\ldots 1 & \alpha_{M-1,M} \\ 0 & 0 \ \ldots\ldots\ldots\ldots 0 & 1 \end{bmatrix}$$

The breeding space from the P_i vectors is the same space breeding by the W_i vectors, and the equation system's (5) can be rewritten in new form:

$$d= W*G+E \qquad (7)$$

Where $G=A.\theta$ is the researching solution.

Noting that: $$H=W^T.WE \qquad (8)$$

Since the columns of W matrix are orthogonal one by one, H is diagonal matrix with h_i elements we have:

$$h_i=w_i^t w_i=\sum_{j=1} w_{ij} w_{ji}, 1\le i\le M \qquad (9)$$

This propriety makes useful the OLS method, for the following reason: the orthogonal solution G is calculated by:

$$G=H^{-1}.W^t d \qquad (10)$$

We can calculate the G_i by:

$$g_i=\frac{w_i^t d}{w_i^t w_i}, 1\le i\le M \qquad (11)$$

This signifies that g_i elements of orthogonal solution G are dependant only on w_i column, in other words they depend on orthogonal image of calculated output for each centre. This part defines the quotient of approximation error reduction, introduced by each w_i vector and can be expressed by:

$$[err]_i=\frac{G_i^2 w_i^t w_i}{d^t d}, 1\le i\le M \qquad (12)$$

This equation is used to construct the RBF network in iterative manner. Beginning with initial set of M centres, the network is constructed to each iteration, by adding a centre

which have the $[err]_i$ maximal, and we take the correspondent G_i. For each iteration, we calculate the A and W elements by:

$$\alpha^i_{j,k}=\frac{w^t_j * pi}{w^t_j * w_j} \quad (13)$$

and $$w^i_k=P-\sum_{j=1}^{j=k-1} \alpha^i_{j,k} * w_j \quad (14)$$

We use the criterion Akaike to stop the iterations:

$$1-\sum_{i=1}^{M} err_i > \varepsilon \quad (15)$$

In the end of iterations, we calculate θ_i (the synaptic weights), by the following system:

$$G=A.\theta \quad (16)$$

III. T-RBF MODELS

A. Model 1

In this model, we consider the network with bloc delay τ1, without bloc delay τ2 (see Fig1 & Fig.2.), it means that we take into account the "l" delays of measures of input layer with simply one connection between hidden and output layers, for each hidden neurone.

B. Model 2

In this case, we consider the network with bloc delay τ1 and bloc delay τ2, it means that we take in consideration the "l" delays of measures of input layer, with "m" delays of hidden layer, representing the "m" history states for each hidden neurones.

IV. EXPERIMENTATION

A. Application of « Mackey-Glass series »

The Mackey-Glass series (1977) is a case of typical dynamic system [4], which describes the production of white blood cells, and can be generated from the delay differential equation:

$$\frac{dx(t)}{dt}=\frac{ax(t-\tau)}{1+x^c(t-\tau)}-bx(t) \quad (17)$$

With a=0.2, b=0.1, c=10, τ=17 and $x(t)$=0.8 for t<=17 in our application, for τ>17 the series become chaotic and for τ=17 is quasi periodic series [4][5][2].

B. Data base

We have constructed a base of 300 (examples and targets), each example being represented by
$<x(t_i), x(t_i -\tau)>$ and his corresponding target is $<x(t_{i+1})>$, where

$$x(t_{i+1})=x(t_i)+\frac{dx(t)}{dt}|t=t_i \quad (18)$$

C. Parameters

- *Complexity:* We have used the model2 of TRBF, for the length of bloc delay τ1 is 1 and the length of bloc delay τ2 is 2. We need to one input layer node, one output layer node and sufficiently some hidden layer nodes.
- *Akaike criterion:* This criterion fixed at 0.001 is used in OLS Algorithm. First part it stops the iterations based on quadratic error, second part it determine the complexity of hidden layer (number of kernel functions characterised by theirs centres and receptor field). In our case we have obtained 3 hidden nodes.
- *Gaussian kernel:* This kernel function is characterised by their asymptotic properties and getting accuracy in learning and generalisation phases. It is presented by the cluster (meaning centre) and spreading deviation.

D. Evaluation

- *Mean Square Error*
  To take part of veracity and the goodness results we have computed the MSE between calculated outputs from TRBF and targets from the learning base. We have obtained the MSE equal to 0.02.
- *Comparison*
  We show the results obtained by the application of analytic differential equation towards the results obtained by TRBF approach, we see the different cases in the following figures 6and 7. For example in figure 6 the curve of x(t) towards x(t-1) in our approach falls approximately on the same position of curve obtained by analytical differential equations.

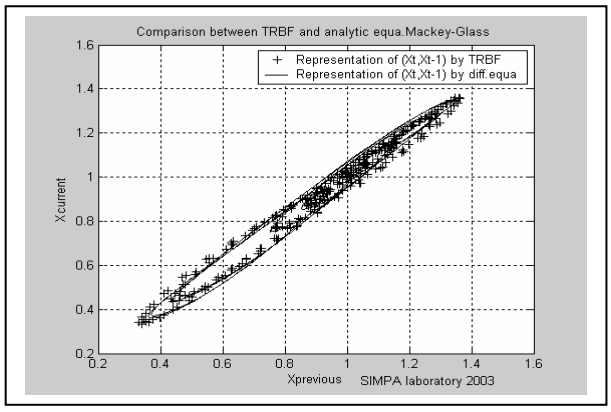

Fig.6 State space(current and previous states)

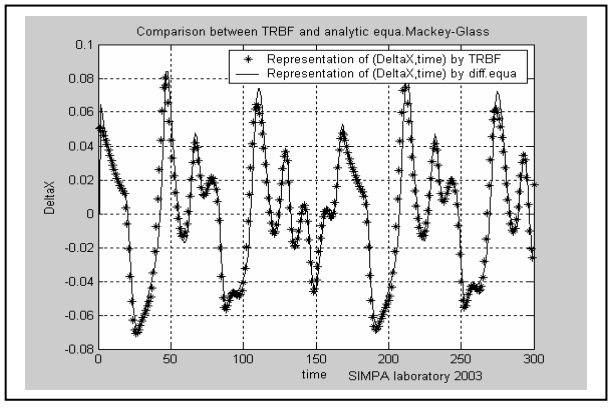

Fig.7 Strange attractors on state space

We conclude this work with a comparative table which summarise the results obtained by different temporal neural network approaches [5] against our model which have given a good results with minimum of parameters.

We define the abstract terms given in table1:
ATNN « Adaptive temporal neural network », SOM «Self Organising Map», MLP«Multi Layers Perceptron», AMB «An improved Memory Based regression», LSTM « Local Short Term Memory », TRBF « Temporal Radial Basis Function ».

Table1. Position of TRBF among different temporal methods

Temporal Methods	Units	Parameters	NMSE		
			T=2	T=6	T=84
ATNN	20	120	-	0.005	-
SOM	-	10x10	-	0.013	0.06
MLP	4	25	-	0.0511	0.46
AMB	-	-	-	-	0.054
LSTM	4	113	0.0214	0.1184	0.47
TRBF	5	9	0.0198	0.005	-

V. CONCLUSION

Inspection of the comparative table and the figures suggests that the proposed Spatio-temporal estimator based on temporal radial basis function is effective and accurate with a minimum complexity. The advantage of this work is to show that a complicated problem based on the Mackey- Glass time series prediction can be recognised by such a simple TRBF architecture. We thus conclude from this study that the TRBF is a good tool to tackle temporal signal processing and prediction problems.

The network accomplishes the identification of background model of Mackey-Glass delay differential equation again with high accuracy (NMSE =0.0198). This proposed architecture can be useful and promising in biomedical prediction tasks, recognition of trajectories from moving targets, motion visual images, robotics application and speech recognition. Also we can introduce the prior probability and cost errors in terms of Bayesian probabilistic approach, an application that needs to determine the winner class with flexible decision, since our model is based on collection of kernel functions with theirs features in probabilistic approximation.

REFERENCES

[1] S.P. Day, M.R. Davenport. Continuous-time temporal back-propagation with adaptable time delays, *IEEE Transaction on Neural Networks*, vol.2, pp. 348-354, 1993.

[2] D.T. Lin. The adaptive time delay neural network characterization and application to pattern recognition, Prediction and signal processing, *Thesis Report PHD*, University of Maryland, 1994.

[3] C. Wohler, J.K. Anlauf. Real time object recognition on image sequences with adaptable time delay neural network algorithm - application to autonomous vehicles, Image and Vision, *Computing Journal* , vol. 19, No. 9, pp. 593-618, 2001.

[4] R. Bonne, M. Cruciano, J.P.A. De Beauville. An algorithm for the addition of time-delayed connections to recurrent neural network, *ESANN proceeding*, Bruge (Belgium), 293-298, 2000.

[5] F. Gers, D. Eck, J. Schmidhuber. Applying LSTM to time series predictable through time window approaches, Technical Report, *IDSIA-IDSIA-22-00*, Institute Dalle Molle di studi Sull intelligenza artificiale, Galleria2, Switzerland, 1999.

[6] N.B. Karyianis. Reformulated radial basis function neural network trained by gradient descent, *IEEE transaction on Neural Network*, vol. 10, No. 3, pp. 657-669, may 1999.

[7] S. Haykin. Neural Networks a comprehensive foundation, Prentice Hall Upper Saddle River, New Jersey, 1999.

[8] G. Zheng , S. Billings. Radial basis function network configuration using mutual information and the orthogonal least squares algorithms, *Neural Network* vol. 9, No.9, pp. 1619-1637, 1996.

[9] Z. Mekkakia, L. Mesbahi, M. Lakehal. Initialising of RBF centers by SOM Algorithm, *MS'2002*, International conference on modeling and simulation in technical and social sciences, pp. 717-723, Girona, Catalonia, Spain 25- 27 june 2002.

[10] D.T. Lin, J.E. Dayhoff. Network unfolding algorithm and universal spatio-temporal function approximation, *Technical Research Report, TR95-6*, Institute for system research ISR, University of Maryland, College Park MD, 1995.

[11] M.K. Titsias, A.C. Likas. Shared kernel model for class conditional density estimation, IEEE transaction on Neural Network, vol. 12, No. 5, pp. 987-996, September 2001.