MODELLING OF NONSTATIONARY PROCESSES USING RADIAL BASIS FUNCTION NETWORKS[1]

D Lowe and A M^{c}Lachlan

Aston University, UK

ABSTRACT

This paper reports preliminary progress on a principled approach to modelling nonstationary phenomena using neural networks. We are concerned with both parameter and model order complexity estimation. The basic methodology assumes a Bayesian foundation. However to allow the construction of pragmatic models, successive approximations have to be made to permit computational tractibility. The lowest order corresponds to the (Extended) Kalman filter (1) approach to parameter estimation which has already been applied to neural networks (2). We illustrate some of the deficiencies of the existing approaches and discuss our preliminary generalisations, by considering the application to nonstationary time series.

INTRODUCTION

There have been several major advances in the development of the theoretical foundations of neural networks over the last decade. These have been primarily motivated by statistical pattern recognition techniques, and from the complementary deterministic dynamical systems approaches. However one basic assumption has been that the underlying *generator* of the data is stationary, i.e. that the (unknown) process which produced the observations was itself time invariant. Note that this is true even for 'dynamic' time series problems (3). It is not the dynamic nature of the observed data we are interested in, but in the dynamic nature of the underlying process.

The assumption of stationarity is acceptable in many circumstances, particularly if the time evolution permits quasi–stationary 'windowing' of the data. In these circumstances the neural network may be re-optimised at regular intervals, when the network is no longer able to describe the structure of the changing generator. However there are other real world circumstances when this approach does not give adequate approximation abilities.

Consider the example of predicting averaged UK short term electricity load demand. Figure 1 illustrates the effect of training a Radial Basis function network over a fixed window over the first 100 samples, and using this static model of the generator to perform short term forecasts. It is clear that because of the seasonal trend in this data, the ability of the static model to forecast is severely impaired once the assumptions inherent in the training data become invalid.

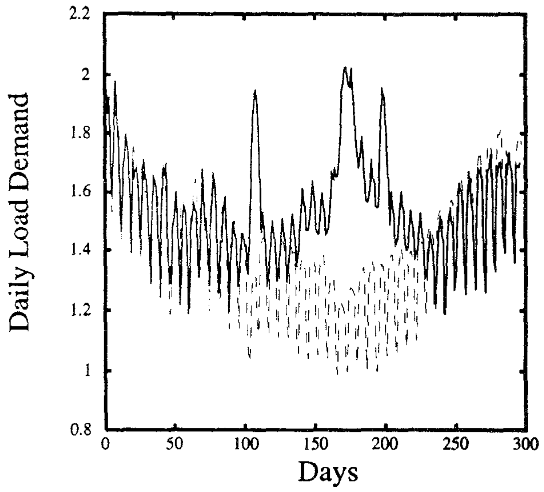

Figure 1: *Daily electricity load demand. Dashed line denotes the actual consumption, solid line denotes the predicted demand. The network exhibits poor performance beyond the training set.*

In this case the dominant cause of the error is due to the drift of the average target values. This is a slow time scale effect and hence the problem may be alleviated by using just the bias weight to track this slow time scale variation[2]. Figure 2 illustrates the effect of using a static network generator, apart from the bias weight on the output node, which is adapted on–line by feeding back the actual error to the bias node. This on–line adaptation of the bias allows the network to track the actual errors more accurately. Clearly this minor change to the network has a profound effect on performance.

In this paper we are concerned with problem domains in which batch and recursive training processes ('recursive' implies we are allowed to cycle repeatedly through the training data) are not viable options.

[1]The authors would like to thank the EPSRC for support for this work under contract #GR/J75425

[2]The different weights in a network have different functional roles, and in the context of nonstationary problems this also means they have different characteristic time scales

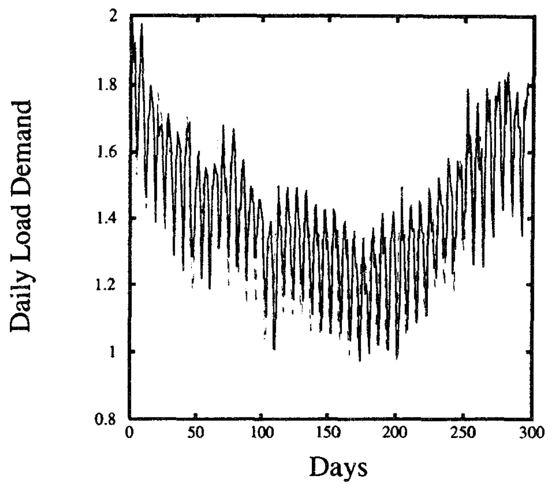

Figure 2: Daily electricity load demand. Solid line denotes the actual consumption, dashed line denotes the new, predicted demand by on-line updating the bias vector.

One reason is due to potentially very large data sets, but the main reason is because the assumptions governing different portions of data evolve in time (such as periodicity, seasonality or multiple trending behaviours on different time scales).

RELATED WORK

Despite the obvious importance of designing principled approaches to sequential nonlinear models for nonstationary environments, there has been surprisingly little activity in the neural network domain. However two notable exceptions are due to Platt (4), and Kadirkamanathan and Niranjan (2). (There have been other network schemes, for example the Adaptively Trained Neural Network of Park et al (5), which trains sequentially by linearising the error function. However the developments of Platt and Kadirkamanathan are closer to the Bayesian extensions we consider.) Platt has previously developed the Resource Allocating Network which is intended to dynamically increase the number of (Gaussian) basis functions in a Radial Basis Function network by processing information sequentially. An improvement over the RAN network, the EKF-RAN network was developed by Kadirkamanathan and Niranjan. This involved replacing the LMS update algorithm with the extended Kalman filter (1, 6) as well as providing a geometric description of the increasing model complexity criterion. The approach adopted in this paper is a natural extension of this work, based on a Bayesian philosophy.

METHODOLOGY

We are interested in the sequential estimation of nonlinear stochastic systems, primarily by a probabilistic description. For example, the class of stochastic approximation methods would be appropriate. One of us (7) has previously considered a stochastic approximation method for the unsupervised updating of the positions and smoothing factors of the first layer of an RBF using effectively a sequential EM algorithm. However for nonstationary problems we have to be prepared to trade off 'plasticity' for 'convergence'. This is because one criterion for the stochastic approximation methods to converge asymptotically is that the effective 'learning rate' should decrease to zero. However this is in conflict with the requirement of tracking and adapting to novel data. We cannot have both asymptotic consistency and adaptability.

For this paper the general approach we wish to follow is through the exploition of Bayes theorem. Our problem may be expressed as follows: Given an estimate of the network parameters $\hat{\boldsymbol{w}}_{t-1}$ at instant $t-1$, which was based on all the information Y_{t-1} up to this instant, we wish to obtain a new estimate $\hat{\boldsymbol{w}}_t$ based on the previous estimate and the new received information $\boldsymbol{y}_t$. In the general case we are concerned with the *distribution* of possible weight values, i.e. the posterior

$$p(\boldsymbol{w}_t|Y_t) = \frac{p(\boldsymbol{y}_t|\boldsymbol{w}_t)p(\boldsymbol{w}_t|Y_{t-1})}{p(\boldsymbol{y}_t|Y_{t-1})} \tag{1}$$

where $p(\boldsymbol{w}_t|Y_{t-1})$ represents the prior over the parameters given the measurements Y_{t-1} and the likelihood $p(\boldsymbol{y}_t|\boldsymbol{w}_t)$ is determined from the noise density. Once we have calculated this conditional density, any estimate of the parameter vector may be obtained, at least in principle. For example the conditional mean or the maximum a posteriori estimate, $\hat{\boldsymbol{w}}_t$, of the weights may be calculated from $p(\boldsymbol{w}_t|Y_t)$. It is not generally possible to obtain analytic solutions for the Bayesian recursive relations, or even extract computationally tractable algorithms except in special circumstances. For instance, under assumptions of Gaussian processes and linear systems

$$\begin{aligned} p(\boldsymbol{y}_t|\boldsymbol{w}_t) &= e^{-\frac{1}{2}(\boldsymbol{y}_t-\boldsymbol{f}_t(\boldsymbol{w}_t))^T R_t^{-1}(\boldsymbol{y}_t-\boldsymbol{f}_t(\boldsymbol{w}_t))} \\ p(\boldsymbol{w}_t|Y_{t-1}) &= e^{-\frac{1}{2}(\boldsymbol{w}_t-\hat{\boldsymbol{w}}_{t-1})^T P_{t-1}^{-1}(\boldsymbol{w}_t-\hat{\boldsymbol{w}}_{t-1})} \\ \boldsymbol{f}_t(\boldsymbol{w}_t) &= C_t\boldsymbol{w}_t \end{aligned} \tag{2}$$

the Kalman filter is obtained as a point estimate of the distribution in equation 1, i.e.

$$\begin{aligned} K_t &= P_{t-1}C_t^T(R_t + C_tP_{t-1}C_t^T)^{-1} \\ \hat{\boldsymbol{w}}_t &= \hat{\boldsymbol{w}}_{t-1} + K_t(\boldsymbol{y}_t - \boldsymbol{f}_t(\hat{\boldsymbol{w}}_{t-1})) \\ P_t &= (I - K_tC_t)P_{t-1} \end{aligned} \tag{3}$$

For nonlinear systems (such as neural networks), the likelihood function is no longer a Gaussian in the weights, so we cannot perform the analysis and therefore have to make approximations. The extended Kalman filter is an estimate of the parameter vector based upon a linearisation of the model around the current state estimate

$$\boldsymbol{f}_t(\boldsymbol{w}_t) = \boldsymbol{f}_t(\hat{\boldsymbol{w}}_{t-1}) + \nabla_w \boldsymbol{f}_t(\hat{\boldsymbol{w}}_{t-1})(\boldsymbol{w}_t - \hat{\boldsymbol{w}}_{t-1}) \quad (4)$$

and this basically represents the state of the art in sequential nonlinear modelling.

Although the extended Kalman filter can be used in slowly varying nonstationary environments (though there is no guarantee of convergence), there are several deficiencies with the EKF–RAN approach:-

- Model order adaptation relies upon user supplied threshholds with no a priori indication of suitable values,
- there is no mechanism for reducing model order even though excessive model order can have a detrimental effect on filter performance,
- the linearised approximation in the extended Kalman filter used in the calculation of the filter gain may not be appropriate for the high degrees of nonlinearity in neural networks,
- the filter relies upon an initial estimate of the covariance which, even if valid initially, may become inaccurate in nonstationary environments,
- there is no principled way to initialise the new covariance matrix entries when the model order is increased in a neural network,
- the weight initialisation presription is designed to work with Gaussian basis functions and, as it fits noise in the data, is not robust against outliers.

ITERATED EKF, HIGHER ORDER FILTERS AND QUASI-NEWTON OPTIMISATION

The calculation of the gain in the extended Kalman filter relies on a linearisation of the network function. While this ensures that the likelihood is Gaussian in the network weights, it is a poor approximation given that some quadratic terms in the log likelihood are being discarded, along with higher order terms.

However we can obtain a better approximation to the nonlinearity in several ways. By using the Hessian of the network, we can expand the network output (as a function of the weights) to second order about the current value, thus giving the *second order extended Kalman filter* (6). Alternatively, we can still linearise the model, but make use of the *iterated EKF* (sometimes known as the recursively iterated EKF to distinguish it from a simplified version (6)). The iterated EKF re-linearises the model about each new weight vector prediction *using the same data point and the same prior*, iterating until convergence is achieved, i.e.

$$\begin{aligned} K_t^i &= P_{t-1}(C_t^i)^T(R_t + C_t^i P_{t-1}(C_t^i)^T)^{-1} \\ \hat{\boldsymbol{w}}_t^i &= \hat{\boldsymbol{w}}_{t-1} + K_t^i(\boldsymbol{y}_t - \boldsymbol{f}_t(\hat{\boldsymbol{w}}_t^i)) \\ &\quad - K_t^i C_t^i(\hat{\boldsymbol{w}}_{t-1} - \hat{\boldsymbol{w}}_t^i) \end{aligned} \quad (5)$$

where $C_t^i = \nabla_w \boldsymbol{f}_t(\hat{\boldsymbol{w}}_t^i)$ and $\hat{\boldsymbol{w}}_t^0 = \hat{\boldsymbol{w}}_{t-1}$. Only then is the covariance updated as in equation 3 and the next data point analysed.[3]

If the iteration converges, then this will give a more accurate approximation to the maximum of the posterior than the EKF, but it cannot circumvent the limitations in performance due to its failure to capture significant nonlinear effects.

Ideally, the posterior should be maximised directly using some conventional optimisation scheme. Once an optimum has been found, the (Gaussian) prior for the next datum can be constructed by calculating the Hessian of the log posterior (8). This is generally far more computationally expensive, but this should not be a problem unless the data are arriving at too high a frequency.

Irrespective of whether the prior at each timestep is estimated via the network Hessians or some filter evolution equation, the fact that the entire data history is contributing can inhibit the network from responding to evolution of the data generator. One simple method of alleviating this problem involves adding a small multiple of the identity to the prior covariance matrix at each stage - this has the effect of widening the Gaussian centred on the old weights, and hence allows the likelihood a greater influence. It should be noted that this is a bit of an ad hoc fix, and that a more principled mechanism should be found for dynamically adjusting the contribution of the prior.

The Quadratic Map

In order to examine the relative merits of these optimisation schemes, a simple non-stationary problem was examined. The quadratic map is generated by the following simple iterative equation

[3] An alternative formulation updates the covariance P on each iteration and uses a slightly different weight iteration procedure. This can be shown to be entirely equivalent to equation 5, but is more computationally expensive.

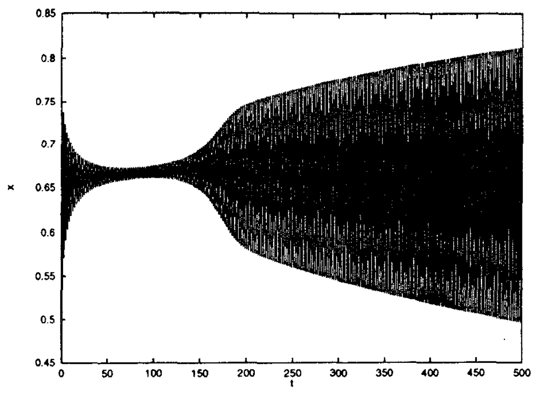

Figure 3: *Quadratic map waveform with uniformly increasing coefficient.*

$$x_{t+1} = \alpha x_t(1 - x_t) \qquad (6)$$

The nature of the series generated by equation 6 depends on the value of the parameter α. For $\alpha \in [0, 1]$ there is a single fixed point at the origin, and the series quickly collapses to zero. For $\alpha \in (1, 3)$, there will be a fixed point at some non-zero value to which the series would converge. When $\alpha > 3$, the series generates attractors of period 2^r, $r \rightarrow \infty$ as $\alpha \rightarrow 3.5699\ldots$, beyond which the series is chaotic.

For fixed alpha, all of the above are stationary series, but by letting α vary with time, we can generate a simple non-stationary series. The series used here consisted of 500 points initialised with a random $x_0 \in (0, 1)$. The initial value of α was 2.95, and was incremented in uniform steps to a final value of 3.25. The waveform thus generated is shown in Figure 3. For $\alpha < 3$ the waveform is trying to converge to a slowly growing fixed point, whereas there is a new attractor of period 2 beyond the critical value $\alpha = 3$.

Fixed size networks

Before investigating RAN networks, we shall illustrate typical performances of the various algorithms when used to train a fixed size network. The network used in the following example consists of 4 Gaussian basis functions with fixed widths. The weights are initialised using the RAN procedure, whereby the units are added one at a time, each new unit being centred on the corresponding data point, with the second layer weights being adjusted to fit the signal exactly. While this is not optimal (as mentioned earlier) it is likely to lead to quicker convergence than a random initialisation.

Several initialisations for the respective covariance matrices in the likelihood and prior in equation 2 were tried – while the exact form of the innovations sequence varied with the initialisation as expected, the comparative performances of the different algorithms were not affected significantly.

Figures 4 and 5 show example innovations sequences for the networks with 4 basis functions. Unlike the stationary case, neither algorithm consistently outperforms the other. BFGS tended to outperform EKF in regions where the waveform is changing slowly, whereas EKF generally gave the better performance when the waveform was undergoing its most significant changes.[4]

RAN networks

The RAN procedure has been claimed to yield networks of optimal size (4, 2) for a given problem. The reality is that the model order increment procedure is highly dependent on user supplied parameters. Figures 6 (EKF) and 7 (BFGS) show the innovations sequence obtained from RAN networks starting

[4]Even though the coefficient in equation 6 is changing at a constant rate, the effects on the waveform can be more dramatic at some timesteps than at others – see Figure 3.

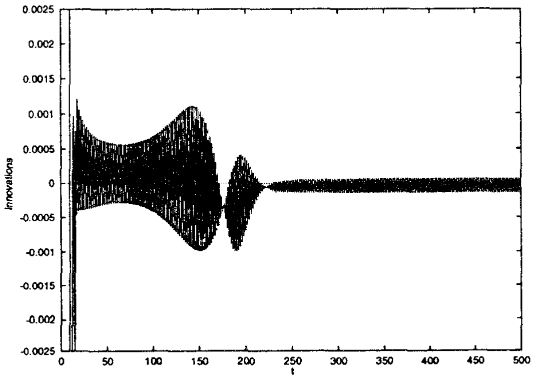

Figure 4: *Innovations sequence for a 4 unit network using EKF.*

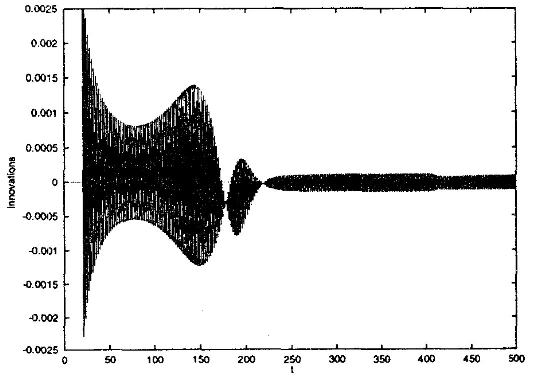

Figure 5: *Innovations sequence for a 4 unit network using BFGS.*

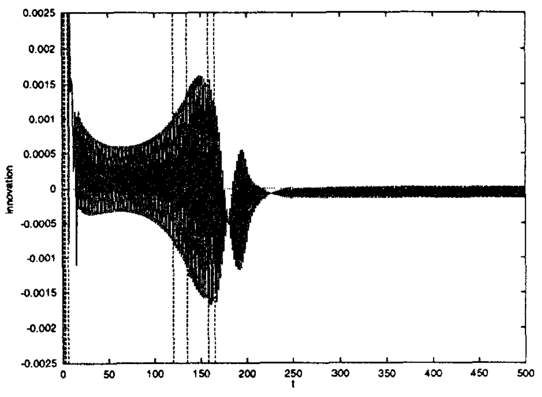

Figure 6: *Innovations sequence for RAN network using EKF - vertical lines indicate where new units are added.*

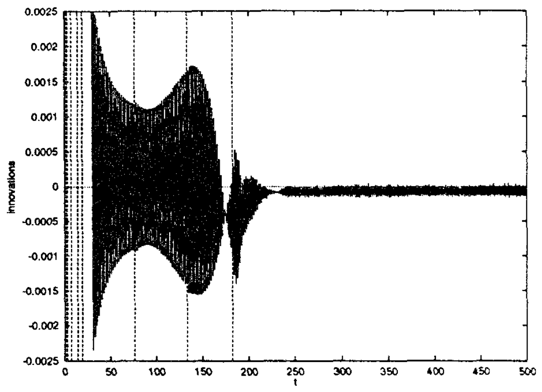

Figure 7: *Innovations sequence for RAN network using BFGS - vertical lines indicate where new units are added.*

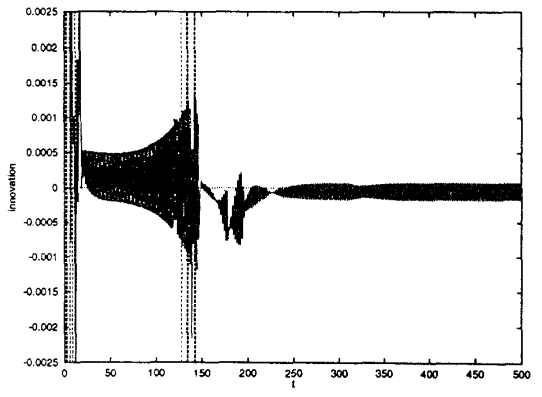

Figure 8: *Innovations sequence for RAN network using EKF - vertical lines indicate where new units are added.*

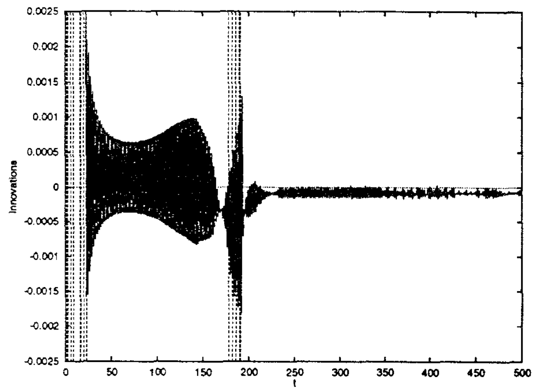

Figure 9: *Innovations sequence for RAN network using BFGS - vertical lines indicate where new units are added.*

with a single basis function and growing according to the following prescription :- a new Gaussian basis function, centred on the corresponding data point, is added if the absolute value of the innovation is greater than 0.001 and the distance from the data point to the nearest existing centre is greater than a threshold which starts at 0.25 and decays to a minimum value of 0.1. The vertical lines show where a new unit is added. The EKF produces a network of 7 basis functions, while the BFGS produces one with 8. It can be seen in both cases that new units can be added in regions where there doesn't seem to be any major change in the structure of the data - this comes from the decaying distance criteria, and is an unsatisfactory feature of this approach. It should also be noted that the performance is actually poorer than the corresponding fixed-order networks with only 4 units.

Figures 8 (EKF) and 9 (BFGS) perform the RAN procedure again, this time with the maximum and minimum distance thresholds reduced to 0.1 and 0.01 respectively. The EKF network now grows to 11 units, with the BFGS growing to 14. While the innovations are smaller in this case, we can again see units being added at inappropriate moments :- in Figure 9, we can see a series of units being added merely due to the fact that the network is not being given time to converge from a possible inappropriate initialisation of the new elements in the prior covariance.

DISCUSSION OF RESULTS

From a wide series of tests on quadratic map waveforms, it was found that the BFGS algorithm could consistently outperform the EKF and the iterated EKF in stationary situations. In the non-stationary cases, the EKF occasionally seemed better able to cope in regions where the waveform was undergoing significant changes in structure. This is most probably due to the loss of information in the EKF making it more adaptable to a changing environment - the increased accuracy of BFGS can result in a more restrictive prior. Further investigations are necessary to determine whether this wll be the case in general, rather than just a peculiarity of this particular experiment.

In the RAN tests, the inadequacies of the model order increment procedure were such that no definite statements can be made regarding the relative merits of the different training algorithms. It should be noted that the performance of RAN networks can frequently be worse than smaller sized fixed order networks, and on occasion can be dramatically worse. This is due to the arbitrariness of the model order increase criteria – as illlustrated in Figures 6 to 9, units can be added at inappropriate moments.

As far as the iterated EKF is concerned, it had a tendency to fail to converge, and would occasionally diverge. Initial results show that the EKF can perform almost as well as BFGS when the iteration succeeds, but in other cases, the failure to converge on a solution degrades performance significantly.

The second order EKF, on the other hand, tended to approximate the posterior maximum sufficiently innaccurately that the Hessian could not in general be guaranteed to be positive definite. Consequently, this yielded highly inappropriate priors for the next timestep, thus degrading performance further. In cases where this did not occur, the performance was found to be no better on average than the EKF, and thus there seems no point in devoting further time to the investigation of second order filters when the model is so highly non-linear.

WORK IN PROGRESS

The fact that the prior contains information about the entire sequence means that the network may be slow to adjust to genuine novelty in the data. Work is in progress to investigate whether the introduction of hyperparameters in the likelihood and prior can help in this matter. Such hyperparameters could be optimised at the next level of Bayesian inference, thus varying the weighting of the prior as the data sequence is processed. This is in a similar vein to work already carried out in static environments (9), though modifications to the procedure are necessary when using sequential training. This may also help reduce sensitivity to the boundary conditions at $t = 0$.

As is discussed here, the results can be sensitive to the choice of boundary conditions and RAN parameters. Not only can basis functions be introduced at inappropriate moments merely due to the arbitrary distance criterion, but the error criterion can be biased by poor initialisation of a previous basis function (and its corresponding covariance). New units can be added merely because the network has not been given a chance to converge since the last unit was added, resulting on occasion in sequences of units being added in short order irrespective of the nature of the data sequence. More sophisticated measures of non-stationarity are therefore required for model order increment.

The initialisation scheme used here assumes Gaussian basis functions, and initialisation on an outlier will have an adverse effect on convergence. Alternative initialisations are required which do not fit the noise and which may be used for non-Gaussian basis functions, such as r^3 or $r \log r$.

Local basis functions, as used here, can drift out of the relevant region of the input space, thus becoming redundant. In the RAN context, this can result in the creation of unnecessarily large networks. A model order decrement procedure is therefore desirable in such cases. Note that this would not be a problem with non-local basis functions.

REFERENCES

1. Candy J. V., 1986, "Signal Processing : The Model-Based Approach", McGraw-Hill.
2. Kadirkamanathan V. and Niranjan M., 1993, "A function estimation approach to sequential learning with neural networks", Neural Computation, 5, 954–975.
3. Lowe D. and Webb A. R., 1991, "Time series prediction by adaptive networks: A dynamical systems perspective", IEE Proceedings–F, 138, 17–24.
4. Platt J. C., 1991, "A resource allocating network for function interpolation", Neural Computation, 3, 213–225.
5. Park D. C. and El-Sharkawi M. A., 1991, "An adaptively trained neural network", IEEE Transactions on Neural Networks, 2, 334–345.
6. Bar-Shalom Y. and Fortmann T. E., 1988, "Tracking and Data Association", Academic Press.
7. Lowe D., 1991, "On the iterative inversion of RBF networks: A statistical interpretation". In "2ndIEE International Conference on Artificial Neural Networks".
8. Pearlmutter B. A., 1994, "Fast exact multiplication by the hessian", Neural Computation, 6, 147–166.
9. MacKay D. J. C., 1991, "Bayesian interpolation", Neural Computation, 4, 415–447.