Radial basis functions: normalised or un-normalised?

M. R. Cowper, B. Mulgrew, and C. P. Unsworth,

Abstract— **In this paper a simple and robust combination of architecture and training strategy is proposed for a radial basis function network (RBFN). The proposed network uses a normalised Gaussian kernel architecture with kernel centres randomly selected from a training data set. The output layer weights are adapted using the numerically robust Householder transform. The application of this normalised radial basis function network (NRBFN) to the prediction of chaotic signals is reported. NRBFN's are shown to perform better than un-normalised equivalent networks for the task of chaotic signal prediction. Chaotic signal prediction is also used to demonstrate that a NRBFN is less sensitive to basis function parameter selection than an equivalent un-normalised network. Normalisation is found to be a simple alternative to regularisation for the task of using a RBFN to recursively predict, and thus to capture the dynamics of, a chaotic signal corrupted by additive white Gaussian noise.**

I. Introduction

Normalised radial basis function networks (NRBFN's) were first introduced by Moody and Darken [1] in 1989. Since then there has been a mixed response towards NRBFN's in the literature. Indeed, one of the motivating forces for this paper is to provide detailed results from a number of simulation studies, which can be used to come to a more positive opinion on NRBFN's. The specific interest for this work is in applying a NRBFN to the prediction of chaotic signals.

In its favour the NRBFN has been theoretically proven to be capable of universal approximation in a satisfactory sense [2], it has been shown to have good generalisation properties [2], [3], and it has been said that the NRBFN is less affected by a poor choice of basis function parameters than an un-normalised network [4], [5], [3]. A possible explanation for why NRBFN's can offer advantages with respect to un-normalised radial basis function networks (UNRBFN's) could be associated with the fact that a NRBFN can exhibit both localised *and* non-localised behaviour [6], whereas an UNRBFN only features localised behaviour.

Despite the above positive aspects of NRBFN's, certain disadvantages have also been reported. For example, Cha and Kassam [6] found that although a NRBFN had a better interpolation capability than its un-normalised counterpart, they discovered it was rather difficult to adapt the basis function parameters of a NRBFN using the stochastic gradient (SG) nonlinear optimisation technique. However, whilst this can lead to better solutions than using a linear optimisation technique, there is not a guarantee of this, and nonlinear optimisation is more time consuming. Furthermore, Cha and Kassam found that the SG algorithm often yields a sub-optimal solution. Shorten and Murray-Smith [4] warned against what they called "side-effects" of normalisation. Examples of these side-effects include basis function reactivation and shifts in basis function maxima. However, such side-effects do not necessarily mean NRBFN's are not, for example, good classifiers or function approximators. Indeed, they may even be part of the explanation why a NRBFN has been shown to be less sensitive to poor basis function parameter selection, than an UNRBFN. Shorten and Murray-Smith showed an example of function approximation using a NRBFN and also an UNRBFN. They demonstrated that the reactivation and maxima shift side-effects were present in the NRBFN. However, although they did not give numerical values for the relative performance of each network, graphically it appeared that the NRBFN achieved a better fit to the target function than the UNRBFN. If indeed this is the case it would support the view that such side-effects do not have a negative impact on network performance, and they may even be part of the explanation behind any performance benefits which normalisation brings. More recent results [3] have also shown a NRBFN to outperform an UNRBFN for a function approximation task.

In this paper a simple and robust combination of architecture and training strategy is proposed for a RBFN. The proposed network uses a normalised Gaussian kernel architecture with kernel centres randomly selected from a training data set. The output layer weights are adapted using the numerically robust Householder transform [7], [8]. The use of a nonlinear optimisation technique was not considered due to the disadvantages of such an approach already highlighted above.

II. Radial basis function networks

A radial basis function network (RBFN) [12] is linear in its parameters, therefore once suitable basis function parameters have been chosen it can be trained using a fast linear supervised training scheme. In this section the structure of a RBFN is described, an approach to basis function parameter selection is explained, and a least squares (LS) adaptation technique for training the output layer weights is discussed. A RBFN with M kernels, or hidden units, has the following overall response function,

$$f(\mathbf{x}(n)) = \sum_{i=1}^{M} w_i \phi_i(n) \qquad (1)$$

where $\mathbf{x}(n)$ is a vector in the input space of the RBFN, $\phi_i(n)$ is the response function of the i^{th} kernel, and w_i is the

Signals and Systems Institute Division of Engineering and Electronics University of Edinburgh, King's Buildings, Mayfield Road, Edinburgh, EH9 3DQ, UK.

weight associated with the i^{th} kernel. The most common nonlinear kernel function used in RBFN's is the Gaussian function,

$$\phi_i(n) = \exp\left[\frac{-||\mathbf{x}(\mathrm{n}) - \mathbf{c}_\mathrm{i}||^2}{2\sigma_\mathrm{i}^2}\right], \quad \mathrm{i} = 1, 2, ...\mathrm{M} \qquad (2)$$

where $||.||$ is the Euclidean distance measure, $\mathbf{c}_i$ is the position of the i^{th} kernel's centre in the input space of the RBFN, and σ_i is known as the width of the i^{th} kernel.

Two different Gaussian RBFN architectures were considered for the prediction analysis reported in this paper. Firstly, a RBFN which used the Gaussian kernel function in equation (2) was considered. This will be referred to from now on as the un-normalised RBFN or UNRBFN. Secondly, a RBFN which used the normalised Gaussian kernel function [1] was considered. This will be referred to from now on as the normalised RBFN or NRBFN. The normalised Gaussian kernel function is defined below,

$$\phi_i(n) = \frac{\exp\left[\frac{-||\mathbf{x}(\mathrm{n})-\mathbf{c}_\mathrm{i}||^2}{2\sigma_\mathrm{i}^2}\right]}{\sum_{j=1}^{M}\exp\left[\frac{-||\mathbf{x}(\mathrm{n})-\mathbf{c}_\mathrm{j}||^2}{2\sigma_\mathrm{j}^2}\right]} \qquad (3)$$

where $\phi_i(n)$ is the i^{th} normalised kernel output. This is very similar to equation (2), except here the kernel output is divided by the sum of all the kernel outputs. Therefore, the outputs of all the kernels add up to one.

The method used to select the centres of the RBFN will be referred to as the randomly selected centres (RSC) technique. It involves choosing centres from the training data at random. A uniform random number generator [23] (RNG) is used to pick points at random from the training data. These points are used as the starting elements of the centres. For example if the RBFN has an embedding dimension of N (*i.e.* N input nodes), and an embedding delay of 1 sample (*i.e.* a 1 sample delay between each input node), then a starting element is picked, along with the next $N-1$ successive data points, to obtain centres in the RBFN's N-dimensional input space. This centres selection technique was implemented so that, given two RBFN's with the same embedding dimension, embedding delay, training length, and training data set, but one with M_1 kernels, and the other with M_2 kernels, where $M_2 > M_1$, the set of M_1 centres is a subset of the M_2 centres.

A universal kernel width was used (*i.e.* the same width for each kernel): $\sigma_i = \sigma, i = 1, 2, ..., M$. The following equation was used to calculate the width σ,

$$\sigma = \frac{d_{me}}{\sqrt{2M}} \qquad (4)$$

where d_{me} is the maximum Euclidean distance between any 2 centres, and M is the number of kernels. Such a choice for σ in an UNRBFN ensures that the Gaussian kernel functions are neither too peaked nor too flat [17]. Once the centres and widths of the M kernel functions have been selected, the M output layer weights $w_i, i = 1, 2, .., M$, can be trained using a supervised linear least squares technique. The Householder transform [7], [8], which is numerically robust as it avoids estimating the inverse of the autocorrelation function (ACF) matrix directly, was used to train the output layer weights of the RBFN.

III. Nonlinear prediction

Given a vector $\mathbf{x(n)}$, from a time series $\{x(n)\}$, which has an embedding dimension N and an embedding delay τ, *i.e.*

$$\mathbf{x}(n) = [x(n), x(n-\tau), ..., x(n-(N-1)\tau)]^T \qquad (5)$$

an estimate $\hat{x}(n+1)$ of the next data sample $x(n+1)$ is formed by constructing a nonlinear predictor function $f()$ where,

$$\hat{x}(n+1) = f(\mathbf{x}(n)) \qquad (6)$$

Equation (6) is for 1-step ahead prediction, but this could be generalised for K-step ahead prediction, *i.e.*

$$\hat{x}(n+K) = f_K(\mathbf{x}(n)) \qquad (7)$$

where $f_K(\mathbf{x}(n))$ would, in general, represent a different predictor function for each value of K.

A problem that must be addressed when using a RBFN to approximate the predictor function $f_K(\mathbf{x}(n))$ is that of over-fitting. Over-fitting [21] occurs when a model, (*i.e.* RBFN), is too complex and fits to spurious quirks (*i.e.* noise) in the data. This means that the model will perform less well on non-training data than on training data. To avoid over-fitting a technique known as early-stopping [21] was used. The data was divided into three equal sets of length Y samples in each: a training set (the first Y samples), and 2 non-training sets; a testing set (the next Y samples), and a validation set (the next again Y samples). By monitoring the predictor's performance on the non-training data, the problem of over-fitting can be avoided: if the non-training error started to increase, then the predictor's training can be stopped. The prediction performance measure that has been used for this paper is the normalised mean square error (NMSE).

In order to choose suitable embedding parameters and thus to design a NLP for a chaotic signal, Haykin and Li [25] pursued the idea of dynamical reconstruction from Takens' embedding theorem [26], [27]. Effectively, Takens' theorem states that dynamical reconstruction of a system can be achieved using a single dimensional subspace of the actual system. Reconstruction of the original signal is possible provided that,

$$N \geq 2d + 1 \qquad (8)$$

where d is the order, or dimension, of the dynamical system, which can be estimated using the maximum likelihood correlation dimension [28]. It should be noted that equation (8) is a sufficient, but not necessary condition for dynamical reconstruction [11]. Takens' embedding theorem actually permits the use of any τ as long as the observed time series is infinitely long. Unfortunately, in practice there is access only to finite data sequences. Therefore, τ

should be chosen so that it is large enough for $x(t)$ and $x(t-\tau)$ to be independent enough of each other, so that they serve as (independent) coordinates of the reconstruction space, *but*, not so independent as to have no correlation with each other: if the gap is too large, chaos makes $x(t)$ and $x(t-\tau)$ disconnected, or statistically independent [29].

Haykin and Li [25] used a method for estimating a suitable τ based on the definition of generalised dimensions of a dynamical system [30]. The approach adopted herein was to use the average mutual information [29]. It has been suggested [31] that the optimum embedding delay (for dynamical reconstruction) is at the first minimum of the average mutual information. In order to achieve dynamical reconstruction using a NLP, Haykin and Li actually advocated using embedding parameters somewhat different to those which follow on from Takens' embedding theorem: *i.e.* using equation (8) to select the embedding dimension, and selecting τ using, for example, the first minimum of the average mutual information. The embedding dimension that they suggested using to reconstruct the dynamics of a chaotic signal of dimension d is given as,

$$N \geq \tau D_E \tag{9}$$

where D_E is an estimate[1] of d, and τ is estimated using, for example, the first minimum of the average mutual information. Haykin and Li used an embedding delay of 1 sample together with this choice of embedding dimension.

IV. Nonlinear prediction of chaotic signals

In this section it will be demonstrated that to design the best NLP for a chaotic signal, in terms of NMSE, using Takens' embedding theorem criteria (or indeed Haykin and Li's criteria) to select the embedding parameters is not the most effective approach. Instead, as suggested by Casdagli [9], the best technique for selecting the embedding parameters is to use a trial and error approach. That is to say, N and τ should be varied until the minimum NMSE is achieved.

According to the discussion relating to Takens' embedding theorem in section 3 and the maximum likelihood correlation dimension estimate for the Lorenz data given above, a suitable embedding dimension for a nonlinear predictor of Lorenz data would be 6 or greater. Similarly, from the discussion relating to Takens' embedding theorem in section 3, a suitable embedding delay for a Lorenz predictor would be 3. However, it has been found, in terms of NMSE, that for a 1-step ahead NLP the optimum embedding dimension is 3, and the optimum embedding delay is 1 sample. This is illustrated in Figure 1, which shows Lorenz prediction results for a NRBFN predictor (NRBFNP), and an UNRBFN predictor (UNRBFNP), for various values of embedding dimension in Figure 1(a), and various values of embedding delay in Figure 1(b). Linear prediction results are shown as a performance benchmark. As can be seen from Figure 1, the NRBFNP consistently performed as well as or better than the UNRBFNP. In particular, for an embedding dimension of 3 and an embedding delay of 1 sample, the NRBFNP outperformed the UNRBFNP by more than 10dB. However, these results do not take into consideration the effect basis function parameter selection has on the performance of a radial basis function network predictor (RBFNP). The results discussed above were obtained using centres picked at random from the training data, as described in section 2, and the kernel width used was calculated as described in equation (4). It has been found that kernel width has a critical impact on the performance of a RBFNP, as will now be discussed.

[1]Estimated using for example the correlation dimension.

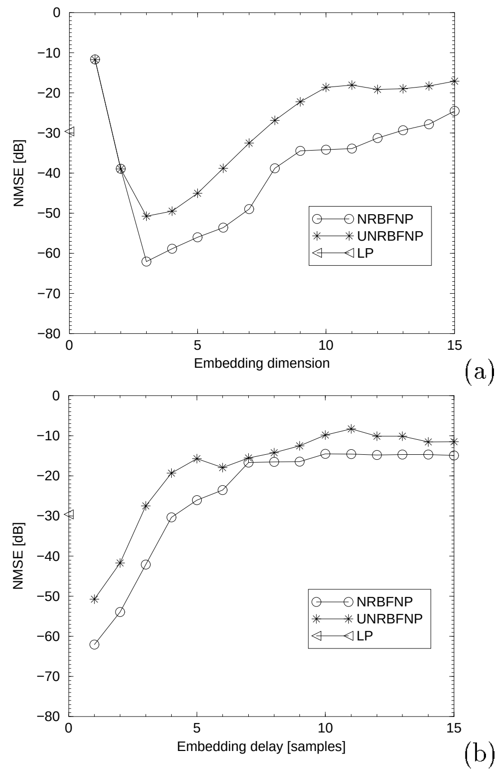

Fig. 1. *1-step ahead prediction of Lorenz data using a NRBFNP and an UNRBFNP, each with 100 kernels, for (a) different values of embedding dimension with an embedding delay of 1 sample, and for (b) different values of embedding delay with an embedding dimension of 3. Linear 1-step ahead prediction results are shown for a 30 tap LP, with a 1 sample delay between each tap. Validation data set results are shown for each predictor, and the training length used in each case was 6000 samples.*

Figure 2 shows Lorenz prediction results for a NRBFNP and an UNRBFNP, using different sets of kernel centres. A different random number generator seed corresponds to a different sequence of uniform random numbers, and thus to a different set of centres, see section 3. For each set of centres, a new set of RBFNP output layer weights was obtained. The same set of centres were used by the NRBFNP as for the UNRBFNP, for a given random number generator seed. The results in Figure 2(a) are for a kernel width calculated as in equation (4), those in Figure 2(b) are for twice this value of kernel width. A separate set of output layer weights was obtained for each value of kernel width. Observing the results in Figure 2, the importance of kernel width selection is highlighted. Focusing on the results in Figure 2(a) for the smaller of the 2 kernel widths considered it can be seen that the NRBFNP performed better in terms of NMSE than the UNRBFNP, and was less sensitive to changes in the position of the kernel centres. However, by using a larger kernel width the difference in NMSE performance was slightly reversed, and the difference in sensitivity to kernel centres selection vanished.

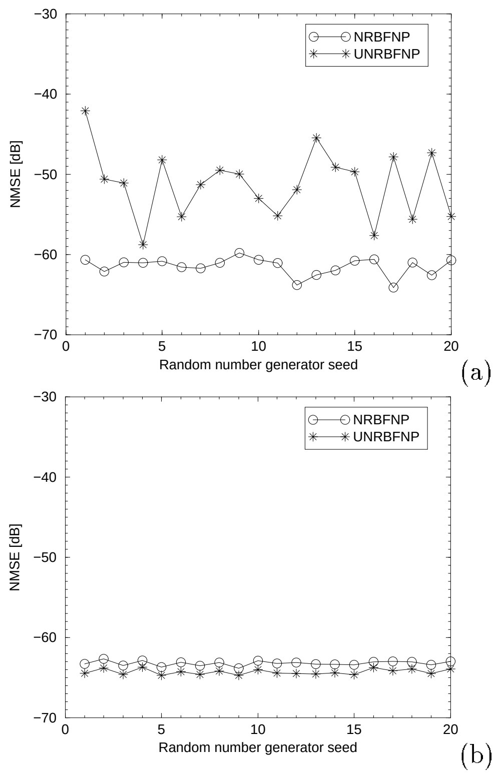

Fig. 2. *1-step ahead RBFN prediction of Lorenz data using different sets of kernel centres with (a) a kernel width calculated using equation (4), and (b) a kernel width twice the value of that used in (a). An embedding dimension of 3, 100 kernels, an embedding delay of 1 sample, and a training length of 10,000 samples were used by the RBFNP's. Validation data set results are shown.*

To further investigate the critical impact width selection can have on the performance of a RBFNP, Lorenz prediction results are shown in Figure 3 for a range of kernel widths. Different kernel widths were obtained as follows: a kernel width was calculated using equation (4), and this was multiplied by different width multiplication factors. The results in Figure 3 show that even although the UNRBFNP performed marginally ($\leq$1dB) better than the NRBFNP around a multiplication factor of 2, the NRBFNP was less sensitive to kernel width selection as its NMSE performance did not vary over as large a range as that for the UNRBFNP.

The importance of kernel width selection for RBFNP's is now further demonstrated using 2 other chaotic time series: Ikeda map data [29] and laser data [30]. The results in Figures 3 and 4 show that for each signal analysed the NRBFNP is less sensitive to kernel width selection than the UNRBFNP. This result supports the discussion on NRBFN's presented by others [4], [5], [3]. The results also demonstrate that for each signal analysed the best

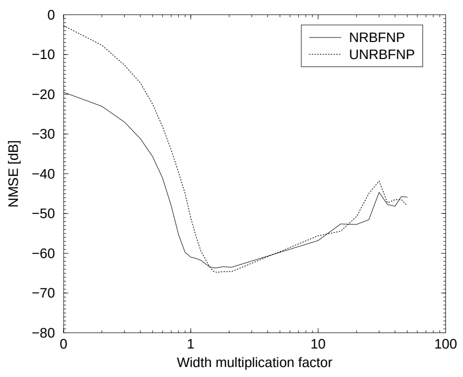

Fig. 3. *1-step ahead RBFN prediction of Lorenz data using different kernel widths for a fixed set of kernel centres. The kernel width was calculated using equation (4), and this was multiplied by different width multiplication factors. A separate set of output layer weights was obtained for each kernel width. An embedding dimension of 3, 100 kernels, an embedding delay of 1 sample, and a training length of 10,000 samples were used by the RBFNP's. Validation data set results are shown.*

prediction performance is achieved using different width multiplication factors, so there appears to be no optimal choice for the kernel width multiplication factor for the 3 signals considered: for any one of these signals the width which resulted in the best prediction performance would need to be searched for. For each case analysed in Figures 3 and 4 the NRBFNP consistently performed better than the UNRBFNP up to and including the point where the NMSE started to degrade with increasing kernel width, except for the Lorenz data. For the case of the Lorenz data the NRBFNP performed better than the UNRBFNP up to a width multiplication factor of 1.5. The UNRBFNP only performed marginally better ($\leq$1dB) than the NRBFNP between width multiplication factors of 1.5 and 1.6 before the NMSE started to degrade with increasing kernel width. The above results demonstrate that a NRBFNP is better than an UNRBFNP for predicting the chaotic signals considered.

V. Recursive prediction

As already discussed in section 3, Haykin and Li [25] pursued the idea of dynamical reconstruction when it came to choosing suitable embedding parameters for a NLP of a chaotic signal. They also advocated using recursive prediction to test the generalisation properties of their NLP's. Recursive prediction is performed by first of all training a predictor to obtain the mapping in equation (6). Then the trained predictor is given one input vector from the available data. From then on the output of the predictor is fed back to its input, and the system becomes autonomous. Haykin and Principe [11] suggested using recursive prediction as a pragmatic approach for testing how well a 1-step ahead predictor had managed to model the underlying dynamics of a chaotic signal. If the predictor is successful at modelling the underlying dynamics of the chaotic signal, then the predictor's output, in recursive prediction mode, should satisfy the two conditions [11] (i) short term behaviour - the reconstructed signal should closely follow the

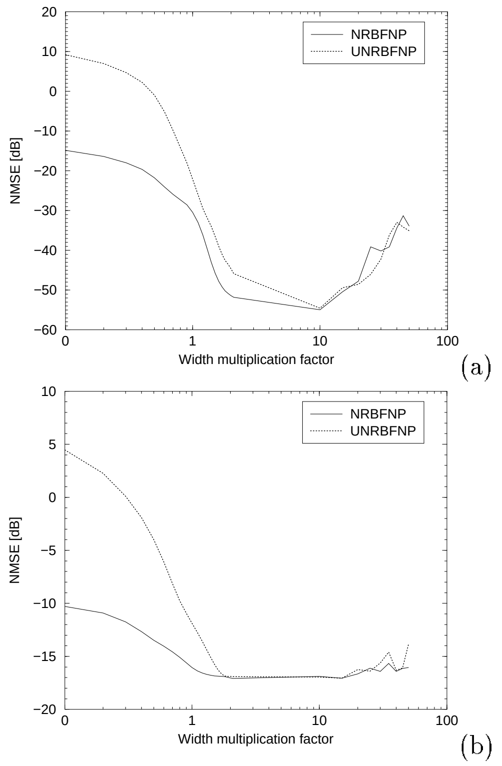

Fig. 4. *1-step ahead RBFN prediction of (a) Ikeda map data and (b) laser data, using different kernel widths for a fixed set of kernel centres. The kernel width was calculated using equation (4), and this was multiplied by different width multiplication factors. A separate set of weights was obtained for each kernel width. An embedding dimension of 4, 100 kernels, and an embedding delay of 1 sample were used for the prediction of both time series. A training length of 10,000 samples was used for the Ikeda map, and one of 4,000 samples was used for the laser data. Validation data set results are shown.*

original time series until the prediction horizon; (ii) long term behaviour - the dynamic invariants of the reconstructed signal should match those of the original.

Haykin and Principe [11] investigated whether it was possible for a RBFNP to recursively predict and capture the underlying dynamics of chaotic Lorenz data corrupted by additive white Gaussian noise. They reported that this was not possible without the use of regularisation [17], [10]. The signal to noise ratio (SNR),

$$\mathrm{SNR} = 10\log_{10}\left[\frac{\sigma^2_{\text{signal}}}{\sigma^2_{\text{noise}}}\right] \qquad (10)$$

where σ^2_{signal} is the variance of the signal of interest (in this case the Lorenz data), and σ^2_{noise} is the variance of the noise, used by Haykin and Principe was 25dB. The primary aim of this section is to investigate if a NRBFNP is able to recursively predict (and also to capture the underlying dynamics of) noisy Lorenz data, as normalisation is simpler to implement than regularisation. Of additional interest was an investigation into the relationship between a NLP's ability to capture the underlying dynamics of a chaotic signal and the NMSE performance of that NLP: in other words, would the network which learnt the underlying dynamics also have the best NMSE prediction performance? Finally, also considered in this section is a comparison of using the 2 different criteria discussed in section 3 for dynamical reconstruction: the criteria which follows on from Takens' embedding theorem, equation (8), and Haykin and Li's criteria, equation (9).

To investigate if a NRBFNP could be used to recursively predict noisy Lorenz data, white Gaussian noise was added to the Lorenz data described in section 4, the SNR was 25dB. Haykin and Principe used the following parameters in their regularised RBFNP: an embedding dimension of 20, an embedding delay of 1 sample, and 400 kernels. The selection of the embedding parameters was done in accordance with Haykin and Li's criteria, as given in equation (9). A NRBFNP with the same parameters was used to recursively predict the noisy Lorenz data. It was also investigated whether it would be possible to use a NRBFNP with a smaller embedding dimension, which was selected using Takens' embedding theorem, to recursively predict the data. A maximum likelihood correlation dimension estimate for the noisy Lorenz data was estimated, using 35,000 samples, to be 2.38. This gives a minimum embedding dimension of 6, for dynamical reconstruction using Takens' theorem. The mutual information plot for the noisy Lorenz data revealed that the first minimum was at a delay of 3 samples. A NRBFNP with an embedding dimension of 7, an embedding delay of 3 samples, and 400 kernels was also used to recursively predict the noisy Lorenz data. In addition, because of the good prediction results (see section 4) achieved using an embedding delay of 1 sample, a NRBFNP with an embedding dimension of 7 and 400 kernels was used with an embedding delay of 1 sample. Recursive prediction results are shown for all three networks in Figure 5.

TO APPEAR (a)
TO APPEAR (b)
TO APPEAR (c)

Fig. 5. *Recursive prediction of noisy Lorenz data (SNR=25dB) using (a) a NRBFNP with an embedding dimension of 7 and embedding delay of 1 sample, (b) a NRBFNP with an embedding dimension of 7 and embedding delay of 3 samples, and (c) a NRBFNP with an embedding dimension of 20 and embedding delay of 1 sample. Each NRBFNP used 400 kernels and a training length of 2000 samples. In each case the recursive prediction was initialised with the first vector from the testing data set.*

As can be seen from Figure 5(a) the NRBFNP with an embedding dimension of 7 and an embedding delay of 1 sample did not appear to capture much of the original signal's detail after the prediction horizon, which was approximately 50 samples: after 50 samples the NRBFNP's output merely seemed to oscillate between a positive value and a negative value. In contrast, both of the other NRBFNP's managed to capture the 3 and 4 peak detail observable in the original signal after the prediction horizon, which was approximately 20 samples for each of these NRBFNP's. To try to obtain a more quantitative idea of how each network performed, the maximum likelihood correlation dimension was used as follows. Each predictor network was trained using a training length of 2000 samples, and was initialised with the first vector from the testing data set. Each NRBFNP was then used to generate 35,000 samples, and the maximum likelihood correlation dimension was estimated for each data set, as tabulated in Table I. Training, testing, and validation data set NMSE's are also shown for each NRBFNP, for a training length of 2000 samples.

N	K	**Correlation dimension estimate**	**Training NMSE [dB]**	**Testing NMSE [dB]**	**Validation NMSE [dB]**
7	1	2.03	-23.63	-20.97	-20.85
7	3	2.28	-21.59	-19.16	-17.92
20	1	2.25	-21.43	-19.64	-18.78

TABLE I

Recursive prediction of noisy Lorenz data (SNR=25dB), using NRBFNP's.

As can be seen from Table I the NRBFNP with an embedding dimension of 7 and an embedding delay of 1 sample produced the recursively predicted time series with the worst correlation dimension estimate with respect to the 2.38 value of the original noisy data. The NRBFNP with an embedding dimension of 7 and embedding delay of 3 samples produced the time series with the closest correlation dimension estimate to that of the original time series. This can be explained by considering the evolution of the NRBFNP input vectors in state space. Increasing the embedding delay has the effect of "unfolding" the attractor in state space, which reduces the likelihood that noise will cause any vector to erroneously evolve (or jump) to the wrong part of the attractor. Avoiding such erroneous evolution eventualities results in correctly capturing the underlying dynamics of the noisy Lorenz data. Therefore, whilst an embedding delay of 1 sample is a better choice for the general prediction problem, an embedding delay of 3 samples is preferred for the recursive prediction problem. A reason why the correlation dimension estimate was slightly poorer for the NRBFNP with an embedding dimension of 20, than for the NRBFNP with an embedding dimension of 7 and embedding delay of 3 samples, is that this embedding dimension is actually too large, and as a result the additive Gaussian noise has degraded the quality of the dynamical reconstruction [11]. It should also be noted that the network which best managed to capture the underlying dynamics was not the network with the best NMSE values.

Owing to the critical impact kernel width was shown to have on the performance of a 1-step ahead RBFNP, as discussed in section 4, recursive prediction was attempted using an UNRBFNP for a range of kernel widths. Contrary to the work reported by Haykin and Principe [11], it was found that an UNRBFNP was able to recursively predict and capture the underlying dynamics of the noisy Lorenz data for certain values of kernel width (at width multiplication factors of 3, 4, 6 and 7). However, in order to find kernel widths which would sustain recursive prediction, a trial and error approach would be required, as no obvious criterion for kernel width selection. This is in contrast to the NRBFNP which achieved dynamical reconstruction using a kernel width calculated from equation (4). Moreover, based on the D_{ML} estimate, the UNRBFNP was not able to capture the underlying dynamics as well as the NRBFNP.

It has been demonstrated that a NRBFNP can be used, with embedding parameters selected using the maximum likelihood correlation dimension estimate and average mutual information, to recursively predict noisy Lorenz data. The advantage of using a NRBFNP instead of a regularised RBFNP (which was the technique used by Haykin and Principe to achieve recursive prediction) is that implementing a NRBFNP is easier: in designing a regularised RBFNP there is an additional parameter, the regularisation parameter, to find a suitable value for. It has also been shown that Takens' criteria for dynamical reconstruction is a better choice than Haykin and Li's criteria, as not only does it result in the choice of a smaller embedding dimension, it also results in designing a network which has been shown to more accurately model the underlying dynamics of a chaotic time series. It was noted that designing a NLP to capture the underlying dynamics of a chaotic signal is not always consistent with obtaining the best NMSE prediction performance. Finally, it was shown that an UNRBFNP was able to recursively predict the noisy Lorenz data for certain kernel width values. However, determining suitable kernel widths for the UNRBFNP to achieve this was found to be a trial and error procedure, in contrast to the kernel width selection technique for the NRBFNP. Furthermore, based on D_{ML} estimates, the UNRBFNP was not able to capture the underlying dynamics as well as the NRBFNP.

VI. Conclusions

The normalised RBFN has been re-examined. The effective and robust nature of the structure and training strategy proposed here was demonstrated using the problem of chaotic signal prediction. It was shown that normalisation can be used to make a RBFN less sensitive to kernel width selection. Normalisation was shown to be a simple and effective alternative to regularisation for the task of recursively predicting and capturing the dynamics of a chaotic signal corrupted by additive white noise. Moreover, Takens' embedding criteria were shown to be preferred to that of Haykin and Li's, as they resulted in both a smaller embedding dimension, and a NRBFN which was able to more accurately model the underlying dynamics of a chaotic signal. It was established that designing a NLP to capture the underlying dynamics of a chaotic signal is not always consistent with obtaining the best NMSE prediction performance. To conclude, the above results point towards the NRBFN being a useful tool for the processing of chaotic signals. Furthermore, this work has attempted to address some of the concerns raised about NRBFN's in the literature.

References

[1] J. Moody and C. J. Darken, "Fast learning in networks of locally-tuned processing units", Neural Computation, Vol. 1, No. 2, 1989, pp. 281-294.

[2] M. Benaim, "On the functional approximation with normalized Gaussian units", Neural Computation, Vol. 6, 1994, pp. 314-333.

[3] G. Bugmann, "Normalized Gaussian radial basis function networks", Neurocomputing, Vol. 20, No. 1-3, 1998, pp.97-110.

[4] R. Shorten and R. Murray-Smith, "Side effects of normalising radial basis function networks", International Journal of Neural Systems, Vol. 7, May 1996, pp. 167-179.

[5] H. W. Werntges, "Partitions of unity to improve neural function approximators", IEEE International Conference on Neural Networks, San Francisco, California, March 28 - April 1, 1993, pp. 914-918.

[6] I. Cha and S. A. Kassam, "Interference cancellation using radial basis function networks", Signal Processing, Vol. 47, 1995, pp. 247-268.

[7] A. O. Steinhardt, "Householder transforms in signal processing", IEEE Acoustics, Speech and Signal Processing Magazine, Vol. 5, July 1998, pp. 4-12.

[8] S. Haykin, Adaptive Filter Theory, Prentice Hall, 1996, 3rd Edn.

[9] M. Casdagli, "Nonlinear prediction of chaotic time series", Physica D, Vol. 35, 1989, pp. 335-356.

[10] T. Poggio, "Networks for approximation and learning", Proc. IEEE, Vol. 78, September 1990, pp. 1481-1497.

[11] S. Haykin and J. Principe, "Making sense of a complex world", IEEE Signal Processing Magazine, Vol. 15, May 1998, pp. 66-81.

[12] D. S. Broomhead and D. Lowe, "Multivariable functional interpolation and adaptive networks", Complex Systems, Vol. 2, 1988, pp. 321-355.

[13] S. Chen, A. Billings, C. F. N. Cowan, and P. M. Grant, "Nonlinear systems identification using radial basis functions", International Journal of Systems Science, Vol. 21, December 1990, pp. 2513-2539.

[14] J. A. Leonard and M. A. Kramer, "Radial basis function networks for classifying process faults", IEEE Control Systems Magazine, Vol. 11, April 1991, pp. 31-38.

[15] S. Chen, C. F. N. Cowan, and P. M. Grant, "Orthogonal least squares learning algorithm for radial basis function networks", IEEE Transactions on Neural Networks, Vol. 2, March 1991, pp. 302-309.

[16] A. Sherstinsky and R. W. Picard, "On training Gaussian radial basis functions for image coding", Vision and Modelling Group Technical Report 188, MIT Media Laboratory, Cambridge Massachusetts, February 1992.

[17] S. Haykin, Neural Networks: A Comprehensive Foundation, Prentice Hall, 1994.

[18] F. Miguel and A. Acosta, "Radial basis function and related models: an overview", Signal Processing, Vol. 45, 1995, pp. 37-58.

[19] M. J. L. Orr, "Introduction to radial basis function networks", http:www.anc.ed.ac.uk/mjo/intro/intro.html, 1996.

[20] B. Mulgrew, "Applying radial basis functions", IEEE Signal Processing Magazine, Vol. 13, March 1996, pp. 50-65.

[21] C. M. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, 1996.

[22] L. Xu, A. Kryzyzak, and A. Yuille, "On radial basis function nets and kernel regression: statistical consistency, convergence rates, and receptive field size", Neural Networks, Vol. 7, No. 4, 1994, pp. 609-628.

[23] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, Numerical Recipes in C: The art of scientific computing, Cambridge University Press, 1992, 2nd Edn.

[24] A. S. Weigend and N. A. Gershenfeld (Eds), Time Series Prediction: Forecasting the Future and Understanding the Past, Addison Wesley, 1994.

[25] S. Haykin and X. B. Li, "Detection of signal in chaos", Proc. IEEE, Vol. 83, January 1995, pp. 95-122.

[26] F. Takens, "On the numerical determination of the dimension of an attractor", in D. Rand and L.S. Yong (Eds), *Dynamical Systems and Turbulence, Warwick 1980. Lecture Notes in Mathematics, Vol. 898*, Springer-Verlag, 1981.

[27] T. Sauer, J. A. Yorke, and M. Casdagli, "Embedology", Journal of Statistical Physics, Vol. 65, No. 3/4, 1991, pp. 579-618.

[28] J. C. Schouten, F. Takens, and C. M. van den Bleek, "Estimation of the dimension of a noisy attractor", Physical Review E, Vol. 50, September 1994, pp. 1851-1861.

[29] H. D. I. Abarbanel, Analysis of observed chaotic data, Springer-Verlag, 1996.

[30] F. J. Pineda and J. C. Sommerer, "Estimating generalized dimensions and choosing time delays: a fast algorithm", in A. Weigend and N. Gershenfeld (Eds), *Forecasting the Future and Understanding the Past*, Addison-Wesley, 1994, pp. 367-385.

[31] A. M. Fraser and H. L. Swinney, "Independent coordinates for strange attractors from mutual information", Physical Review A, Vol. 33, February 1986, pp. 1134-1140.

[32] K. T. Alligood, T. D. Sauer, and J. A. York, Chaos: An Introduction To Dynamical Systems, Springer-Verlag, 1997.

[33] C. Chinrungrueng and C. H. Sequin, "Optimal adaptive K-means algorithm with dynamic adjustment of learning rate", IEEE Transactions on Neural Networks, Vol. 6, January 1995, pp. 157-169.