A Unifying View of Some Training Algorithms for Multilayer Perceptrons with FIR Filter Synapses

Andrew Back*, Eric A. Wan**, Steve Lawrence*, Ah Chung Tsoi*
*Department of Electrical and Computer Engineering,
University of Queensland, St. Lucia, Queensland 4072. Australia.
**Department of Electrical Engineering and Applied Physics
Oregon Graduate Institute of Science & Technology
P.O. Box 91000, Portland, Oregon 97291, USA
back@s1.elec.uq.oz.au, ericwan@eeap.ogi.edu
lawrence@s1.elec.uq.oz.au, act@s1.elec.uq.oz.au

Abstract– Recent interest has come about in deriving various neural network architectures for modelling time-dependent signals. A number of algorithms have been published for multilayer perceptrons with synapses described by finite impulse response (FIR) and infinite impulse response (IIR) filters (the latter case is also known as *Locally Recurrent Globally Feedforward Networks***). The derivations of these algorithms have used different approaches in calculating the gradients, and in this note, we present a short, but unifying account of how these different algorithms compare for the FIR case, both in derivation, and performance. New algorithms are subsequently presented. Simulation results have been performed to benchmark these algorithms. In this note, results are compared for the Mackey-Glass chaotic time series against a number of other methods including a standard multilayer perceptron, and a local approximation method.**

INTRODUCTION

As a means of capturing time-dependent signals in a nonlinear framework, multilayer perceptrons (MLPs) with synapses described by filters have recently been proposed [1, 2, 17]. These approaches replace the traditional scalar synaptic weights with finite impulse response (FIR) filters commonly used in digital filter theory. The architecture can be considered an extension of earlier work in which time delays were incorporated as a means of capturing time-dependent input information. For example, in the Time Delay Neural Network used by Waibel et al [20], the outputs of a layer in a feedforward network are buffered several time steps and then fed fully connected to the next layer. Lapedes and Farber's [10] use of a time-window as the input to a multilayer network for applications in time series prediction is equivalent to one layer of time delay synapses at the input. FIR networks provide a more general model for distributed time representations.

An algorithm for training networks having FIR synapses was first published by Wan [17]. A similar algorithm for the same network as well as the case for IIR synapses was published by Back and Tsoi [1, 2]. We focus on these algorithms in this paper, comparing their derivations and presenting a brief, but unifying view of them. Related work which has been presented in [4, 6, 7, 11] and [14] among others, will not be considered here. Our aim is to compare the forms of the training algorithms, and to provide an indication of how they perform on some practical prediction problems. In this brief summary, we show only one set of results, the Mackey-Glass chaotic time series which allows us to easily highlight the differences in performances of various methodologies for prediction of nonlinear time series.

The network architecture is defined below:

Definition 1. *An FIR MLP of size* (L, n_w) *with* $N_0, N_1, ..., N_L$ *nodes per layer, is defined by*

$$z_k^l(t) = f\left(\hat{x}_k^l(t)\right) \tag{1}$$

$$\hat{x}_k^l(t) = \sum_{i=1}^{N_l} \hat{y}_{ik}^l(t) \tag{2}$$

$$\hat{y}_{ik}^l(t) = c_{ik}^l y_{ik}^l(t) \quad (3)$$

$$y_{ik}^l(t) = W_{ik}^l(q^{-1}) z_i^{l-1}(t), \quad (4)$$

where each neuron i in layer l has an output at time t of $z_i^l(t)$; a layer consists of N_l neurons ($l = 0$ denotes the input layer, and $l = L$ denotes the output layer, $z_{N_l}^l = 1.0$ may be used for a bias); $\hat{y}_{ik}^l(t)$ is the output of a synapse connecting from neuron i in the previous layer to neuron k in layer l; c_{ik}^l is a synaptic gain; and $f(\cdot)$ is a sigmoid function typically evaluated as $\tanh(\cdot)$. *An FIR synapse is represented by $W_{ik}^l(q^{-1}) = \sum_{j=0}^{n_w} w_{ikj}^l(q^{-j})$ where w_{ikj}^l correspond to the variable coefficients, and q^{-1} is a delay operator ($q^{-1}z(t) \triangleq z(t-1)$), and n_w is the number of delays.*

The algorithms use first order stochastic gradient descent to minimize an error function. We define the instantaneous performance criteria

$$\mathcal{E}(t) = \frac{1}{2}\sum_{k=1}^{N_L} e_k^2(t) = \frac{1}{2}\sum_{k=1}^{N_L} \left(d_k(t) - z_k^L(t)\right)^2 \quad (5)$$

where $d_k(t)$ is the desired output at time t, and the sum is taken over the output neurons. The total error or cost is given by summing the instantaneous error over all T time steps in a training sequence

$$\mathcal{E}_T = \sum_{t=0}^{T} \mathcal{E}(t). \quad (6)$$

The different forms of the training algorithms for FIR networks differ in the manner in which the gradients are calculated and on whether the instantaneous or total error is used in the calculations.

GRADIENT COMPUTATION IN FIR SYNAPSES USING AN INSTANTANEOUS COST FUNCTION

An algorithm for updating the weights in an FIR network may be obtained by considering the instantaneous error $\mathcal{E}(t)$ [1, 17]. The weight changes can be adjusted using a simple gradient method

$$w_{ikj}^l(t+1) = w_{ikj}^l(t) + \Delta w_{ikj}^l(t) \quad (7)$$

$$c_{ik}^l(t+1) = c_{ik}^l(t) + \Delta c_{ik}^l(t) \quad (8)$$

$$\Delta w_{ikj}^l(t) = -\eta \frac{\partial \mathcal{E}(t)}{\partial w_{ikj}^l(t)} \quad (9)$$

$$= -\eta \frac{\partial \mathcal{E}(t)}{\partial \hat{x}_k^l(t)} \frac{\partial \hat{x}_k^l(t)}{\partial w_{ikj}^l(t)} \quad (10)$$

$$\Delta c_{ik}^l(t) = -\eta \frac{\partial \mathcal{E}(t)}{\partial c_{ik}^l(t)}, \quad (11)$$

$$= -\eta \frac{\partial \mathcal{E}(t)}{\partial \hat{x}_k^l(t)} \frac{\partial \hat{x}_k^l(t)}{\partial c_{ikj}^l(t)} \quad (12)$$

where η is the learning rate. A derivation of the partial terms is given in [2]. In the derivation, it is necessary to define a secondary variable $\delta_k^l(t) = -\frac{\partial \mathcal{E}(t)}{\partial \hat{x}_k^l(t)}$. If we consider only the gradient at the exact time t, then we have

Algorithm IC-1 Instantaneous Cost - Instantaneous Gradient

$$\delta_k^l(t) = f'\left(\hat{x}_k^l(t)\right) \sum_{m=1}^{N_{l+1}} \delta_m^{l+1}(t) c_{km}^{l+1} w_{km0}^{l+1}. \tag{13}$$

This can be considered an exact instantaneous gradient. We have not taken into account the performance surface over time. This is the method adopted in [1, 2]. Note the δ terms are essentially calculated using standard backpropagation through the w_{km0} taps; the rest of the coefficients in the FIR synapse are ignored, since we only assume a relationship between $z_k^l(t)$ and $\hat{y}_{km}^{l+1}(t)$ instantaneously at time t.

A novel form is achieved if we calculate the gradient over a short time period by delaying the calculation of the gradient until all contributions from feedforward delay elements can be combined.

Algorithm IC-2 Instantaneous Cost - Accumulated Gradient

$$\begin{aligned}
\delta_k^l(t) &= f'\left(\hat{x}_k^l(t)\right) \sum_{m=1}^{N_{l+1}} \sum_{d=0}^{n_w} c_{km}^{l+1} w_{kmd}^{l+1}(q^{-d}) \delta_m^{l+1}(t) \\
&= f'\left(\hat{x}_k^l(t)\right) \sum_{m=1}^{N_{l+1}} \sum_{d=0}^{n_w} c_{km}^{l+1} w_{kmd}^{l+1} \delta_m^{l+1}(t-d) \\
&= f'\left(\hat{x}_k^l(t)\right) \sum_{m=1}^{N_{l+1}} c_{km}^{l+1} W_{km}^{l+1}(q^{-1}) \delta_m^{l+1}(t).
\end{aligned} \tag{14}$$

This is similar to the second algorithm proposed by Wan in [17] (discussed in a subsequent section of this paper). In this case, we have the backpropagated error being obtained from a backward *filter* and all coefficients in the FIR synapse have an influence on the δ value.

For both cases, the final update equations for the FIR MLP are

$$w_{ikj}^l(t+1) = w_{ikj}^l(t) + \eta \delta_k^l(t) c_{ik}^l z_i^{l-1}(t-j) \tag{15}$$

$$c_{ik}^l(t+1) = c_{ik}^l(t) + \eta \delta_k^l(t) W_{ik}^l(q^{-1}) z_i^{l-1}(t), \tag{16}$$

where $\delta_k^l(t)$ may be computed by one of the two methods described above. We will discuss the relative performance of the different methods in the results section.

GRADIENT COMPUTATION IN FIR SYNAPSES USING A TOTAL COST FUNCTION

This section reviews the algorithms derived by Wan in [17, 18]. Gradient adaptation is based on the *total* squared error over the entire sequence of inputs, as opposed to the instantaneous error measure used previously. This should not be confused with the fact that in all cases, we use an on-line updating scheme which makes use of error measurement computed for that particular time instant.

Fundamentally, the weight changes in eqns. (9) and (11) are replaced by

$$\Delta w_{ikj}^l(t) = -\eta \frac{\partial \mathcal{E}_T}{\partial w_{ikj}^l(t)}, \qquad \Delta c_{ik}^l(t) = -\eta \frac{\partial \mathcal{E}_T}{\partial c_{ik}^l(t)}. \tag{17}$$

We have simply substituted the total error $\mathcal{E}_T$ for the instantaneous error $\mathcal{E}(t)$. In this case an expression for δ is obtained by maintaining the dependence over all values

of the input sequence. Derivations given in [19] leads to the following algorithms. Algorithm TC-1 is very inefficient for networks with more than two layers. Algorithm TC-2 on the other hand, uses the same update equations (15) and (16) as before. In this case we derive a slightly different equation for the δ term.

Algorithm TC-1 **Total Cost - Instantaneous Gradient**

$$\frac{\partial \mathcal{E}(t)}{\partial w_{ikj}^l(t)} = \sum_{n=0}^{n_w} c_{ik}^l z_i^{l-1}(t-n) f'(\hat{x}_k^l(t-n)) \sum_{m=1}^{N_{l+1}} \delta_m^{l+1}(t) c_{km}^{l+1} w_{kmn}^{l+1}(t) \quad (18)$$

$$\frac{\partial \mathcal{E}(t)}{\partial c_{ik}^l(t)} = \sum_{n=0}^{n_w} y_i^l(t-n) f'(\hat{x}_k^l(t-n)) \sum_{m=1}^{N_{l+1}} \delta_m^{l+1}(t) c_{km}^{l+1}(t) w_{kmn}^{l+1}(t) \quad (19)$$

Algorithm TC-2 **Total Cost - Temporal Backpropagation**

$$\delta_k^l(t) = f'(\hat{x}_k^l(t)) \sum_{m=1}^{N_{l+1}} c_{km}^{l+1} W_{km}^{l+1}(q^{+1}) \delta_m^{l+1}(t). \quad (20)$$

Note that in this case, Algorithm TC-2 needs to be delayed n_w time steps to maintain causality. It can be seen as very similar to Algorithm IC-2, though the evaluation of δ and w terms occurs at different times (cf. eqn. (14)). In this algorithm, the backward filtering comes about directly as a result of using the *total* cost function over time, thereby necessitating the accumulation of gradient information. In Algorithm IC-2, gradient computations are accumulated over time after initially considering an instantaneous gradient.

SIMULATION RESULTS

As a means of comparing the different algorithms, we present some preliminary results for the application of the neural network algorithms to some time series prediction tasks. In this extended summary, only the results obtained in modelling the Mackey-Glass delay-differential equation[1] are presented, due to the widespread interest in using it as a benchmark. In the full paper, results pertaining to other time series prediction problems are considered, specifically:

1. Prediction of Mackey-Glass chaotic time series,
2. Prediction of Laser data as used in Santa Fe Time Series Prediction Competition.
3. Nonlinear speech prediction
4. Financial time series prediction

The algorithms discussed above are each trained on the data for the Mackey-Glass time series. In each case, multiple simulations were performed and the results averaged to obtain a reasonable indication of the networks performance. After some initial testing to determine suitable learning rates, we selected a specific learning rate which remained the same for each network when trained on a particular time series.

Our aim was to test two basic approaches to time series prediction, namely, the traditional approach of using past values of the time series directly, and secondly, the

[1]The Mackey-Glass [12] equation is described by $\dot{x}(t) = -bx(t) + \frac{ax(t-\tau)}{1+x(t-\tau)^{10}}$, where τ=30, a=0.2, and b=0.1.

approach of embedding the time series in a phase space, and using the delay coordinates as the vector of inputs[2]. This approach, proposed by Takens [13, 15] involves sampling the time series at some *delayed* time values to create a *delay coordinate* vector. This is sometimes referred to as a phase space.

As a means of comparing each algorithm, we benchmark their relative performances againsta windowed input MLP, and a local approximation method developed by Casdagli [5] (a version of the nearest neighbor method). Obviously, there are many variations in which this could have been done. Our intent is to provide a reasonable means of quickly assessing the performance of these algorithms which may provide a starting point for anyone interested in considering them further.

The work we present here consists of benchmarking each of the above algorithms on some representative time series as listed above. We consider three main cases:

- Single-step ahead prediction using a vector of past inputs (spaced one time unit apart)
- Multi-step ahead prediction using a delay coordinate (Takens) vector of past inputs (spaced τ time units apart, where τ is the delay parameter)
- Iterated-prediction problem, using past outputs of the model as inputs for future predictions[3].

In the simulations performed, we used delay coordinate vectors with 6 elements (D=6), and a time delay of $\tau = 6$. The order of FIR filters was $n_w = 5$. The results shown in Figures 1 and 2 are for the multi-step prediction problem, and the iterated-prediction problem using a prediction time-step of 6.

These results are interesting, in that they show, for the problem at hand, the FIR MLP structure appears to be better able to model the dynamics of the chaotic time-series. The multi-step ahead prediction performance for the test set shows that each model is able to do quite well. However, when we consider the more difficult problem of iterated-prediction, we observe that the networks with FIR synapses perform much better (see for example, the generated phase space plots in fig. 3).

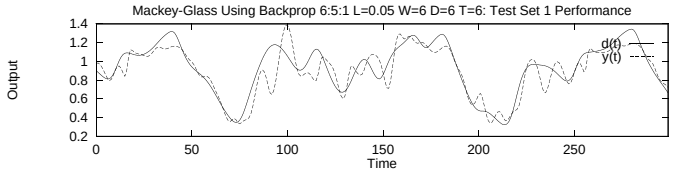
It is interesting also to observe the different behaviours of the algorithms possible for the FIR MLP model, indicating that while they may be better than other methods generally, there are differences between how the algorithms operate in practice. Results on the other simulation problems will be presented at the workshop.

CONCLUSIONS

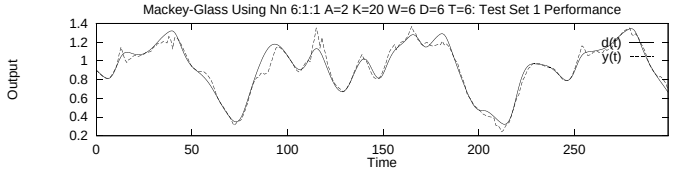
The aim of this brief note was to clarify some of the issues in calculating the gradients for multilayer perceptrons with FIR synapses. This contributes to a further understanding of these types of network architectures. Results in using these networks have shown promise for a variety of nonlinear signal prediction tasks and we look forward to continued activity in this area.

[2]In our models, we only have a single input. However this is equivalent to the case where, for example, a linear predictor or multilayer perceptron has a window of inputs. In our case, the window of each filter exists in each synapse already.

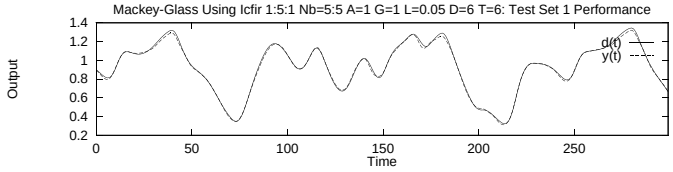
[3]Our approach is to allow the models to recursively predict further and further into the future, based on the initial predictions obtained. Therefore, if we allow the model to predict 6 time-steps into the future, and we wish to see how it performs out to 400 time-steps, we allow the model to use its shorter predictions to "bootstrap" itself out. This follows the conventions adopted by Lapedes and Farber [10] and Stokbro and Umberger [16].



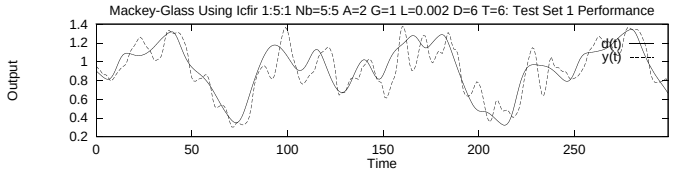
(a)



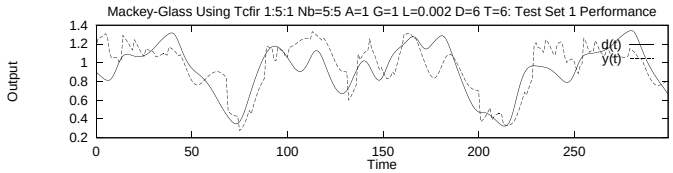
(b)



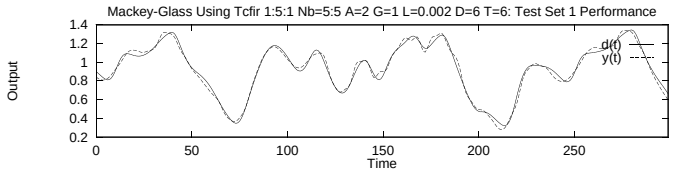
(c)



(d)

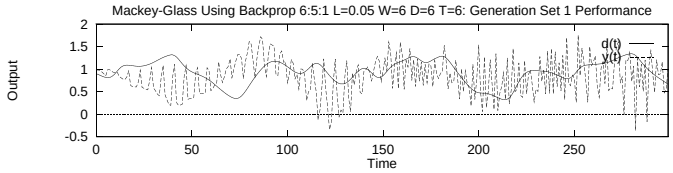


(e)

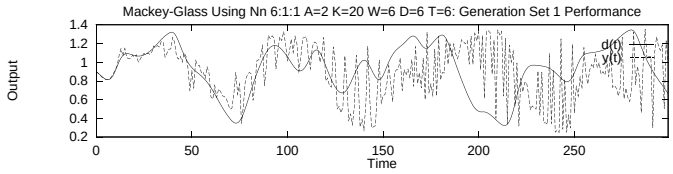


(f)

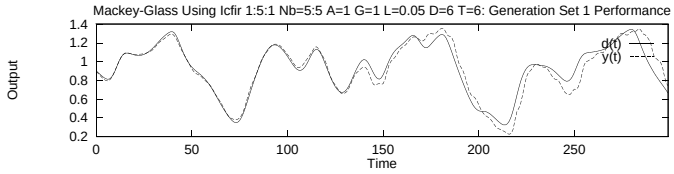
Figure 1: Test set performance on 6-step ahead prediction of the Mackey-Glass chaotic time series ($T = 30$). (a) Backpropagation (b) Nearest Neighbour ($k = 20$) (c) Algorithm IC1 (d) Algorithm IC2 (e) Algorithm TC1 (f) Algorithm TC2. (G=1 indicates synaptic gain is used, D is embedding delay, T is prediction time-step, W is input window (backpropagation only), L is learning rate, A is algorithm, Nb is FIR filter order).



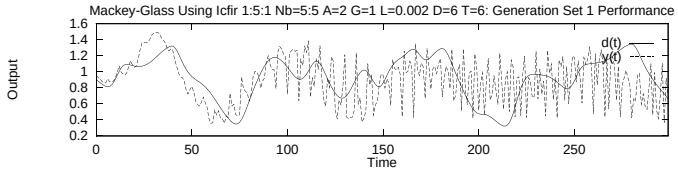
(a)



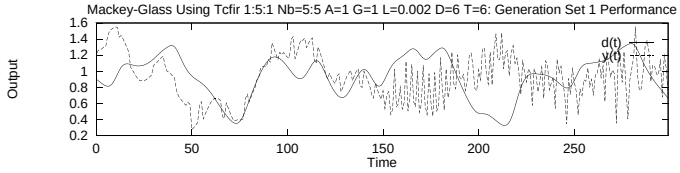
(b)



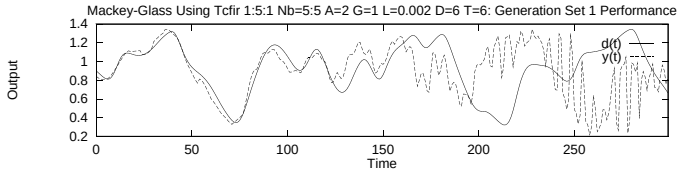
(c)



(d)



(e)



(f)

Figure 2: Iterated prediction performance on Mackey-Glass chaotic time series ($T = 30$). (a) Backpropagation (b) Nearest Neighbour ($k = 20$) (c) Algorithm IC1 (d) Algorithm IC2 (e) Algorithm TC1 (f) Algorithm TC2.

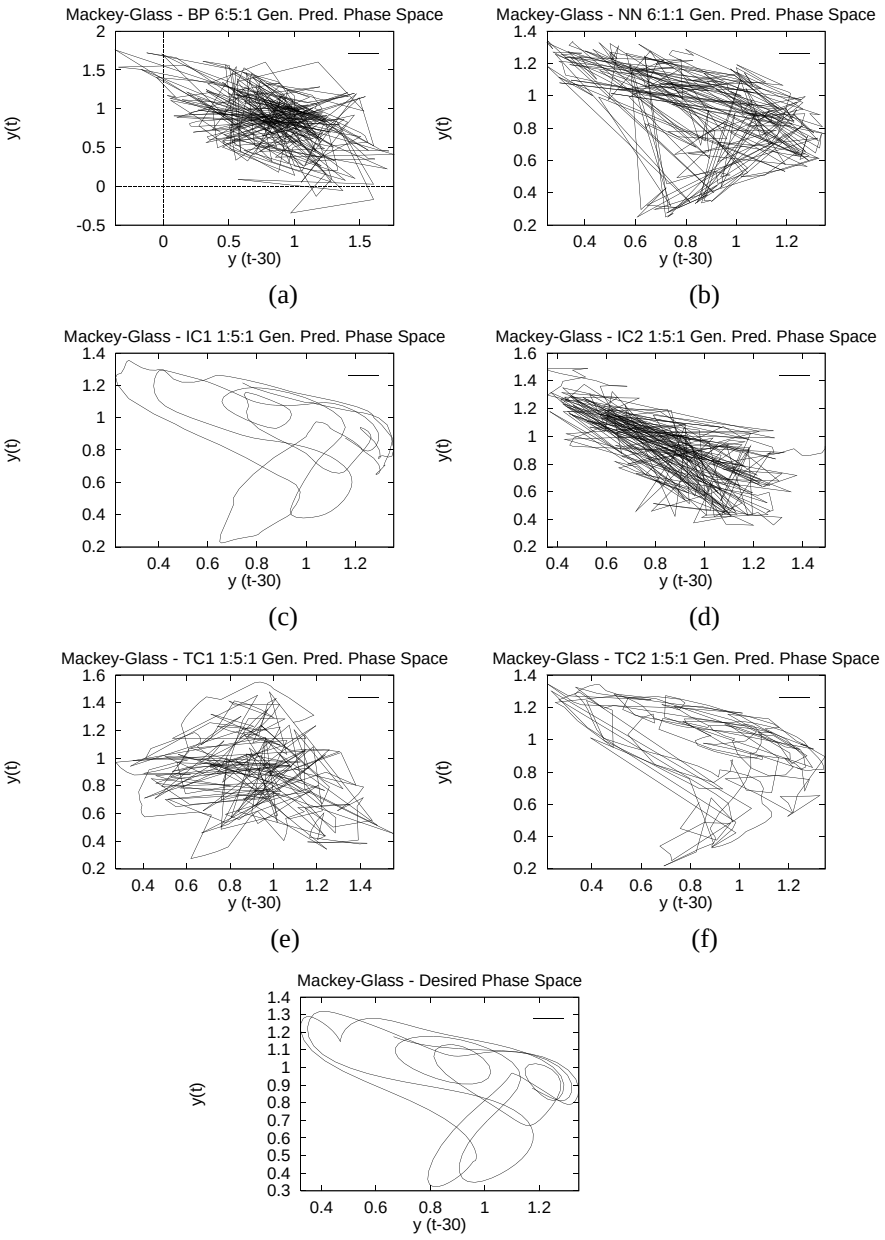


Figure 3: Phase space generation performance plotting $y(t)$ vs. $y(t-30)$ for Mackey-Glass chaotic time series ($T = 30$). (a) Backpropagation (b) Nearest Neighbour ($k = 20$) (c) Algorithm IC1 (d) Algorithm IC2 (e) Algorithm TC1 (f) Algorithm TC2. (g) Desired phase space.

Acknowledgements.

The first author acknowledges financial support from the Australian Research Council. The third author acknowledges support from the Australian Research Council and Australian Telecommunications and Electronics Research Board. The fourth author acknowledges partial support from the Australian Research Council.

References

[1] Back, A.D. and Tsoi, A.C., "A Time Series Modelling Methodology Using FIR and IIR Synapses", *Proc. Workshop on Neural Networks for Statistical and Economic Data*, Dublin, DOSES, Statistical Office of European Communities, F. Murtagh (Ed.), pp. 187-194, 1990.

[2] Back, A.D. and Tsoi,A.C., "FIR and IIR Synapses, a New Neural Network Architecture for Time Series Modelling". *Neural Computation*, vol 3. no. 3, pp. 375-385, 1991.

[3] Back, A.D. and Tsoi, A.C., "An Adaptive Lattice Architecture for Dynamic Multilayer Perceptrons", *Neural Computation*, Vol 4, No. 6, pp. 922-931, 1992.

[4] Bengio, Y. De Mori, R., Gori, M. "Learning the dynamic nature of s peech with backpropagation for sequences". *Pattern recognition Letters.* Vol. 13, pp 375 - 385, 1992.

[5] M. Casdagli, "Chaos and Deterministic versus Stochastic Non-linear Modelling", J. R. Statist. Soc. B, 1991, 54, No. 2, pp. 303-328.

[6] P. Frasconi, M. Gori and G. Soda, "Local Feedback Multilayered Networks", *Neural Computation*, vol. 4, no. 1, 1992.

[7] Gori, M., Bengio, Y., Mori, R.D. "BPS: a learning algorithm for capturing the dynamic nature of speech". *Intern. Joint Conf on Neural Networks*, Vol II, pp 417 - 423, 1989.

[8] N.A. Gershenfeld, and A.S. Weigend, "The Future of Time Series: Learning and Understanding", in Time Series Prediction: Forecasting the Future and Understanding the Past, Eds. A.S Weigend, and N.A. Gershenfeld, Addison-Wesley: Reading MA, 1993.

[9] U. Hübner, C.O. Weiss, N.B. Abraham, and D. Tang, "Lorenz-like Chaos in NH_3-FIR Lasers", in Time Series Prediction: Forecasting the Future and Understanding the Past, Eds. A.S Weigend, and N.A. Gershenfeld, Addison-Wesley: Reading MA, 1993.

[10] Lapedes, A. and Farber, R., "Nonlinear Signal Processing using Neural Networks: Prediction and System modelling", Tech Report LA-UR87-2662, Los Alamos National Laboratory, 1987.

[11] Leighton, R.R. and Conrath, B.C., "The Autoregressive Backpropagation Algorithm", Proc. Int. Joint Conf. Neural Networks, 1991.

[12] Mackey, M.C., and Glass, L., "Oscillation and Chaos in Physiological Control Systems", Science, vol. 197, pp. 287, 1977.

[13] Packard, N., Crutchfield, J., Farmer, D., and Shaw, R., "Geometry froma time series", Phys. Rev. Lett., vol 45., pp. 712-716.

[14] Poddar, P. Unnikrishnan, K.P. "Memory neuron networks: A Prolegomenon". General Motors Research Laboratories Report GMR-7493, October 21, 1991.

[15] Takens F., "Detecting Strange Attractors in Turbulence", Lecture Notes in Math., vol 898, Springer-Verlag, 1981.

[16] Stokbro K. and Umberger D.K., "Forecasting with Weighted Maps", in Nonlinear Modeling and Forecasting, SFI Studies in the Sciences of Complexity, Proc. Vol. XII, Eds. M. Cadagli, and S. Eubanks. Addison-Wesley, 1992.

[17] Wan, E.A., "Temporal backpropagation for FIR neural networks", *Proc. Int. Joint Conf. Neural Networks*, San Diego, June 1990, pp I 575-580.

[18] Wan, E.A., "Time Series Prediction by Using a Connectionist Network with Internal Delay Lines'", in A. Weigend and N. Gershenfeld, eds., *Time Series Prediction: Forecasting the Future and Understanding the Past.* Addison-Wesley, pages 195-218, 1994.

[19] Wan, E.A., "Finite Impulse Response Neural Networks with Applications in Time Series Prediction", *PhD Dissertation*, Stanford University, November, 1993.

[20] Waibel, A., Hanazawa, T., Hinton, G., Shikano, K., and Lang, K., "Phoneme recognition using time-delay neural networks", *IEEE Trans. Acoust., Speech, Signal Processing*, vol ASSP-37, March, 1989.