International Journal of Soft Computing and Engineering (IJSCE)
ISSN: 2231-2307, Volume-5 Issue-4, September 2015

Modelling and Control of Dynamical Systems Using Neural Network – A Review

Abubakar S. Umar, Muntaqa D. Alhassan, Kabiru Aminu, Salahuddeen G. Ahmad

Abstract: This paper presents a brief review on how artificial neural networks can be used in modelling and control of dynamical systems. The paper is broadly categorized into two; the first part is a short overview on artificial neural networks, particularly its generalization property, as applied to systems identification. The subsequent part contains a review onsome of the typical approaches used in the control of dynamical systems using neural networks which includes model predictive control, NARMA-L2 Control and model reference control. Finally, a comparative conclusion was made to distinguish the performances of the different control methods presented in this paper.

Index Terms: Neural Network Controllers; Generalization; Systems Modelling; Control Systems

I. INTRODUCTION

In a system, when the output quantity is controlled by varying the input quantity, the system is called a control system. Therefore control system is a device or set of devices, that manages, commands, directs or regulates the behaviour of other devices or systems. Generally, control systems are categorized into two; open loop control system and closed loop control system.

In open loop control, output is generated based on inputs only. This type of control is usually inaccurate and the changes in the output due to external disturbance are not corrected automatically. On the other hand, closed loop control system takes into consideration the current output and corrections are made automatically based on feedback. A closed loop system is also called feedback control system or automatic control system. The human body is a classic example of feedback systems.

The use of neural networks in control systems can be seen as a natural step in the evolution of control methodology to meet new challenges. Looking back, the evolution in the control area has been fueled by three major needs: the need to deal with increasingly complex systems, the need to accomplish increasingly demanding design requirements, and the need to attain these requirements with less precise advanced knowledge of the plant and its environment-that is, the need to control under increased uncertainty. Today, the need to control, in a better way, increasingly complex dynamical systems under significant uncertainty has led to a re-evaluation of the conventional control methods, and it has made the need for new methods quite apparent [1]. With the emergence of intelligent computing techniques, particularly artificial neural networks, control systems can be realized conveniently using the neural networks which provides many advantages, for instance robustness, stability and so on, over the conventional approach. In this paper some typical approaches are presented on the use of neural networks in control systems which include model predictive control [2], NARMA-L2 Control [3] and model reference control [4]. But first of all, a brief overview on generalization property of neural networks will immediately follow for better comprehension of the paper.

Revised Version Manuscript Received on August 12, 2015.

Abubakar S. Umar, M.Tech. Students, Department of Electrical Engineering, Jodhpur National University, Jodhpur, Rajasthan, India.
Muntaqa D. Alhassan, M.Tech. Students, Department of Electrical Engineering, Jodhpur National University, Jodhpur, Rajasthan, India.
Kabiru Aminu, M.Tech. Students, Department of Electrical Engineering, Jodhpur National University, Jodhpur, Rajasthan, India.
Salahuddeen G. Ahmad M.Tech. Students, Department of Electrical Engineering, Jodhpur National University, Jodhpur, Rajasthan, India.

II. OVERVIEW OF NEURAL NETWORKS AND GENERALIZATION

In machine learning and cognitive science, artificial neural networks, *ANN* are a family of statistical learning models inspired by biological neural networks (the central nervous systems of animals, in particular the brain) and are used to estimate or approximate functions that can depend on a large number of inputs and are generally unknown. Artificial neural networks are generally presented as systems of interconnected "neurons" which send messages to each other. The connections have numeric weights that can be tuned based on experience, making neural nets adaptive to inputs and capable of learning.

A multilayer perceptron network trained with the back-propagation algorithm may be viewed as a practical vehicle for performing a non-linear input-output mapping of a general nature. A network trained to generalize will perform as well in new situations as it does on the data on which it was trained [5].

In terms of neural networks, the simplest model is the one that contains the smallest number of free parameters (weights and biases), or equivalently, the smallest number of neurons and layers necessary to achieve an accurate approximation in a given problem [6].

The trick is to use enough neurons to capture the complexity of the underlying function without having the network over-fit the training data, in which case it will not generalize to new situations. We also need to have sufficient training data to adequately represent the underlying function.

To illustrate the problems we can have in network training, consider the following general example. Assume that the training data is generated by the following equation:

$$\boldsymbol{t_k} = \boldsymbol{f}(\boldsymbol{p_k}) + \boldsymbol{e_k} \qquad (1)$$

where; $\boldsymbol{p_k}$ is the system input, $\boldsymbol{f(\,.\,)}$ is the underlying function we wish to approximate, $\boldsymbol{e_k}$is measurement noise, and $\boldsymbol{t_k}$ is the system output (network target). Fig. 1shows an example of the underlying function (thick line), training data target values (circles), and total trained network response (thin line). The two graphs of Fig. 1 and that of Fig.2 represent different training strategies.

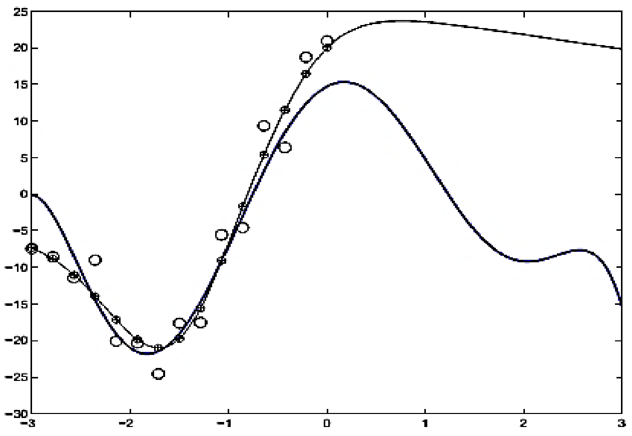
Fig.1. Good Fitting

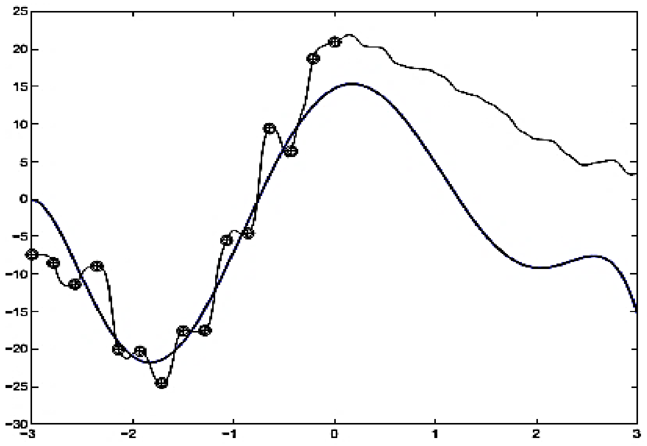
Fig.2. Over Fitting

As we can see from Fig. 2, there are two major problems. First, the network has over-fit on the training data. The network response is too complex, because the network has more than enough independent parameters, and they have not been constrained in any way. The second problem is that there is no training data for values greater than 0. Neural networks (and other nonlinear black box techniques) cannot be expected to extrapolate accurately. If the network receives an input that is outside of the range covered in the training data, then the network response will always be suspect [5][6].

While there is little we can do to improve the network performance outside the range of the training data, we can improve its ability to interpolate between data points. Improved generalization can be obtained through a variety of techniques. In one method, called early stopping [4][10], we place a portion of the training data into a validation data set. The performance of the network on the validation set is monitored during training. During the early stages of training the validation error will come down. When over-fitting begins, the validation error will begin to increase, and at this point the training is stopped.

Another technique to improve network generalization is called regularization. With this method the performance index is modified to include a term which penalizes network complexity [6].

III. TRAINING MULTILAYER NETWORKS

One of the key issues in designing such a multilayer network is determining the number of neurons to use, particularly in the hidden layer. If the number of neurons is too large, the network will over-fit the training data. This means that the error on the training data will be very small, but the network will fail to perform as well when presented with new data. A network that generalizes well will perform as well on new data as it does on the training data [13][11].

The complexity of a neural network is determined by the number of free parameters that it has (weights and biases), which in turn is determined by the number of neurons. If a network is too complex for a given data set, then it is likely to over-fit and to have poor generalization.

There are at least five different approaches that people have used to produce simple networks: growing, pruning, global searches, regularization, and early stopping. Growing methods start with no neurons in the network and then add neurons until the performance is adequate. Pruning methods start with large networks, which likely over-fit, and then remove neurons (or weights) one at a time until the performance degrades significantly. Global searches, such as genetic algorithms, search the space of all possible network architectures to locate the simplest model that explains the data [5]-[8],[12].

The final two approaches, regularization and early stopping, keep the network small by constraining the magnitude of the network weights, rather than by constraining the number of network weights.

IV. NEURAL NETWORK APPROACH IN SYSTEMS MODELLING AND CONTROL

When using neural networks for control, there are typically two steps involved: system modelling and control design. In the system modelling or identification stage, we normally develop a neural network model of the plant that we want to control. In the control design stage, we use the neural network plant model to train the controller. In each of the three control architectures described in this paper, the system identification stage is identical. The control design stage, however, is different for each of the architectures. The next three subsections of this paper discuss model predictive control, NARMA-L2 control and model reference control [9]-[11]. The first stage of neural control, as mentioned earlier, is to train a neural network to represent the forward dynamics of the plant. The prediction error between the plant output and the neural network output is used as the neural network training signal. The process is represented by Fig. 3.

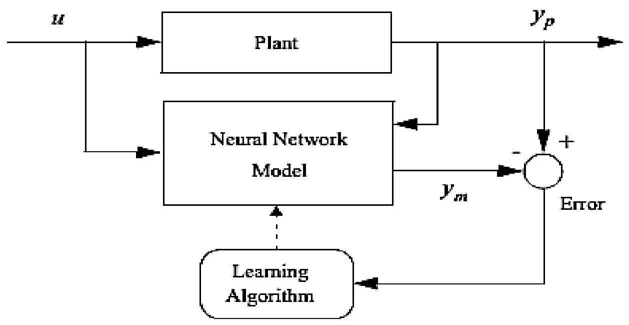

Fig.3. Plant Identification

One standard model that has been used for nonlinear identification is the Nonlinear Autoregressive-Moving Average, *NARMA model* [3]:

$$\boldsymbol{y(k+d) = h[y(k), y(k-1), \dots, y(k-n+1), u(k), u(k-1), \dots, u(k-m+1)]} \quad (2)$$

where $\boldsymbol{u(k)}$ is the system input, $\boldsymbol{y(k)}$ is the system output and $\boldsymbol{d}$ is the system delay. For the identification phase, we train a neural network to approximate the nonlinear function. The structure of the neural network plant model is given in Fig. 4, where the blocks labeled TDL are tapped delay lines that store previous values of the input signal. The equation for the plant model is given by:

$$\boldsymbol{y_m(k+1) = \tilde{h}[y_p(k), y_p(k-1), \dots, y_p(k-n+1), u(k), u(k-1), \dots, u(k-m+1); x]} \quad (3)$$

where,$\boldsymbol{\tilde{h}}[.,\boldsymbol{x}]$is the function implemented by the neural network, and $\boldsymbol{x}$ is the vector containing all network weights and biases.

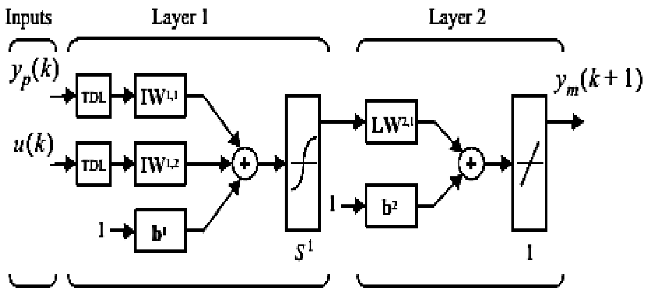

Fig.4. Plant Model

A. Model Predictive Control

The model predictive control method is based on the receding horizon technique [2]. The neural network model predicts the plant response over a specified future time horizon. The predictions are used by a numerical optimization program to determine the control signal that minimizes the following performance criterion over the specified horizon [16].

$$\boldsymbol{J = \textstyle\sum_{j-n_1}^{n_2}\left(y_r(k+j) - y_m(k+j)\right)^2 + \rho\sum_{j=1}^{n_u}(\tilde{u}(k+j-1) - uk+j-2)2} \quad (4)$$

where,$\boldsymbol{n_1}$, $\boldsymbol{n_2}$and $\boldsymbol{n_u}$define the horizons over which the tracking error and the control increments are evaluated. The $\boldsymbol{\tilde{u}}$ variable is the tentative control signal,$\boldsymbol{y_r}$is the desired response and $\boldsymbol{y_m}$is the network model response. The $\boldsymbol{\rho}$ value determines the contribution that the sum of the squares of the control increments has on the performance index.

The following block diagram illustrates the model predictive control process. The controller consists of the neural network plant model and the optimization block. The optimization block determines the values of *u'* that minimize *J*, and then the optimal *u* is input to the plant.

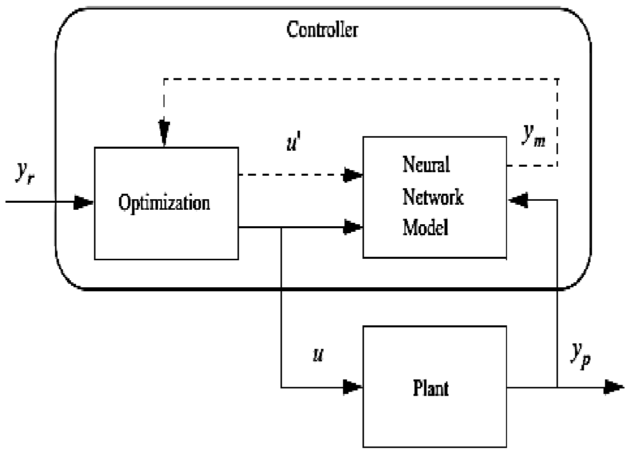

Fig.5. NN Predictive Control

B. NARMA-L2 Control

The advantage of the NARMA-L2 form is that you can solve for the control input that causes the system output to follow a reference signal:$\boldsymbol{y(k+d) = y_r(k+d)}$. The resulting controller would have the form;

$$u(k) = \frac{y_r(k+d) - f[y(k), y(k-1), \dots, y(k-n+1), u(k-1), \dots, u(k-m+1)]}{g[y(k), y(k-1), \dots, y(k-n+1), u(k-1), \dots, u(k-m+1)]} \quad (5)$$

which is realizable for,$\boldsymbol{d \geq 1}$. Fig. 6 is a block diagram of the NARMA-L2 controller [2][17].

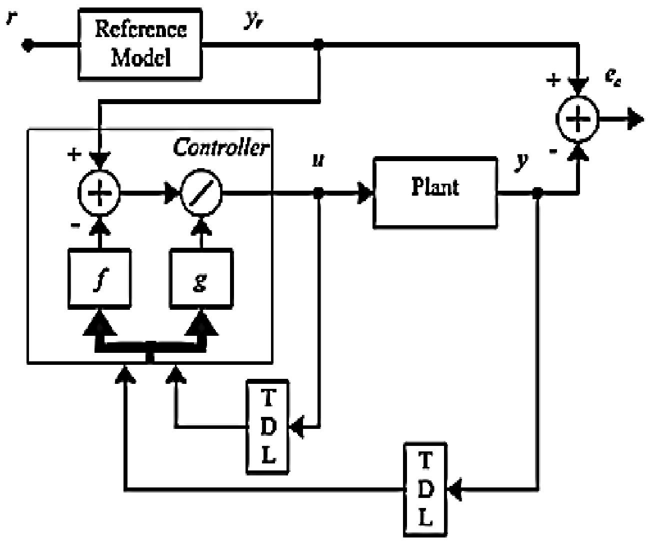

Fig.6. NARMA-L2 Controller

C. Model Reference Control

The third neural control architecture we will discuss in this paper is model reference control [4]. This architecture uses two neural networks: a controller network and a plant model network, as illustrated in Fig. 7. The plant model is identified first, and then the controller is trained so that the plant output follows the reference model output.

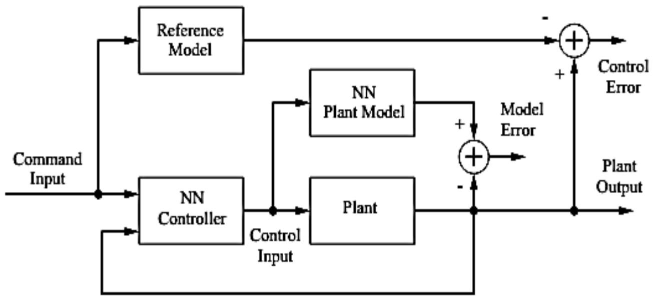

Fig.7. Model Reference Control

The online computation of the model reference controller, as with NARMA-L2, is minimal. However, unlike NARMA-L2, the model reference architecture requires that a separate neural network controller be trained, in addition to the neural network plant model. The controller training is computationally expensive, since it requires the use of dynamic backpropagation [12][4]. On the positive side, model reference control applies to a larger class of plant than does NARMA-L2 control, which requires that the plant be approximated by a companion form model.

There are three sets of controller inputs: delayed reference inputs, delayed controller outputs (plant inputs), and delayed plant outputs. For each of these inputs, we select the number of delayed values to use. Typically, the number of delays increases with the order of the plant. There are two sets of inputs to the neural network plant model: delayed controller outputs and delayed plant outputs [14][15].

The plant identification process for model reference control is the same as that for the model predictive control, and uses the same NARMA model given by (2). The training of the neural network controller, however, is more complex.

V. CONCLUSION

Model Predictive Control uses a neural network plant model to predict future plant behaviour. An optimization algorithm determines the control input that optimizes plant performance over a finite time horizon. The plant training requires only a batch algorithm for feedforward networks and is reasonably fast. The controller requires an online optimization algorithm, which requires more computation than the other two controllers. *NARMA-L2 Control* is a variation of the feedback linearization controller. An approximate companion form plant model is used. The next control input is computed to force the plant output to follow a reference signal. The neural network plant model is trained with static back propagation. The controller is a rearrangement of the plant model, and requires minimal online computation. In *Model Reference Control*, neural network plant model is first developed. The plant model is then used to train a neural network controller to force the plant output to follow the output of a reference model. This control architecture requires the use of dynamic back propagation for training the controller. This generally takes more time than training static networks with the standard back propagation algorithm. However, this approach applies to more general class of plant than does the NARMA-L2 control architecture. The controller requires minimal online computation [11].

REFERENCES

1. P.J. Antsaklis,'Neural Networks in Control Systems,' The IEEE Control Systems Magazine, Vol.10, No.3, pp.3-87, April 1990.
2. Soloway, D. and P.J. Haley, 'Neural Generalized Predictive Control,' Proceedings of the 1996 IEEE International Symposium on Intelligent Control,277-281 (1996).
3. Narendra, K.S. and S. Mukhopadhyay, 'Adaptive Control Using Neural Networks and Approximate Models,' IEEE Transactions on Neural Networks, vol. 8, 475-485 (1997).
4. Narendra, K.S. and K. Parthasarathy, 'Identification and Control of Dynamical Systems Using Neural Networks,' IEEE Transactions on Neural Networks, vol. 1, 4-27 (1990).
5. Haykin, S., Neural Networks: A Comprehensive Foundation, 2nd Ed., Prentice-Hall, Englewood Cliffs, NJ, 1999.
6. Zurada, J.M., Introduction to Artificial Neural Networks, 2nd Ed., West Publishing Company, 1992.
7. Hagan, M. T. and H.B. Demuth, 'Neural Networks for Control,' Proceedings of the 1999 American Control Conference, San Diego, CA, 1642-1656 (1999).
8. Hunt, K.J., D. Sbarbaro, R. Zbikowski and P.J. Gawthrop, 'Neural Networks for Control System - A Survey,' Automatica, vol. 28, 1083-1112 (1992).
9. K ˚ urkov´ a, "Approximation of functions by perceptron networks with bounded number of hidden units", Neural Networks, vol. 8, pp. 745–750, 1995.
10. Hagan, M.T., Demuth, H.B., Beale, M.H., 'Neural Network Design,' 2nd Ed., Campus Publication Service, University of Colorado Bookstore, 2002.
11. Hagan, M.T., Demuth, H.B., O. De Jesus, 'An Introduction to the Use of Neural Networks in Control Systems,' International Journal of Robust and Nonlinear Control, John Wiley & Sons, 2002
12. Hagan, M. T., O. De Jesus, and R. Schultz, 'Training Recurrent Networks for Filtering and Control,' Chapter 12 in Recurrent Neural Networks: Design and Applications, L. Medsker and L.C. Jain, Eds., CRC Press, 311-340 (1999).
13. Pham, D. T. and X. Liu, 'Neural Networks for Identification, Prediction, and Control,' Springer-Verlag, New York, 1995.
14. Omatu, S., M. B. Khalid, R. Yusof, 'Neuro-Control and its Applications,' Springer-Verlag, London, 1996.
15. Omidvar, O. and D. Elliott,Neural Systems for Control, Academic Press, New York, 1997.
16. Norgard, M., O. Ravn, N.K. Poulsen, and L.K. Hansen,Neural Networks for Modelling and Control of Dynamic Systems, Springer-Verlag, London, 2000.
17. Liu, G.P., 'Nonlinear Identification and Control,' Springer-Verlag, London, 2001.