Recurrent Neural Networks in Linear Systems Controlling

P. Patic, Ryad Zemouri, L. Duţă

Recurrent Neural Networks in Linear Systems Controlling

P. C. Patic[1], R. M. Zemouri R.[2], L. Duta[1]

[1] Valahia University of Târgovişte,
18-24, Unirii Ave., Târgovişte, Romania
patic@valahia.ro, duta@valahia.ro

[2] Laboratoire d'Automatique du CNAM,
21, Rue Pinel, 75013 Paris, France
ryad.zemouri@cnam.fr

Abstract. This paper presents an application of an ANN (Artificial Neural Network) of a RNRF type (Recurrent Network with Radial basis Function) in controlling a linear system. The performance of ANN-based control solution is compared with a classic controller and the results show that ANN behaves better than the classic controller. MATLAB simulation performed show that the coupling between the ANN and a proportional controller gives the best performance.

1. Introduction

The linear systems control is today an important research and development area in control engineering. In the real world, however, many processes are characterized by a nonlinear dynamic behavior which makes impossible to use conventional tools for automatic control. The same applies to systems for which mathematical models are incompletely specified or of a poor quality. There is currently no systematic theory to be applied to control such processes. To solve this problem, one solution is to use a learning phase to identify the process model or controller. The term "learning" is about changing the structure and/or system settings in order to improve its future performance, based on previous experimental observations [1, 4]. Some adaptive controlling methods have been developed, to enable the evolution of controller depending on the task [3, 9]. The structure of the controller being already chosen, those methods allow fixing a number of parameters of this one. If the general principle used by these algorithms is similar to learning done by ANN, adaptation is done by a simple setting of a small number of coefficients of the control loop without storage capacity [5]. Systems with learning characteristics such as ANN (Artificial Neural Networks) can be successfully utilized in control problems such as the decision support or situation recognition [4, 6, 12, 13]. In this case we speak about learning in command rather than adaptive control.

This article consists of three parts. Section I reviews a dynamic variation of ANN that one of the authors of this paper proposed in a previous work [21]. Section II presents the basic principles and some methods of neurons control technique. Finally, Section III presents the tests and the obtained experimental results.

2. Recurrent Neural Networks

The used neural network is a variant of dynamic networks radial basis functions of dynamic RNRF: Recurrent Neural Networks with Radial Basis Function [19] (Figure 1).

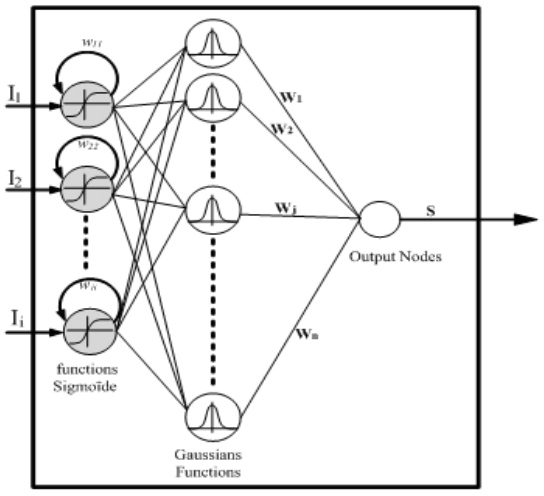

Figure 1. RNRF network [11]

The RNRF considers the time as an internal parameter of the network [2, 5, 7]. This dynamical aspect is obtained by a recurrence of connections between neurons of the input layer. These self connections provide the input neurons a capacity for taking into account of past input data. The neural network is equipped with two types of memories: a dynamic memory, for taking into account the dynamic data input and a static memory to store prototypes. The output layer represents the layer of Gaussian weighting [13].

Each neuron from the input layer performs an addition at time t of its input $I_i(t)$ and at the output of the preceding time $x_i(t-1)$ weighted by the coefficient of self connection w_{ii} [17]. The neuron outputs the result of the activation function:

$$\begin{aligned} a_i(t) &= w_{ii}x_i(t-1) + I_i(t) \\ x_i(t) &= f\left(a_i(t)\right) \end{aligned} \quad (1)$$

where $a_i(t)$ and $x_i(t)$ represent the activation of neuron i and its output at time t respectively, w_{ii} is the weight of self-connection of neuron i, f is the activation function with the expression of the sigmoid:

$$f(x) = \frac{1-\exp(-kx)}{1+\exp(-kx)} \quad (2)$$

The layer of Gaussian neurons [21] provides a representation of the Euclidean space. The answer $\phi_j(\mathbf{x})$ is given according to the kernel $\boldsymbol{\mu}_j$ and the standard deviation σ_j:

$$\phi_j(\mathbf{x}) = \exp\left(-\frac{\left\|\mathbf{x}-\boldsymbol{\mu}_j\right\|^2}{2\sigma_j^2}\right) \quad (3)$$

the network output is given by the following linear expression:

$$S = \boldsymbol{\phi}(\mathbf{x}).\mathbf{W} \quad (4)$$

The learning process of RNFR involves two steps. The first is to define the number and the parameters of Gaussian functions. The method of k-means coupled with the method Min-Max is used [16], [20]. The second step is to calculate the weights of the connections of the output layer W by matrix inversion.

3. Neural Control

The data generally available for the realization of a learning control system are the couples combining the output (or state) $y(t)$ with the desired process output (or state) $y^*(t)$ [18]. We can measure the error ε_y (the difference between the desired output value and the measured value) obtained at the output of the system after applying the command.

Contrary, the error ε_u of the controller output which is the difference between the ideal command, which is not known, and the command $\hat{u}(t)$ recommended by the controller, can not be directly obtained. Yet, this mistake is necessary to achieve the adjustment of controller by a method of supervised classical learning (the back-propagation gradient).

3.1 Replication of an existing controller

The first method used to realize a neural control system was to replicate the operation of an existing controller (Figure 2).

Although, this approach seems at first sight unattractive, since it requires the existence of another controller, it can be useful if the controller is too complex or too slow to be used in real time, or if data is not always available. One can take into account the case when the control system is reproduced by a human operator.

This method teaches the network to reproduce the command $\hat{u}(t)$ recommended by the first controller from the desired output $y^*(t)$ or earlier output. One can note that this approach requires the appliance of all modes of operation of the controlled system, during the learning phase. Therefore, it is necessary to have a good prior knowledge of the operating conditions of the controller.

3.2 Improving a system of linear control

This approach consists of using together a classical linear controller and a neural controller (see Figure 2 and 3). The main idea is to achieve an amount of orders from both

controllers, gradually increasing the importance given to the command $\hat{u}(t)$ recommended by the ANN, as an extent of the learning network [10].

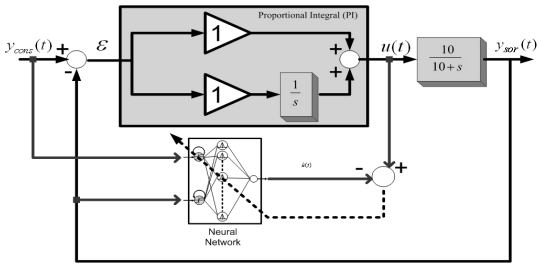

Figure 2. Schematic learning of the neural network [11,17].

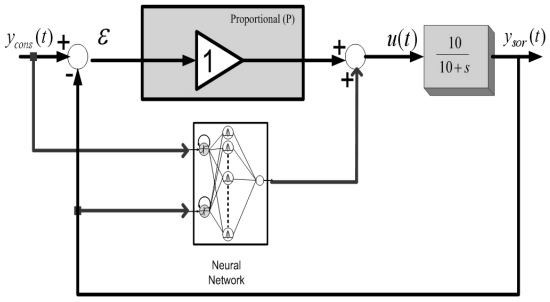

Figure 3. Improved performance of the neural control by a proportional gain [10, 17]

3.3 Direct use of the error output of the process

A simple approach is to try directly to use the measured error at the output of the process as to adapt the controller (Figure 4). Several strategies are possible. The first is to use this error as the error at the output of the controller. This approach can work only if these errors are highly correlated, which is rarely the case. The most widely used method is to consider the process as an additional layer of the network through which the error feeds back [15], [22].

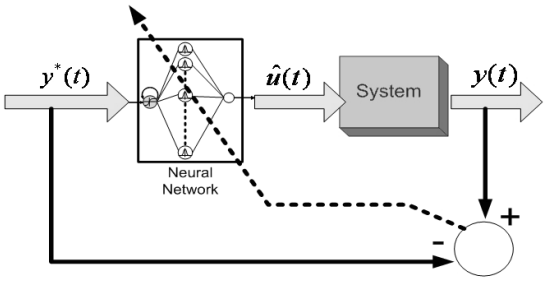

Figure 4. Learning neural network based on error Release System

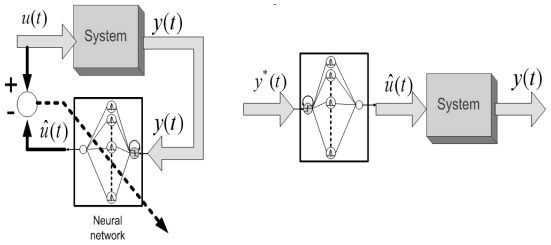

Figure 5. Learning the inverse model

3.4 Identification of the inverse model

This approach requires two separate phases for learning and for using the network (Fig. 5). During learning, the network and the process are placed in parallel. A sample command $u(t)$ is given to the process. It will use the output $y(t)$ of the network as an input of the network to find the output command $\hat{u}(t)$. The network learns an inverse model of the process, a function giving the command $\hat{u}(t)$ from the current output of the process $y(t)$, eventually the last output $y(t-1)$. After this learning phase the network is theoretically capable of providing the command u(t) required to obtain an output $y^*(t)$ which is given as input. This is placed directly in series with the controlled system. Figure 5 summarizes these two phases. During the learning phase, the process passes through all its possible states, or at least, all the states that will be used during the checkout. While in many cases, a simple sampling of the control space is enough in this phase it may be necessary to use another control system to guide the process.

4. Application

4.1 Description

We applied the RNRF network on control of a first order command system $(10/(10+s))$ whose behavior is in open loop. The simulations were performed in MATLAB.

4.2 Learning phase

We have learned the behavior of an ANN from a Proportional Integral (PI) controller whose configuration has not been optimized (Figure 2) in order to test the behavior of ANN on an incomplete knowledge of a system. For the learning phase of the ANN, control sequence between [0, 1] is defined.

Figure 6 shows the signals Y_{cons} and Y_{sor} which respectively represent the input and the output in closed-loop system served by the PI controller. These two signals are the inputs for the ANN. Figure 7 shows the output of PI controller (U_{com}) which is also the desired output of the ANN.

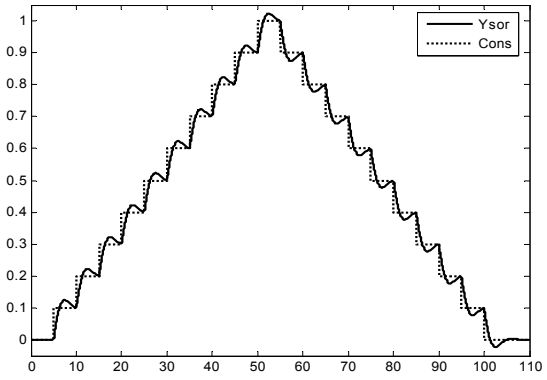

Figure 6. Sequence of the reference signal with the output of the closed loop system used for learning.

4.3 Testing procedure of the ANN

After the learning phase, the neural network was tested on the same system with several different types of signals. We compared the performance of three types of controllers: the Proportional Integrator (PI), the neural controller (Figure 8) and the neural controller coupled to a proportional one (Figure 3).

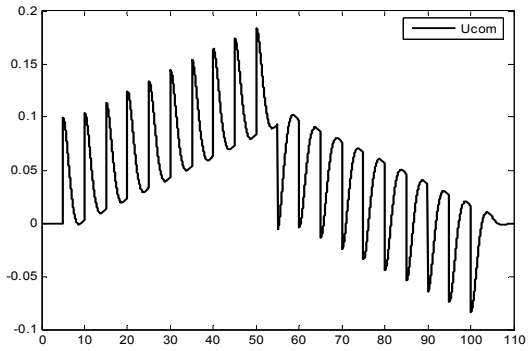

Figure 7. Control signal corresponding to the command sequence for learning

4.4 Results

Test with the same data as in the learning phase:

The first test of the ANN is to see its behavior in control loop (Figure 8) with the same control sequence used in the learning process. Figure 9 shows the output of the system controlled by the neural network. One notes that, apart from the peak of the first step, the neural controller behaves better than the PI controller (Figure 6). The overtaking is nonexistent.

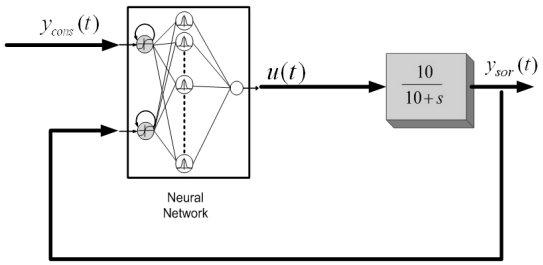

Figure 8. Neural control scheme

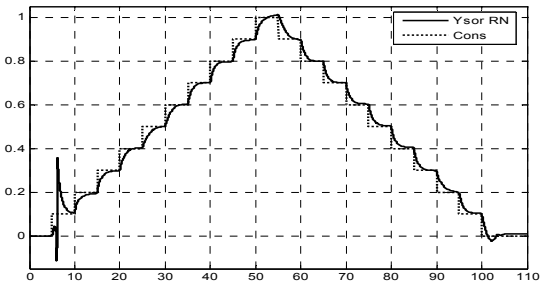

Figure 9. Testing neurons on the same learning sequence

Test with a step standard signal:

The second test is meant to evaluate the behavior of the neural controller to some step signals different from those used in the learning phase (the variation of the signal amplitude during the learning phase being equal to 0.1).

Figure 10 and Figure 11 show the comparison of the three controllers' behaviors: PI, Neural Network (ANN) and neural network coupled to the proportional controller. Both tests show that the neural controller gives better performances than the PI controller. These performances are even better thanks to the coupling of neural controller with the proportional controller.

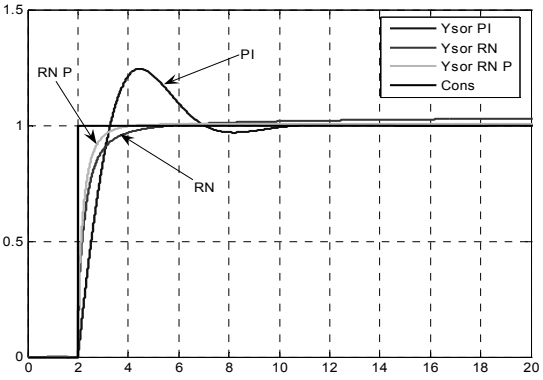

Figure 10. Testing the neural network on a step of 1

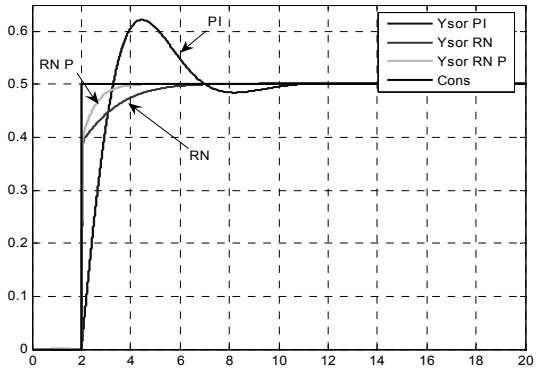

Figure 11. Testing the neural network on a step of 0.5

Test on a sinusoidal signal

The third test of three controllers for sinusoidal signals. Figure 12 and Figure 13 show these results on two sinusoids with different frequencies. As in the previous case, the neural controller gives better performance than the PI controller.

The coupling of the neural network with the proportional controller gives a better approximation of the input.

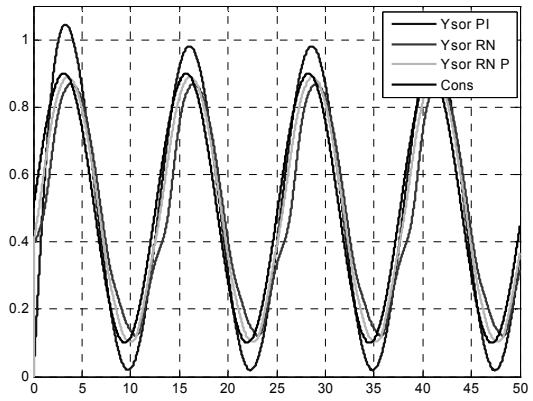

Figure 12. Testing the neural network on a sinusoidal signal of 0.08 Hertz.

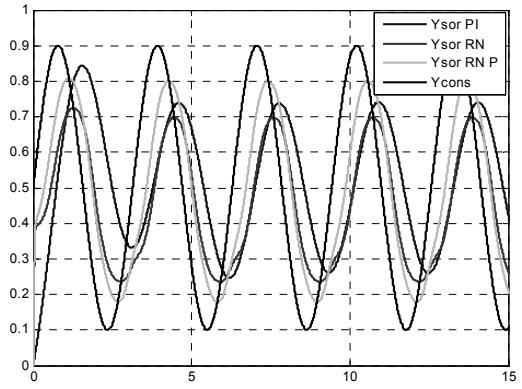

Figure 13. Testing the neural network on a sinusoidal signal of 0.3 Hertz.

5. Conclusions

An application of the ANN in control of linear systems is presented in this paper. A type of dynamic neural network with radial basis functions (RNRF) was tested on an illustrative example. The results show that the neural controller behaves better than the original controller which was used in the learning phase. The future work will be developed on testing the proposed neural network in other more complex applications, such as nonlinear and unstable systems (as the inverted pendulum). One can note that the results may be successfully used in different robotic applications [14] possibly in conjunction with frequency-based methods [8].

REFERENCES

1. BAKER, W. L., J. A. FARRELL, **An Introduction to Connectionist Learning Control Systems**, In D. A. White and D. A. Sofge, editors, Handbook of Intelligent Control, Van Nostrand Reinhold Ed., 1992, pp. 35-64.

2. CHAPPELIER, J. C., **RST: une architecture connexionniste pour la prise en compte de relations spatiales et temporelles**, Thèse de doctorat, Ecole Nat. Sup. des Télécomm., 1996.

3. COSTAFTIS, M., **Comportement et contrôle des systèmes complexes: introduction aux méthodes algébriques, qualitatives et fonctionnelles**, Diderot éd., Arts et Sciences, 1997.

4. ELFELLY, N., J.-Y. DIEULOT, P. BORNE, **A Neural Approach of Multi-model Representation of Complex Systems**, International. Journal of Computers Communication & Control-IJCCC vol. 3, no. 2, 2008, pp. 149-160.

5. ELMAN, J. L., **Finding Structure in Time**, Cognitive Science, vol. 14, 1990, pp. 179-211.

6. FILIP, F. G., K. LEIVIISKA, **Large-scale complex systems**, Springer

Handbook of Automation (S. Y. Nof , Ed.). Springer, Berlin, 2009, pp. 619-638.

7. GAUTHIER, E., **Utilisation des Réseaux de Neurones Artificiels pour la Commande d'un Véhicule Autonome**, Thèse de doctorat, Institut National Polytechnique de Grenoble, Janvier 1999.

8. IVANESCU, M., M. C. FLORESCU, **The Control of Hyper-redundant Manipulators by Frequency Criteria**, Studies in Informatics and Control vol.18, no3, 2009, pp. 279-288.

9. JORDAN, M. I., D. E. RUMELHART, **Forward Models: Supervised Learning with a Distal Teacher**, Cognitive Science, vol. 16, 1992, pp. 307-354.

10. KAWATO, M., **Schemes and Models for Control of Arm Trajectory**, In W. T. Miller, R. S. Sutton, and P. J. Werbos editors, Neural Networks for Control, MIT Press, 1990, pp. 197-228.

11. NARENDRA, K., A. M. ANNASWAMY, **Stable Adaptive Systems**, Prentice-Hall, Englewood, NJ, 1989.

12. NARENDRA, K. S., K. PARTHASARATHY, **Identification and Control of Dynamical Systems Using Neural Networks**, IEEE Transactions on Neural Networks, vol 1(1), 1990, pp. 4-5.

13. PATIC, P. C., **Neural Approaching of the Materials Processing Establishing Problem**, Scientific Bulletin of the Electrical Engineering Faculty, 2006, pp. 57-59.

14. POMERLEAU, D. A., **Neural Network Perception for Mobile Robot Guidance**, Kluwer Academic Press, 1993.

15. PSALTIS, D., A. SIDERIS, A. YAMAMURA, **A Multi-layer Neural Network Controller**, IEEE Control Systems Magazine, no. 8, 1988, pp. 17-21.

16. SIMPSON, P. K., **Fuzzy Min-max Neural Networks – Part II: Clustering**, IEEE Transaction on Fuzzy Systems, Vol.1, 1993, pp. 32-45.

17. STERGIOU, C., D. SIGANOS, **Neural Networks**, (http://www.docstoc.com/docs/15050/NEURAL-NETWORKS-by-Christos-Stergiou-and-Dimitrios-Siganos) on-line 2007

18. WIDROW, B., F. W. SMITH, **Pattern-Recognizing Control Systems**, Proceeding of Computer and Information Sciences, 1964.

19. ZEMOURI, R., D. RACOCEANU, N. ZERHOUNI, **Recurrent Radial Basis Function Network for Time-Series Prediction**, Engineering Applications of AI, The International Journal of Intelligent Real-Time Automation, journal IFAC, Elsevier Science, vol. 16, Issue 5-6, 2003, pp. 453-463.

20. ZEMOURI, R., D. RACOCEANU, N. ZERHOUNI, **Réseaux de neurones à fonctions de base radiales: application au pronostic**, Revue des sciences et technologies de l'information - RSTI, série Revue d'intelligence artificielle-RIA, vol. 16, nr. 13/2002, ISBN: 2-7462-0568-8, Ed. Hermès – Lavoisier, Paris, 2002, pp. 307-338.

21. ZEMOURI, R., **Contribution à la surveillance des systèmes de production à l'aide des réseaux de neurones dynamiques: Application à la e-maintenance**, Thèse de doctorat, Université de Franche-Comté, Nov. 2003

22. ZHANG, Y., P. SEN, G. E. HEARN, **An On-line Trained Adaptive Neural Controller**, IEEE Control Systems Magazine, vol. 15(5), 1995, pp. 67-75.