Proceedings of the
American Control Conference
Baltimore, Maryland • June 1994

TM7 - 2:10

INTEGRATION OF CONTROLS AND DIAGNOSTICS USING NEURAL NETWORKS

Ioannis K. Konstantopoulos and Panos J. Antsaklis
Department of Electrical Engineering
University of Notre Dame
Notre Dame, IN 46556, U.S.A.
e-mail: antsakli@saturn.ece.nd.edu

Abstract

Controllers capable of performing failure diagnosis have additional diagnostic outputs to detect and isolate sensor and actuator faults. A linear such controller is usually called a four-parameter controller. In this paper, a neural network model of a controller with diagnostic capabilities (CDC) is presented for the first time. This nonlinear neural controller is trained to operate as a traditional controller, while at the same time it provides reproduction of the failure occurring either at the actuator or the sensor. The cases of actuator and sensor failures are studied independently. Several simulation results concerning a nonlinear plant are provided.

1. Introduction

The four-parameter controller is a linear controller used for both control and diagnostics. It has an additional output that can be viewed as a diagnostic output, which is monitored to detect and isolate sensor and actuator faults. Therefore, in addition to attaining conventional control goals such as reference input tracking, the controller must also provide information about possible failures that occur. A neural network model of the controller with diagnostic capabilities (CDC) is implemented here for the first time. This paper, which is a very brief version of [8], is based on work that was reported in [6]; some of this work was presented at [7]. Note that this neural network implementation is applicable to both linear and nonlinear systems, thus extending the applicability of the concept of the four-parameter controller. Here, the cases of actuator and sensor failures are considered separately, that is for each the assumption is made that only one kind of failure could occur.

In section 2, a brief outline of the original linear CDC, the four-parameter controller, is presented and neural networks are briefly discussed. In sections 3 and 4 a feedforward multilayer neural network is used to implement the CDC controller for actuator and sensor failures respectively. Methods to emphasize either the control or the diagnostic aspects of the CDC controller are also presented. Finally in section 5, concluding remarks are discussed.

2. Preliminaries

The four-parameter controller, which is attributed to C. Nett, [11], is a generalization of the familiar two-parameter linear controller. In the four-parameter controller literature, the control configuration of Fig. 1 has been studied extensively, where r is the command or reference input to the controller, a is the diagnostic output, that is the output of the controller that is designed to reproduce the failures, y_c is the ideal actuator input, u_c is the manipulated controller input, n_a and n_s are exogenous inputs which account for unmodelled signals that inevitably appear at the actuator and sensor respectively, u is the actual actuator output, y is the ideal sensor input, z is the unutilized plant output, and w is the disturbance.

The exogenous inputs n_a, n_s represent the deviations from the ideal sensor and actuator outputs and are defined as $n_a = f_a+\eta'$ and $n_s = f_s+\eta$ respectively, where f_a, f_s represent the failures in the actuator and sensor and η', η the respective noises. We can now specify the objective of the introduction of the additional controller output a, which is <u>to be able to identify and reproduce F at a</u>, where we define $F = (\, f_a^T \quad f_s^T \,)^T$. This is the diagnostic objective. The control objective is to achieve set point tracking, that is follow or track the reference input r at the plant output z and at the same time reject the unmeasured disturbance w at the plant output z. These requirements have to be satisfied in the context of plant modelling errors. It has been shown that the above requirements lead to certain conflicts; details can be found in [4]. Several applications of the four parameter controller have been reported; see [5], [9], [12], [14].

An *artificial neural network* (ANN) is made up of many parallel elements, called *neurons*, which can be described by a standard nonlinear algebraic or differential equation, and are interconnected via adaptive multiplicative parameters, called *weights*. The type of neural network used here is the *feedforward multilayer neural network*. Such a network is made up of any number of layers with any number of neurons in each of these layers. A popular learning heuristic for multilayer feedforward neural network is the *back propagation algorithm* (BP); details can be found in any book on neural networks, as in [2]. Neural

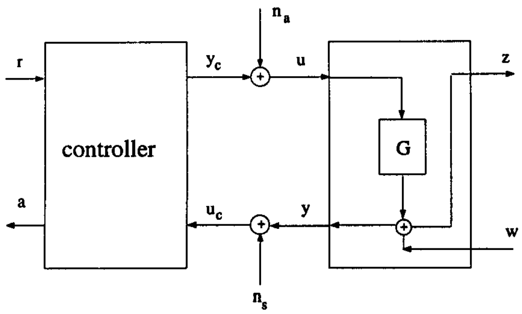

Figure 1: Four-parameter controller: general controller and restricted plant

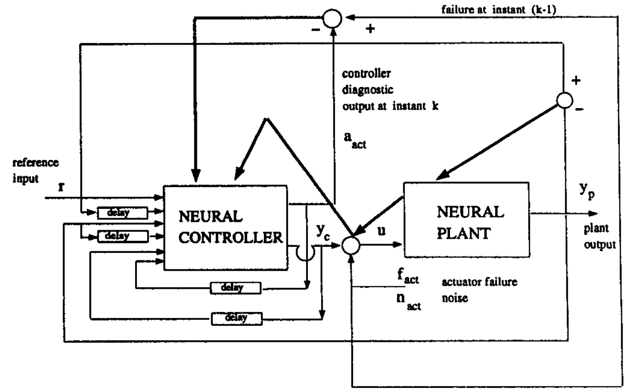

Figure 2: Training of neural CDC controller for actuator failures

networks have been used to detect and identify failures, especially in chemical processes; see [6], [8] and references therein. Neural networks have extensively been used in control systems; see [1], [6], [13].

3. Actuator Failures

3.1. Introduction

In this section we assume that we have only actuator failures. Our task is to train and test a neural network model of the CDC controller, that will have the following task:

"*to achieve set point tracking, that is to follow the reference input r at the plant output y_p, and also to isolate the actuator faults and reproduce them at the diagnostic output a_{act}.*"

It is common in the failure diagnosis research work to establish an upper limit-bound, so that an alarm signal is activated, when this limit is exceeded. Although this limit can be defined theoretically in many possible ways, depending on the theory that one uses, H_∞ for instance, it is basically quite arbitrary, since each theoretical foundation relies on specific assumptions, which are usually arbitrary themselves. In practice, this safety limit is given by the designer and its value designates the magnitude of failure that can be tolerated in each specific case. Therefore, it appears more reasonable and realistic to try to reproduce the failure as accurately as possible, and then allow ourselves to make the decision whether the alarm should be activated or not.

The configuration that we are going to study is shown in Fig. 2. As we see, a neural network model for the CDC controller has been adopted. This controller has two general inputs, the reference signal r and the output of the plant y_p, and two outputs, the diagnostic output a_{act}, and the y_c output, which can be considered as the ideal actuator input. The diagnostic output a_{act} will be trained to give us the reproduction of the actuator failure, if a failure does occur at the actuator. We also have some more additional inputs to the neural controller. These are the delayed reference inputs, delayed plant outputs, as well as delays of both the controller outputs.

In order to train the neural model of the controller, we need to know the error at the output of the controller. As far as the error at the diagnostic output a_{act} is concerned, we simply compute the difference $(a_{act}^{ideal} - a_{act})$. On the other hand, we have no information about what the ideal output y_c^{ideal} should be. The error at the controller output y_c is easily computed, when a neural model of the plant is available, as it is the case here. The purpose of this neural model is to provide a path for the plant output error to reach the controller, so that the error at the controller output y_c can be computed.

The BP algorithm is used to train the neural controller, as well as to back-propagate the error through the plant, that is from the output of the plant identification model to its input. For any specific time instant, the current and delayed reference inputs, the current and delayed neural plant outputs, as well as previous neural controller outputs, both of a_{act} and y_c, are applied to the input of the controller and propagate to its outputs. The diagnostic output, denoted by $a_{act}(k)$, reproduces the failure. The second output, denoted by $y_c(k)$, comes to the actuator and is added to the actuator failure, denoted by $f_{act}(k)$. The resulting signal, $u(k) = y_c(k) + f_{act}(k)$, is the input to the neural plant.

This input, together with the previous plant outputs, propagate through the neural plant to its output. It should be noted that no actuator noise is considered during the training. Note that the desired output at the plant is the current reference input, denoted by $r(k)$, considered at the beginning of this description, and that the desired diagnostic output is the failure at the previous time instant, that is $f_{act}(k-1)$. The error at the neural plant output is obviously $r(k) - y_p(k)$. This error is back-propagated to the input of the neural plant. The error that corresponds to the input $u(k)$ is given by $error_{u(k)} = u^{ideal}(k) - u(k)$, from which we readily have that $u^{ideal}(k) = error_{u(k)} + u(k)$.

Now the ideal output $y_c^{ideal}(k)$ of the controller, that is the one that would satisfy the control requirement of the controller, is given by $y_c^{ideal}(k) = u^{ideal}(k) - f_{act}(k)$, which implies that the error that corresponds to the controller output $y_c(k)$ is given by $error_{y_c(k)} = y_c^{ideal}(k) - y_c(k)$. This error, together with the diagnostic output error mentioned above, are used by the BP algorithm to adjust the weights of the neural controller. The nonlinear plant model that has been used, [10], is described by the following equation:

$$y_p(k+1) = \frac{y_p(k)y_p(k-1)(y_p(k)+2.5)}{1+y_p^2(k)+y_p^2(k-1)} + u(k) \qquad (1)$$

For the above plant model an identification neural model has already been found in [3]. After several experiments, a 2 hidden layer structure was decided for the neural model of the CDC controller; 20 and 5 neurons were chosen for the first and second hidden layer respectively. For the inputs of the neural controller, 3 delays were considered for the reference input r, the plant output y_p, as well as the previous outputs of the neural controller. Regarding the training sets that were used during the training of the controller, a random signal uniformly distributed in the range [-2, 2] was considered for the reference input and samples of a sinusoidal function were considered for the actuator failures. The sinusoidal signal $r = 2\ sin(2\pi k/25)$ was tested as the reference input and a step signal of 0.8 was tested as the actuator failure. Note that actuator noise uniformly distributed in the range [-0.05, 0.05], was also applied as an input to the actuator for all the tests. This noise simulates the plant modelling errors.

To evaluate the performance, the difference vector ($R - Yplant$) was considered, where R is the vector with the sampling values of the reference signal and $Yplant$ is the vector with the respective outputs of the neural plant. Then, the absolute values of all the elements of this difference vector were computed and the mean value, defined as *rmean*, was found. In the same way, we can define the *actmean* for the difference vector ($A_{act} - F_{act}$), where A_{act} is the vector with the values of the diagnostic output of the controller and F_{act} is the vector with the respective samples of the failure introduced at the actuator.

The training of the CDC controller was done in two steps. First, no failure signals participated in the training and the controller was trained, so that it tracks the reference input r at the plant output, and at the same time it gives 0 at the diagnostic output a_{act}. The weights found in this step were used as initial conditions for the second step, where the controller was trained for 200.000 iterations to meet both the control and diagnostic requirements. The behavior of the plant is shown in Fig. 3 - Fig. 4. Similar results were obtained, when a ramp function was used for the actuator failure. For all the simulations in this section a momentum term of 0.9 and a learning rate of 0.01 were used.

3.2. Weight Assignment on Control and Diagnostic Objectives

The BP algorithm minimizes the mean squared error between the actual outputs of the output layer and the de-

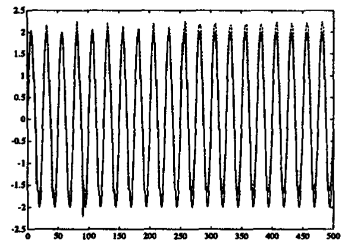

Figure 3: Reference input (solid line) and plant output (dotted line). Rmean=0.2045.

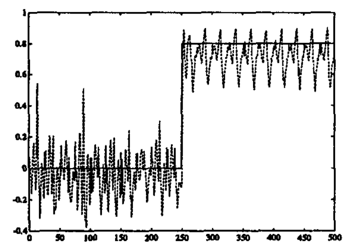

Figure 4: Actuator failure (solid line) and diagnostic output (dotted line). Actmean=0.1132.

sired outputs. We can assign weights to the square terms that participate in the sum, and by doing so we emphasize those terms that are assigned the largest weights. Since the CDC controller has two outputs, the control and the diagnostic, by assigning various weights to these two outputs, we simply train the controller in the direction we desire. Hence, depending on the specific application, we either assign a larger weight to the control objective, which results in a better tracking of the reference input at the output of the plant, or to the diagnostic objective, which results in a better and more reliable reproduction of the actuator failures, if any, at the output of the controller. Therefore, the minimizing quantity at the output of the controller is given as follows:

$$J = \frac{1}{2}\,\omega_1\ error_{a_{act}}^2 + \frac{1}{2}\,\omega_2\ error_{y_c}^2 \qquad (2)$$

where $error_{a_{act}}$, $error_{y_c}$ denote the errors at the diagnostic and control output of the CDC controller respectively.

In order to illustrate the above points, two cases were considered. Again, the weights obtained by training the controller as a conventional controller, that is by requiring the diagnostic output to be equal to zero, were used as initial condition and then the system was trained for additional 200,000 iterations in both cases. The results are given in Table. 1. We see that in the case, where the weight $\omega_1 = 0.3$ was assigned to the diagnostic output, the diagnostic capabilities of the controller became less powerful, whereas the reference tracking seemed to be quite

weights	rmean	actmean
0.2 $error_{y_c}$	0.2303	0.1122
0.3 $error_{a_{act}}$	0.2491	0.2260

Table 1: Rmean, actmean when assigning weights to the controller outputs. Actuator failures.

successful. In the case, where the weight $\omega_2 = 0.2$ was assigned to the second controller output, the diagnostic performance was excellent and the control performance was almost as good as in the first case.

4. Sensor Failures

4.1. Introduction

In this section we assume that we have only sensor failures. Our objective is to train a neural network model of the CDC controller that will have the following task:

"to achieve set point tracking, that is to follow the reference input r at the plant output y_p, and also to isolate the sensor faults and reproduce them at the diagnostic output a_{sens}."

A neural model of the plant was again used for the training, as illustrated in Fig. 5. The sensor failures account for the disturbances that occur at the output of the plant; hence it is these disturbances that we are trying to isolate at the diagnostic output a_{sens}. Note that here, the signal at the second output of the controller propagates through the layers of the neural plant and gets to its output, where it is added to the sensor failure, denoted by f_{sens}. The resulting signal is going to be the second general input to the CDC controller; the other general input is the reference signal. Note that the error at the output of the plant is easily computed, since we know that the expected plant output is the reference input at the controller, and then back-propagated to the plant input. One of the components of the error vector at the input of the plant is the error at the second output of the controller $error_{y_c}$. This error, together with the easily computed error at the diagnostic output of the controller are used by the BP algorithnm to train the neural network.

The internal structure of the neural model of the controller, the learning rate, the training and testing signals were the same as in the actuator case. The criterion for the evaluation of the diagnostic performance was the *sensmean*, defined for the difference vector F_{sens}-A_{sens} similarly to the actuator case. The controller was again trained initially as a conventional controller giving no signal at the diagnostic output a_{sens}, and noise that simulates the plant modelling errors, uniformly distributed in the range [-0.05, 0.05] was applied at the output of the plant for all the experiments. The simulation results were similar to the ones obtained for the actuator case and

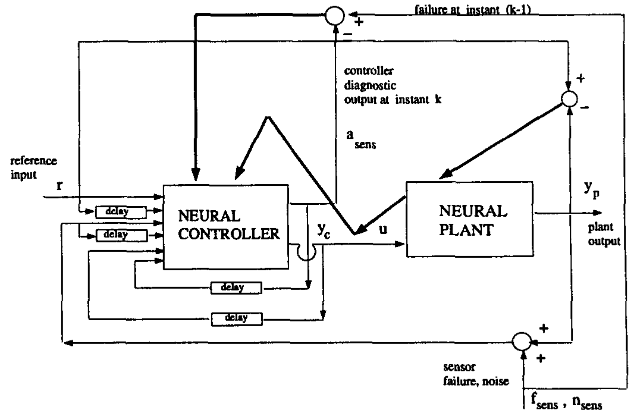

Figure 5: Training of neural CDC controller for sensor failures.

weights	rmean	sensmean
0.3 $error_{y_c}$	0.0979	0.1017
0.3 $error_{a_{sens}}$	0.0699	0.0969

Table 2: Rmean, sensmean when assigning weights to the controller outputs. Sensor failures.

therefore are ommited due to space limitations; for more details see [6], [8].

4.2. Weight Assignment on Control and Diagnostic Objectives

Now, the cost function is given as follows

$$J = \frac{1}{2}\,\omega_1\, error^2_{a_{sens}} + \frac{1}{2}\,\omega_2\, error^2_u \qquad (3)$$

since, as stated before, the error at the second output of the controller is just the error that corresponds to the input u of the plant. Here, we considered as initial condition for the training the layer-weights that were found before after 200,000 iterations of training of the controller, without assigning weights to its ouputs. This was done to see whether the behavior of the CDC controller for this specific plant would change, once we decided to emphasize either of its two objectives. Two cases were investigated. First the error at the diagnostic output of the controller was assigned a weight of $\omega_1 = 0.3$ and then the error at the input of the plant was assigned the same weight $\omega_2 = 0.3$. The results are given in Table. 2. We see that although the control behavior was evidently improved, by assigning the weight of $\omega_1 = 0.3$ at the diagnostic output, the diagnostic performance did not differ significantly in both cases. Hence, for this specific plant the initial training of 200,000 iterations was sufficient for the diagnostic performance of the controller, and the additional 200,000 iterations could not change the picture dramatically. The configuration with $\omega_1 = 0.3$ was then used to test ramp

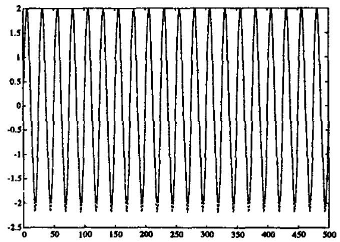

Figure 6: Reference input (solid line) and plant output (dotted line). Rmean=0.0719.

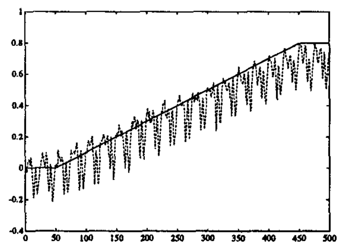

Figure 7: Sensor failure (solid line) and diagnostic output (dotted line). Sensmean=0.0870.

failures and the results are illustrated in Fig. 6, Fig. 7. Note that a momentum term of 0.1 and a learning rate of 0.01 were used for all the experiments in this section.

5. Conclusions

In this paper, a neural network implementation of the controller with diagnostic capabilities (CDC) has been presented for the first time, whose the major advantage is that it can also be applied to nonlinear plants. Here, a unified approach for both sensor and actuator failures has been introduced for the design of a controller, which is capable of reproducing failures at the actuator as well. With our design we have the flexibility of concentrating on one of the two objectives by appropriately emphasizing the contribution of this specific objective in the cost function to be minimized by the BP algorithm.

The methodology presented here does not handle simultaneous sensor and actuator failures. That would suggest a controller with three outputs, two outputs for diagnostics and one output for control. This seems to be quite a difficult computational problem and alternative forms of neural networks or training should be investigated. Note that we have not considered plant failures and have not provided stability theoretical results. We should state, however, that unless a reference signal or failure of magnitude much greater than those used in training are applied, our system is guaranteed (and has been successfully tested in numerous simulations) to meet the desired specifications, avoiding any instabilities. Finally, note that more details can be found in [8].

References

[1] P. J. Antsaklis, "Neural Networks for the Intelligent Control of High Autonomy Systems," *Intelligent Systems Technical Report 92-9-1*, Department of Electrical Engineering, University of Notre Dame, September 1992.

[2] J. Hertz, A. Krogh and R. G. Palmer, *Introduction to the Theory of Neural Computation.* Lecture Notes Volume I, Santa Fe Institute Studies in the Sciences of Complexity. Redwood City, CA: Addison-Wesley Publishing Company, 1991.

[3] Z. Hou, "Analysis of Auto Powertrain Dynamics and Modelling Using Neural Networks," M.Sc. Thesis, Department of Electrical Engineering, University of Notre Dame, February 1992.

[4] C. A. Jacobson and C. N. Nett, "An Integrated Approach to Controls and Diagnostics Using the Four Parameter Controller," *IEEE Control Systems Magazine*, vol. 11, no. 6, pp. 22-29, October 1991.

[5] A. E. Juarez, A. Ajbar and J. C. Kantor, "Multivariable Control with Integrated Diagnostics for Chemical Processes," *Process Systems Engineering*, PSE 91, Quebec, CA, 1991.

[6] I. K. Konstantopoulos, "Controller Design with Failure Diagnostic Capabilities via Neural Networks," M.Sc. Thesis, Department of Electrical Engineering, University of Notre Dame, November 1992.

[7] I. K. Konstantopoulos and P. J. Antsaklis, "The Four-Parameter Controller. A Neural Network Implementation," *IEEE Mediterranean Symposium on New Directions in Control Theory and Applications*, Chania, Crete, Creece, June 21-23, 1993.

[8] I. K. Konstantopoulos and P. J. Antsaklis, "Controllers with Diagnostic Capabilities. A Neural Network Implementation," *Journal of Intelligent and Robotic Systems.* To appear.

[9] S. R. Naidu, E. Zafiriou and T. McAvoy, "Use of Neural Networks for Sensor Failure Detection in a Control System," *IEEE Control Systems Magazine*, vol. 10, no. 3, pp. 49-55, April 1990.

[10] K. S. Narendra and K. Parthasarathy, "Identification and Control of Dynamical Systems Using Neural Networks," *IEEE Transactions on Neural Networks*, vol. 1, no. 1, pp. 4-27, March 1990.

[11] C. N. Nett, "Algebraic Aspects of Linear Control System Stability," *IEEE Transactions on Automatic Control*, vol. AC-31, no. 10, pp. 941-949, October 1986.

[12] K. P. Valavanis, C. A. Jacobson and B. Gold, "Integration Control and Failure Detection with Application to the Robot Payload Variation Problem," *Journal of Intelligent and Robotic Systems 4*, pp.145-173, 1991.

[13] K. Warwick, G. W. Irwin and K. J. Hunt, eds, *Neural Networks for Control and Systems.* IEE Control Engineering Series. London, UK: Peter Peregrinus Ltd., 1992.

[14] X. Wu and A. Cinar, "Reliable Process Control by 4-Parameter Controllers," *Proceedings of 1991 American Control Conference*, pp. 1860-1865, 1991.