EGG-EE--8935

DE91 001844

NEURAL NETWORK SETPOINT CONTROL
OF AN
ADVANCED TEST REACTOR EXPERIMENT LOOP SIMULATION

DOE Contract No. DE-AC07-76ID01570

Gail A. Cordes, Principal Investigator
Steven R. Bryan, Robert H. Powell, Donald R. Chick

Idaho National Engineering Laboratory
Idaho Falls, Idaho 83415

Prepared for the
U. S. Department of Energy

MASTER

crb
DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

ABSTRACT

This report describes the design, implementation, and application of artificial neural networks to achieve temperature and flow rate control for a simulation of a typical experiment loop in the Advanced Test Reactor (ATR) located at the Idaho National Engineering Laboratory (INEL). The goal of the project was to research multivariate, nonlinear control using neural networks. A loop simulation code was adapted for the project and used to create a training set and test the neural network controller for comparison with the existing loop controllers. The results for three neural network designs are documented and compared with existing loop controller action. The neural network was shown to be as accurate at loop control as the classical controllers in the operating region represented by the training set.

ACKNOWLEDGEMENTS

This work was performed under the Exploratory Research and Development Program (ERDP) of the Idaho National Engineering Laboratory. The authors wish to acknowledge the encouragement and direction of Clyde Toole, ERDP Program Manager, the INEL Defense SPU committee, and Mike Bray, Unit Manager for the Artificial Intelligence Staff. We also wish to acknowledge Lee Bowen for his loop simulation code which was modified for this work, and the engineers and scientists at the ATR for their interest and support as the work progressed.

CONTENTS

ABSTRACT.....	ii
ACKNOWLEDGEMENTS.....	iii
ACRONYMS.....	vi
1.0 INTRODUCTION.....	1
2.0 NEURAL NETWORK TECHNOLOGY DESCRIPTION.....	4
3.0 LOOP SIMULATION CODE.....	13
4.0 NEURAL NETWORK CONTROLLERS.....	25
4.1 Series 4 Results.....	27
4.2 Series 5 Results.....	32
4.3 Series 6 Results.....	40
5.0 COMPARISONS.....	47
6.0 CONCLUSIONS.....	60
7.0 REFERENCES.....	62

FIGURES

1. Benchmark sequence.....	3
2. Diagram of a biological neuron.....	5
3. Diagram of an example neural network.....	7
4. Diagram of one processing unit in the middle layer.....	8
5. Simplified schematic diagram of Loop 1-D.....	14
6. Graphical interface to the loop simulation.....	16
7. Simulation flowrate response to the benchmark sequence using the classical controllers.....	17
8. Simulation temperature response to the benchmark sequence using the classical controllers.....	18
9. Diagram of simulation with existing controllers.....	20
10. Portion of the flow data from the training set.....	21
11. Portion of the temperature data from the training set.....	22
12. Portion of the gamma heat variations from the training set.....	23

13. Diagram of simulation with neural network controllers.....	24
14. Flow control by Series 4 neural network.....	30
15. Temperature control by Series 4 neural network.....	31
16. Flow control by the Series 5 neural network 5B2.....	35
17. Temperature control by Series 5 neural network 5B2.....	36
18. Flow control by the Series 5 neural network 5D1.....	37
19. Temperature control by the Series 5 neural network 5D1.....	38
20. Flow control by Series 6 neural network 6A1.....	43
21. Temperature control by Series 6 neural network 6A1.....	44
22. Flow control by Series 6 neural network 6B1.....	45
23. Temperature control by Series 6 neural network 6B1.....	46
24. Time slice plot of temperature control by classical controllers.....	48
25. Time slice plot of temperature control by Series 4 neural network...	49
26. Time slice plot of temperature control by Series 5 neural network 5B2.....	50
27. Time slice plot of temperature control by Series 6 neural network 6A1.....	51
28. Time slice plot of temperature control by Series 6 neural network 6B1.....	52

TABLES

1. Summary of Characteristics for the Figure 3 Network.....	9
2. Input and Output Signals for Series 4.....	29
3. Input and Output Signals for Series 5	33
4. Input and Output Signals for Series 6.....	41
5. Comparison of Controller Performance.....	54
6. Summary of Input and Output Signals for Series 4, 5, and 6.....	57
7. Comparison of Training Set and Benchmark Sequence.....	59

ACRONYMS

ATR Advanced Test Reactor
DOE Department of Energy
ERDP Exploratory Research and Development Program
INEL Idaho National Engineering Laboratory

**NEURAL NETWORK SETPOINT CONTROL
OF AN
ADVANCED TEST REACTOR EXPERIMENT LOOP SIMULATION**

1. INTRODUCTION

This report documents an application of neural network technology to temperature and flow rate control of a nuclear reactor experiment loop simulation. An artificial neural network is an approach to machine learning using computer hardware and software that implements features of the biological neural networks of the human brain.

Neural networks are important to research at the Idaho National Engineering Laboratory (INEL) because neural networks can learn to characterize underlying, general relationships from specific examples of the pattern being analyzed, without a detailed mathematical modeling of the relationship. The Artificial Intelligence Staff have applied neural network technology to increasingly challenging tasks, with the goal of developing neural networks for application to system control.^{1,2} Funding for FY-1990 research on multivariate, nonlinear control was received under the INEL Exploratory Research and Development Program (ERDP) for a workscope which included application of neural network technology to setpoint control of an ATR experiment loop simulation.

The Advanced Test Reactor (ATR) is a Department of Energy (DOE) research reactor located at the Test Reactor Area on the INEL. Experiment packages are brought to the ATR from around the world for irradiation in the experiment loops of the reactor. Each experiment is individually exposed to loop conditions of flow, temperature, and neutron flux as specified by the sponsor.

Therefore, precise control of the ATR loop flow and temperature is essential to accomplishing the mission of the ATR.

The purpose of the ERDP workscope was to design and train several neural networks and compare their performance with that of the existing, classical controllers currently in use at the ATR. Conducting an exhaustive study of this control problem to identify the most efficient neural network learning rule, architecture, input vector, and training set was beyond the funding level of the project. As this report documents, a traditional, nonoptimal approach to neural network design was used in order to quickly investigate the underlying question of whether a neural network can indeed be trained for multivariate, nonlinear control.

To compare the neural network controllers with the existing controllers, a realistic benchmark file of setpoint changes and gamma heat variations was developed. This benchmark represented a complex and challenging sequence of actual loop operations, as determined from conversations with loop engineers and ATR loop procedure documentation.³ The sequence of setpoint changes and gamma heat variations which comprised the benchmark are plotted in Figure 1.

The following sections briefly introduce the emerging technology of neural networks, discuss the modification of the ATR experiment loop simulation, detail the design and training experience of three neural networks, and present the resulting comparisons of the performance of the neural networks and the existing loop controllers.

Figure 1. Benchmark sequence.

2.0 NEURAL NETWORK TECHNOLOGY DESCRIPTION

A neural network is an electronic simulation of the biological neural networks in the human brain.⁴ Artificial neural systems apply the biological design of neuron cells to computer systems in an attempt to achieve the learning capability, speed, compactness, and versatility of the brain.

The fundamental unit of the brain is the neuron, which communicates with another neuron by passing a signal over a gap or synapse between the two cells. Each neuron communicates with more than one other neuron, with communications being performed independently and in parallel. The biological neural network is incredibly complex, but is extremely fast; it learns, adapts to changing inputs, responds to imprecise data, and continues to accurately respond despite degradation or damage to portions of the system.

Figure 2 diagrams a biological neuron. The input signals to the cell arrive at the cell extensions called dendrites. The input signals are the output signals from multiple other cells. Each input signal must pass over a gap called a synapse. The passage of the signal modifies the strength of the synapse. Since knowledge is stored as synaptic strengths, this modification of synaptic strength represents learning. The cell sums all arriving signals and outputs a signal along the cell extension called an axon. This output signal represents the level of activation of the cell caused by the input signals. The axon presents the output signal to many other cells. The human brain contains on the order of 10^{12} neurons, each with connections to an estimated 60,000 other cells.

Research at the INEL has focused on simulating neural networks with software on existing digital computers. Algorithms are used to model the massive parallelism, the multiple connections, the varying signal intensities, and the learning features of the brain.

Figure 2. Diagram of a biological neuron.

The neural network approach described here is a feed-forward network characterized by layers of interconnected units operating in parallel. Input and output signals vary in intensity for each unit according to weights that are assigned to the signal. These weights are unique to each signal path and are modified by the algorithm to model the biological learning process. That is, the algorithm adjusts the weights as the computation proceeds to achieve learning in the artificial neural network. The varying signal intensities also model the biological variation in conductance across the synapse or gap between neurons in contrast to the absolute on/off of digital computer bits.

A neural network can be diagrammed as in Figure 3. The illustrated example network has three layers: the input layer of four units, the hidden layer of six units, and the output layer of three units. Each unit is connected with every unit in the adjacent layers, but not with units in the same layer. In this manner, the massive interconnectedness and parallel operation of the human brain are modeled.

The total numbers of processing units, the input and output signals, and the weights for this simple example are summarized in Table 1.

The weight values, which are associated with each connection to model the synaptic strength of the biological neuron connections, are shown in Figure 4. This figure is a diagram of one processing unit in the middle layer of the example network.

Neural networks learn to characterize underlying, general relationships from specific examples of the pattern being analyzed, without a detailed mathematical model of the relationship. The pattern might involve a mapping of measurable system variables to appropriate control signals, a mapping of speech characteristics to phoneme identity, or a mapping of experimental data

Figure 3. Diagram of an example neural network.

jth Processing Unit in Layer s

$$\text{Sum}_j = \sum_i w_{ji}^{[s]} o_i^{[s-1]}$$

$$\text{Activation}_j = f(\sum_i w_{ji}^{[s]} o_i^{[s-1]}) = \frac{1}{1 + e^{-(\sum_i w_{ji}^{[s]} o_i^{[s-1]} + C)}}$$

009 0532

Figure 4. Diagram of one processing unit in the middle layer.

TABLE 1. SUMMARY OF CHARACTERISTICS FOR THE FIGURE 3 NETWORK

<u>Layer</u>	<u>Units</u>	<u>Input Signals</u>	<u>Output Signals</u>	<u>Weights</u>
Input layer	4	4	4	0
Middle layer	6	24	6	24
Output layer	3	18	3	18
Total units:	13			
Total connections:	42			
Total weights:	42			

values to data validation criteria. By definition, the network has learned a set of input data when it produces a corresponding set of output data within some predetermined level of accuracy compared with the expected output.

Many learning rules are used to train neural networks. The most commonly used one, and the one used for the work described in this report, is the error backpropagation learning rule, which can be described as supervised learning. This learning rule does not model the learning process of the human brain, but it is a reliable method for achieving learning in the artificial neural network. Application of this learning rule proceeds as follows.

Initially, all connection weights are set to some nominal or random value. The training data sets are presented randomly to the network at the input units. For a given data set, the input units each send a signal to every middle unit. These signals are varied by the weights associated with each connection so all units in the next layer perceive the same set of signals, but with different intensities. The middle units sum the input signals they receive. Each middle unit then processes the resultant sum through a transfer function to produce a unique output signal that it transmits to every output unit in the next layer. Weights are again applied to the signals, and the output unit sums all of its inputs, processes the sum, and prepares the output value for the unit. The values from the output units constitute the output data vector that is compared with the expected output vector associated with the input data initially presented to the system.

If the calculated output matches the expected output, the weights are left unchanged, and the second data point is initiated. If they do not match, an error is calculated. This error is backpropagated through the network by the learning rule that adjusts the connection weights between the output layer and middle layer and between the middle layer and input layer to reduce the error.

The adjusted weights are stored and used for the second data point. The same procedure of forward propagation of the input, output vector comparison, error backpropagation, and weight readjustment is followed.

The learned behavior of the network is represented by the final weights stored in the system after all the training data have been randomly processed tens, hundreds, or perhaps thousands of times. The trained network then uses the stored weights to process future input data.

Designing the best network architecture and learning rule for a given application and then training that network is an inexact science requiring both a knowledge of the application area and expertise with neural networks. However, the network design and the learning algorithms are mathematically simple and can operate in parallel on parallel machines. Information can therefore be rapidly transmitted from the input to the output of a trained network. This means that a system control application can be expected to perform in real time even while using a large number of system parameters as input variables.

Knowledge is distributed throughout the trained network and represents a generalization of the training information. Therefore, a neural network can recognize an underlying pattern even when portions of the input data are distorted or erased before presentation to the trained network. Because the network consists of multiple processing units and connections, it is tolerant of hardware problems in a small subset of the units. Such graceful degradation (versus catastrophic failure) makes neural networks an attractive option for autonomous control applications.

The training algorithms remain in place after the neural network is operational. The neural network thus is retrainable if the character of the input data changes or if the neural network performance degrades due to

failure or changed characteristics of one or more units. This is an important characteristic of the neural network in that it retains robustness even when operating under conditions different from those in which it was installed.

Recent and continuing developments in parallel processing hardware, including optical computers, promise to provide hardware implementations of neural networks that are compact and lightweight and require low amounts of power for operation.

3. LOOP SIMULATION CODE

This section provides a physical description of an ATR experiment loop, and a functional description of the loop simulation code.

The ATR has nine experiment loops, each operated individually to provide sponsor-specified temperature, flow rate, and neutron flux test conditions for the experiment packages. That is, the temperature and flow rate in each loop and neutron flux at the inpile tube are controlled independently of the other loops.

This research work studied temperature and flow rate setpoint control for the loops. The design of each loop is unique to that loop; therefore, the response to control actions for each loop is different. A representative loop, Loop 1-D, was selected for the study. A simplified schematic diagram of Loop 1-D is shown in Figure 5.

As the diagram indicates, the loop coolant flows through the pump, through the clamshell heaters, and into the inpile tube where the experiment packages are irradiated. The coolant exits the inpile tube, passes through a flow-regulating valve, and then encounters a diversion valve.

Here the flow is divided, with part going through the heat exchanger and part through the bypass. The flow streams recombine at the mixing tee, upstream from the pump. Thermocouples and venturi flow meters monitor temperatures and flow rates at various locations around the loop.

The measured temperatures and flow rates, the rate of change of these parameters, the positions of the valves, and the current status of the heaters constitute available state variables for control purposes. Both feedback and

Figure 5. Simplified schematic diagram of Loop 1-D.

feedforward control algorithms are used to change the valve positions in order to regulate the flow rate in the loop and the amount of flow (and therefore the heat removal rate) through the heat exchanger to regulate the temperature. Because of the loop design and its feedback characteristics, the loop can be characterized as nonlinear. The two controllable variables classify the loop as multivariate.

A loop simulation package⁵ written in Basic for an IBM personal computer was already in existence at the ATR when the work began. This code was converted to C language for a Hewlett Packard 9000/835 machine. Extensions to the software made it more user friendly and compatible with the neural network input/output software. Further documentation of the simulation package is found in Reference 5.

A graphical interface with the simulation was provided to allow the user to change temperature and flow setpoints and graphically observe the temperatures and flow rates as they respond to the action of the controllers. A snapshot of this graphical interface is shown in Figure 6.

The loop simulation package was used for three distinct purposes.

First, the simulation modeled the response of the system to actions of the existing, classical controllers to control loop flow and temperatures. The simulation output was reviewed by the ATR loop engineers and operators to ensure the simulation indeed provided an accurate model of the loop design, controllers, and loop parameter responses. The benchmark sequence of setpoint changes and gamma heat variations was presented to the classical controllers. The flow and temperature responses of the system to the benchmark sequence are shown in Figures 7 and 8. These responses are used in the section on comparisons for an evaluation of the neural network controller performances.

Figure 6. Graphical interface to the loop simulation.

Figure 7. Simulation flowrate response to the benchmark sequence using the classical controllers.

Figure 8. Simulation temperature response to the benchmark sequence using the classical controllers.

Second, the simulation was used with the classical controllers to develop a comprehensive training set for the neural network. This involved ramping the loop parameters through a wide range of setpoints and gamma heat levels and recording in a data file the system state variables and controller actions needed for network training. A diagram of the data flow for this application of the simulation is shown in Figure 9. The training set represents 504,510 seconds, or 5.84 days, of real time operation of the experiment loop. These data were obtained at intervals of 0.5 seconds, and every twentieth data record was saved in the training set file. Thus, 50,451 data records are available in the training set. Portions of the training set are plotted in Figures 10, 11, and 12 showing the flow, temperature and gamma heat data, respectively.

Third, the simulation was linked to the trained neural network so the neural network provided control commands to the loop valves, replacing the existing controller functions. A diagram of the data flow for this step is diagrammed in Figure 13. The neural network was challenged to control temperature and flow rate in the loop through the benchmark sequence of setpoints and gamma heat variations. An important point is that the training set was completely different from the benchmark sequence so the performance of the neural networks on the benchmark sequence represented a generalization of the knowledge acquired from the training set. The results of this step were dependent on the training set used, the accuracy of the weights developed from each training session, and the design of the neural network. A complete discussion of the three neural network designs and their final control actions is given in the following section.

At the end of the study, the loop simulation code was made available to the ATR staff.

Figure 9. Diagram of simulation with existing controllers.

Figure 10. Portion of the flow data from the training set.

Figure 11. Portion of the temperature data from the training set.

Figure 12. Portion of the gamma heat variations from the training set.

Figure 13

Figure 13. Diagram of simulation with neural network controllers.

4.0 NEURAL NETWORK CONTROLLERS

Existing INEL neural network software designed with feedforward layers of neurons was updated for this project. The modular program is written in C language for the Hewlett Packard 9000/835 computer. The neural network program communicated with the loop simulation code through a shared memory segment. The program's modular design made it possible to research various network designs, nonlinearity functions, and learning rules without the necessity for lengthy revisions of the code.

As work on this project progressed, the design of the neural network was modified by varying the vector of input variables, the number of processing units in the middle or middle layer, and the vector of output variables.

Each network design was assigned a numerical series designation, with Series 2 being the first design and Series 6 being the final design. (Series 1 was used to identify the response of the simulation code to the benchmark sequence using the classical controllers.) The results of Series 2 and 3 represented preliminary research with initial versions of the ATR loop simulation code and the training set. Series 4, 5 and 6 were researched using the final version of the simulation code and a more complete training set.

The neural networks in Series 4, 5, and 6 were fully connected, feedforward networks trained with the error backpropagation learning rule. Three layers were used for each network, with different numbers of nodes in the layers for each series. The sigmoid transfer function with a constant gain and bias was used at the neurons in the middle and output layers to determine the level of activation of the nodes and thus the output signals.

The neural networks were trained with random presentations of the training set records. The same training set was used to train all three neural networks, but the selection of parameters from the records differed with each network design. A total of 5 million training cycles was used for each training sequence. Since one cycle represented the presentation of one record to the neural network and the records were chosen randomly from the training set file, the training session amounted to presenting each record to the network an average of 82 times.

The results for these three series are given separately below, with a comparison of the results provided in the next section.

4.1 Series 4 Results

The Series 4 network provided the least accurate controller for the simulation. The network was built with 11 nodes in the input layer, 30 nodes in the middle layer, 2 nodes in the output layer, and 1 bias node fully connected to both the middle and output layers. This network design provided 422 weights at the node connections.

The input variables included a selection of loop parameters and their 10-second differences plus the specified setpoints. The assortment of simulation parameters selected for this network represented direct sensor output data plus calculated parameter values at points around the loop. The differences were calculated as the difference between the current parameter value and the parameter value in the previous training set record, a time difference of 10 seconds.

The output variables of the network were control commands expressed in terms of the variable flow coefficient C_v for the two loop valves, directing the valves to open or close by a specified amount in order to attain and maintain the flow and temperature setpoints. The demand signal for the temperature control valve was the total request in C_v . The demand signal for the flow control valve was the increment in C_v over the previous time step.

The input and output variables are listed in Table 2. The input signals and their descriptions can be interpreted through reference to the simplified diagram of the loop in Figure 5.

The network was trained with 5 million random presentations of the training set records, and the weights were stored for future use. To test the performance of the trained controller, the network was linked to the simulation code to perform the control actions in place of the conventional

controllers. The simulation was presented with the benchmark sequence of flow and temperature setpoints and gamma heat variations, with the resulting performance as shown in Figure 14, flow response, and Figure 15, temperature response.

As the plots show, the neural network was able to hold the flow and temperature of the loop close to the setpoints, and even gave a reasonable response when challenged with the reactor scram simulation at the end of the sequence.

TABLE 2. INPUT AND OUTPUT SIGNALS FOR SERIES 4

<u>Node</u>	<u>Input Signal</u>	<u>Description^a</u>
1	FE-6	Heat exchanger flow rate
2	FE-6 delta	Change in heat exchanger flow rate
3	TE-31	Heat exchanger exit tee temperature
4	TE-31 delta	Change in heat exchanger exit tee temperature
5	TE-31 setpoint	Temperature setpoint at the mixing tee
6	TE-37-6-2	Inpile tube inlet temperature
7	TE-37-6-2 delta	Change in inpile tube inlet temperature
8	FE-2	Total loop flow
9	FE-2 delta	Change in total loop flow
10	FE-2 setpoint	Flow setpoint for total loop flow
11	Gamma heat	Gamma heat specified in the inpile tube

<u>Node</u>	<u>Output Signal</u>	<u>Description</u>
1	FCV-1 request delta	Requested C_v change for valve FCV-1
2	TCV-31 request	Requested C_v for valve TCV-31

- a. The data changes are calculated by taking the difference in value in one time step. Time steps are uniformly 0.5 second.
-

Figure 14. Flow control by Series 4 neural network.

Figure 15. Temperature control by Series 4 neural network.

4.2 Series 5 Results

The neural network constructed for Series 5 responded better than Series 4 to the benchmark sequence. The network was built with 18 input layer nodes, 51 middle layer nodes, 2 output layer nodes, and 1 bias node fully connected to both the middle and output layers. This network design provided 1073 weights at the node connections.

For this network, the input variables included, in addition to the setpoints, only those gamma heat, temperature, flow, and valve parameter values that represented actual sensor output from the loop instrumentation. These parameters and their differences over one 0.5-second time step were input to the network along with a ratio of the inpile tube exit temperature to the mixing tee temperature. This selection of variables limited the controller to information available directly from the actual loop data system. The temperature ratio was included to provide additional, redundant information on the temperature distribution.

The middle or middle layer was deliberately large to assure that no information would be lost for lack of middle layer nodes and connections.

The output variables were the requested flow coefficients, C_v , for the flow control valve and temperature control valve.

The input and output variables are summarized in Table 3. The signals and their descriptions can be further interpreted through reference to the simplified loop diagram in Figure 5.

The neural network was trained through 5 million presentations of simulation data records that were randomly selected from the training set. In order to

TABLE 3. INPUT AND OUTPUT SIGNALS FOR SERIES 5

<u>Node</u>	<u>Input Signal</u>	<u>Description^a</u>
1	FE-6	Heat exchanger flow rate
2	FE-6 delta	Change in heat exchanger flow rate
3	TE-37-8-1	Heat exchanger exit temperature
4	TE-37-8-1 delta	Change in heat exchanger exit temperature
5	TE-31	Heat exchanger exit tee temperature
6	TE-31 delta	Change in heat exchanger exit tee temperature
7	TCV-31	Present C _v request for valve TCV-31
8	TE-31 setpoint	Temperature setpoint at the mixing tee
9	TE-42	Inpile tube exit temperature
10	TE-42 delta	Change in inpile tube exit temperature
11	TE-37-6-2	Inpile tube inlet temperature
12	TE-37-6-2 delta	Change in inpile tube inlet temperature
13	FE-2	Total loop flow
14	FE-2 delta	Change in total loop flow
15	FCV-1	Present C _v request for valve FCV-1
16	FE-2 setpoint	Flow setpoint for total loop flow
17	Gamma heat	Gamma heat specified in the inpile tube
18	TE-31 / TE-42	Ratio of heat exchanger exit temperature to inpile tube exit temperature
<u>Node</u>	<u>Output Signal</u>	<u>Description</u>
1	FCV-1 request	Requested C _v for valve FCV-1
2	TCV-31 request	Requested C _v for valve TCV-31

a. The data changes are calculated by taking the difference in value in one time step. Time steps are uniformly 0.5 second.

check the variations that might result from the initial randomization of the weights, the network was trained three times. The three sets of final weights were each used with the neural network controller to drive the simulation code through the benchmark sequence.

When the results from these three runs were plotted, the three sets of weights were obviously different and produced different temperature and flow responses of the loop simulation to the benchmark sequence. The results which were judged most reasonable were those from the neural network in run 5B2. The flow time history of this neural network controller plotted along with the flow setpoint request is shown in Figure 16. The temperature setpoint requests and the temperature response of the simulation to the neural network controller actions are shown in Figure 17.

This controller provided better setpoint control than Series 4 but exhibited a larger temperature excursion during the reactor scram simulation in the final seconds of the benchmark.

The question of whether the network had been adequately trained with 5 million training cycles (an average of 82 presentations of each record in the training set) was addressed by continuing the training of network 5B2 through 50 million additional cycles. This additional training time gave the network an average of 900 presentations of each training set record.

The neural network controller developed by the this additional training time was designated 5D1. The 5D1 controller was linked to the simulation to run the benchmark sequence, and the flow and temperature responses of the simulation to the setpoint requests and gamma heat variations are given in Figures 18 and 19, respectively.

Figure 17. Temperature control by Series 5 neural network 5B2.

Figure 18. Flow control by the Series 5 neural network 5D1.

Figure 19. Temperature control by the Series 5 neural network 5D1.

The 5D1 controller exhibited much tighter setpoint control with a faster time response, less offset and less overshoot in most of the sequence. However, a greater offset occurred in the 60,000 to 80,000-second time range, and the controller response to the reactor scram at the end of the sequence allowed the flow and temperature to depart widely from the setpoint values. The data indicate that the controller would have eventually returned the loop parameters to the setpoint values with a longer steady state run.

4.3 Series 6 Results

The Series 6 neural network controller proved to be the most accurate of the three neural network controllers at achieving and maintaining temperature and flow setpoints for the loop. The network architecture provided 20 nodes in the input layer, 51 nodes in the middle layer, 2 nodes in the output layer, and 1 bias node fully connected to both the middle and output layers. This network design provided 1175 weights at the node connections.

The input variable list was similar to that of the Series 5 networks in that it included all of the Series 5 list plus two averages of temperature values. The heat exchanger exit temperature was averaged over 50 seconds, and the inpile tube exit temperature was averaged over 200 seconds. These averaged values were intended to give the network a more complete sense of time variance of the variables than was provided by the temperature ratio alone.

The middle layer contained the same number of nodes as the middle layer for Series 5.

The output layer was the same as Series 4, with one signal being the requested C_v change for flow control valve FCV-1 and the other signal being the total requested C_v for temperature control valve TCV-31.

The input and output variables are summarized in Table 4. The signals and their descriptions can be further interpreted through reference to the simplified loop diagram in Figure 5.

The neural network was again trained through 5 million presentations of randomly selected data records from the training set. The neural network with this set of weights was designated Series 6A1 and was allowed to drive the

TABLE 4. INPUT AND OUTPUT SIGNALS FOR SERIES 6

<u>Node</u>	<u>Input Signal</u>	<u>Description^a</u>
1	FE-6	Heat exchanger flow rate
2	FE-6 delta	Change in heat exchanger flow rate
3	TE-37-8-1	Heat exchanger exit temperature
4	TE-37-8-1 delta	Change in heat exchanger exit temperature
5	TE-31	Heat exchanger exit tee temperature
6	TE-31 delta	Change in heat exchanger exit tee temperature
7	TCV-31	Present C_v request for valve TCV-31
8	TE-31 setpoint	Temperature setpoint at the mixing tee
9	TE-42	Inpile tube exit temperature
10	TE-42 delta	Change in inpile tube exit temperature
11	TE-37-6-2	Inpile tube inlet temperature
12	TE-37-6-2 delta	Change in inpile tube inlet temperature
13	FE-2	Total loop flow
14	FE-2 delta	Change in total loop flow
15	FCV-1	Present C_v request for valve FCV-1
16	FE-2 setpoint	Flow setpoint for total loop flow
17	Gamma heat	Gamma heat specified in the inpile tube
18	TE-31 / TE-42	Ratio of heat exchanger exit temperature to inpile tube exit temperature
19	TE-31 average	Heat exchanger exit temperature averaged over 50 seconds
20	TE-42 average	Inpile tube exit temperature averaged over 200 seconds
<u>Node</u>	<u>Output Signal</u>	<u>Description</u>
1	FCV-1 request delta	Requested C_v change for valve FCV-1
2	TCV-31 request	Requested C_v for valve TCV-31

a. The data changes are calculated by taking the difference in value in one time step. Time steps are uniformly 0.5 second.

simulation through the benchmark sequence. The simulation flow response is shown in Figure 20 and the temperature response is shown in Figure 21. The temperature response had the same characteristics as previous runs, but with less offset and smaller overshoot values.

The training sequence for Series 6A1 was continued through an additional 50 million cycles, as in Series 5D1. The neural network had seen each training set record an average of about 900 times by the end of this sequence. The resulting neural network with the weights from the extended training sequence was designated Series 6B1. The flow response of the simulation to this network controller is found in Figure 22, and the temperature response is found in Figure 23.

As noted in Series 5, the longer training time improved the controller performance in some areas and degraded the performance in others.

Figure 20. Flow control by Series 6 neural network 6A1.

Figure 21. Temperature control by Series 6 neural network 6A1.

Figure 22. Flow control by Series 6 neural network 6B1.

Figure 23. Temperature control by Series 6 neural network 6B1.

5.0 COMPARISONS

In this section, the neural network controllers are compared with each other and with the classical controller.

All controllers managed flow control of the simulated loop similarly and accurately, with the differences in the responses of the simulation loop being of small magnitude. In contrast, the temperature control of the simulated loop varied with the different controllers. Therefore, only the temperature response of the simulation to controller setpoint control was used for comparison of controller capabilities.

For this comparison, time slice plots of the temperature response of the simulated loop to the controllers have been prepared and included as Figures 24 through 28. The classical controller is characterized in Figure 24. The neural networks selected for comparison are the Series 4 network (Figure 25), Series 5B2 network (Figure 26), Series 6A1 network (Figure 27) and Series 6B1 network (Figure 28).

The time slice presented in the plots is the portion of the benchmark sequence that represents actual operating ranges and conditions for the Loop 1-D. This excludes the initial, startup portion of the sequence and the final, reactor scram portion. All controllers behaved in similar fashion during startup while behavior during the scram varied so widely that a careful comparison was inappropriate.

Six measurements are used in this comparison. The comparisons are summarized in Table 5 and discussed in the following paragraphs.

Figure 24. Time slice plot of temperature control by classical controllers.

Figure 25. Time slice plot of temperature control by Series 4 neural network.

Figure 26. Time slice plot of temperature control by Series 5 neural network 5B2.

Figure 27. Time slice plot of temperature control by Series 6 neural network 6A1.

Figure 28. Time slice plot of temperature control by Series 6 neural network 6B1.

The first measurement is the delay time for the simulation to reach 580°F when the setpoint was changed from 555°F to 580°F at about 38,000 s. During this response time the flow setpoint was constant at 78 gpm and the gamma heat was constant at 50 BTU/s.

The second measurement is the amount of temperature overshoot past the 580°F setpoint. During this portion of the benchmark the flow was still constant at 78 gpm, and the gamma heat remained at 50 BTU/s.

The third measurement is the offset in temperature from the 580°F setpoint at about 48,000 s while the flow was constant at 82 gpm and the gamma heat was still 50 BTU/s.

The fourth comparison point is the delay time until the simulation temperature reached 560°F at about 52,000 s. The temperature setpoint was changed to 560°F and the flow setpoint was changed to 72.25 gpm while the gamma heat remained 50 BTU/s.

The fifth measurement used in the comparison is the amount of temperature undershoot past 560°F. During this time, the flow setpoint remained at 72.25 gpm, and the gamma heat was 50 BTU/s.

The sixth measurement is the temperature offset at about 64,000 s, a time when the flow setpoint was 65 gpm, the temperature setpoint was constant at 560°F, and the gamma heat was varied to 49 BTU/s and 51 BTU/s.

With reference to Table 5 where these measurements are summarized for the controllers, the Series 4 neural network controller was the least effective at maintaining setpoint control. The Series 5 architecture with the set of weights designated 5B2 was more effective, but in all measurements the

TABLE 5. COMPARISON OF CONTROLLER PERFORMANCE

<u>Measurement</u>	<u>Classical</u>	<u>Series 4</u>	<u>Series 5B2</u>	<u>Series 6A1</u>	<u>Series 6B1</u>
580°F delay	970 sec	3120 sec	1620 sec	1580 sec	1420 sec
580°F overshoot	+ 5.68°F	+ 4.76°F	+ 3.89°F	+ 3.45°F	+ 4.42°F
580°F offset	+ 0.01°F	+ 2.03°F	+ 0.23°F	- 0.45°F	+ 0.21°F
560°F delay	1590 sec	--- ^a	2130 sec	1900 sec	2120 sec
560°F undershoot	- 2.61°F	--- ^a	- 0.78°F	- 1.68°F	- 0.39°F
560°F offset	0.00°F	+ 3.14°F	+ 1.66°F	+ 0.35°F	+ 1.48°F

a. The temperature did not reach 560°F.

Series 6 architecture provided a better simulated loop response to temperature setpoints.

Two examples of the Series 6 architecture are included, 6A1 and 6B1. Both the Series 6A1 and 6B1 neural network controllers matched the performance of the classical controller. The classical controller produced faster response to setpoint changes and essentially no offset at steady state conditions while the neural network controllers provided less overshoot and undershoot of the setpoint values.

The difference between the 6A1 weights and the 6B1 weights results from the length of training time. As discussed previously, the 6B1 weights were developed from the 6A1 weights with an additional training time 10 times as long as the initial 5 million cycles. The 6B1 weights should have given superior control performance because of the longer training time. Series 6B1 did provide faster response to the setpoint increase (measurement 1), less offset to the 580°F setpoint (measurement 3), and less undershoot to the setpoint change to 560°F (measurement 5). However, it allowed a higher overshoot to the 580°F setpoint request (measurement 2), a slower response to the 560°F setpoint request (measurement 4), and a noticeable offset when responding to small changes in gamma heat (measurement 6). Possible reasons for these differences are discussed below.

Based on past experience with neural networks by the authors and reports in the published literature, the performance of the neural network controllers could be improved through a variation of the network architecture, an optimal training time, and a better crafted training set.

For this study, the architecture of the neural networks was varied by changing the number of nodes in the input and middle layers. The networks with an extended set of input variables provided better control signals to the loop

simulation. For comparison, Table 6 lists these variables for the three network designs. Series 6 architecture included averaged values of the temperatures and performed better than Series 5 without these averaged values. Both performed better than the limited architecture of Series 4. Some neural network research⁶ suggests that these averaged values are important in providing temporal relationships to the input data. The effect of middle layer size on the generalization ability of the neural networks was not studied.

The length of training time might have affected the results of this project. An overtrained network can lose its ability to generalize while an undertrained, or improperly trained, network might not have reached the minimum on its error surface.^{7,8} Some variations in learning time were tried during this research, but the learning rule was not modified to reduce training time or to avoid local error minima.

The training set is of central importance to the final performance of the neural network when the error backpropagation learning rule is applied as it was in this project.⁹ Researchers in the field are still trying to determine which characteristics are necessary and sufficient for a training set and which number of middle layer nodes and connections are required to represent that knowledge. The training set must be statistically significant to the problem to assure the neural network learns exactly what the researcher wishes it to learn. The size of the training set necessary to achieve this mapping while still leaving the network free to generalize has not been theoretically established. The research did not explore the optimal training set required for this problem, either in terms of a minimal number of training records or in terms of a training set that best represented the expected ranges of setpoints and gamma heat variations.

TABLE 6. SUMMARY OF INPUT AND OUTPUT SIGNALS FOR SERIES 4, 5 AND 6

<u>Input Signal</u>	<u>Series 4</u>	<u>Series 5</u>	<u>Series 6</u>
FE-6	X	X	X
FE-6 delta	X	X	X
TE-37-8-1		X	X
TE-37-8-1 delta		X	X
TE-31	X	X	X
TE-31 delta	X	X	X
TCV-31		X	X
TE-31 setpoint	X	X	X
TE-42		X	X
TE-42 delta		X	X
TE-37-6-2	X	X	X
TE-37-6-2 delta	X	X	X
FE-2	X	X	X
FE-2 delta	X	X	X
FCV-1		X	X
FE-2 setpoint	X	X	X
Gamma heat	X	X	X
TE-31 / TE-42		X	X
TE-31 average			X
TE-42 average			X
<u>Output Signal</u>			
FCV-1 request delta	X		X
FCV-1 request		X	
TCV-31 request	X	X	X

The training set used in this project differed from the benchmark sequence by which the network performances were measured. This comparison is summarized in Table 7. The ranges of setpoints and gamma heat variations and the increments of change for both the training set and the benchmark sequence are tabulated. This comparison shows that the networks performed best in areas of the state space most adequately represented by the training set. The drop from full reactor power to zero power, a change from 50 BTU/s to 0 BTU/s, was not found in the training set. This portion of the benchmark sequence representing a reactor scram from full power provided the greatest challenge to the neural network controllers. Similarly, the region of the benchmark sequence where the flow setpoint was 65 gpm, below the lowest value of 66 gpm in the training set, and the gamma heat variations were 1 and 2 BTU/s, smaller than the 5 BTU/s variations in the training set, was confusing to the neural network controllers. With long training, the temperature offset was more than 1°F in this region.

TABLE 7. COMPARISON OF TRAINING SET AND BENCHMARK SEQUENCE

<u>Parameter</u>	<u>Training Set</u>		<u>Benchmark</u>	
	<u>Range</u>	<u>Increments</u>	<u>Range</u>	<u>Increments</u>
Flow setpoint	66-84 gpm	2-12 gpm	65-82 gpm	4-17 gpm
Temperature setpoint	555-585°F	5 and 10°F	555-580°F	20 and 25°F
Gamma heat	0-52 BTU/s	5-15 BTU/s	0-51 BTU/s	1-50 BTU/s

6.0 CONCLUSIONS

The main conclusion to be drawn from this project is that a neural network can learn to generate appropriate control signals for a nonlinear, multivariate system. This conclusion is substantiated by the performance of the traditionally designed and trained neural networks developed during the course of the research. The neural network controllers were quite accurate at temperature and flow setpoint control of the simulated ATR experiment loop under the demanding conditions imposed by the benchmark sequence of setpoint changes and gamma heat variations.

A further conclusion is that the neural network controller performance can be improved through use of a richer set of input system parameters and their mathematical combinations, through an optimal set of middle layer nodes, and through the design of a more relevant training set for use in the error backpropagation learning rule. In this study, the neural network controller with the richest set of input variables had the most accurate control performance, and no indication was observed that this input set was the best that could be used. This study made no attempt to optimize the number of middle layer nodes and the number of node connections, so a more efficient network could probably be designed for the problem. Research is just being published on the role of the training set in network training, and a continued review of the literature and experimentation with training sets would certainly result in improved generalization and control output accuracy of the neural network.

The final conclusion is that better neural network controllers can be developed when the characteristics of the system, its operation, and the classical control algorithms are understood and incorporated in the neural network design. As the research progressed, the selection of input parameters

was varied to better represent the features of the operating experiment loop for the neural network. The input parameters were also extended to include those characteristics of the system which were important to classical controllers. The performance of the neural networks improved as the set of input parameters was refined based on this knowledge of the loop and the existing controllers. This knowledge was qualitative and empirical in nature, not an exhaustive mathematical representation of loop thermohydraulics or feedback and feedforward control algorithms.

The research did not result in a more accurate controller for the ATR experiment loop but it did prove that a neural network controller can be developed within existing technology. This success was necessary before work could be recommended for developing an adaptive neural network controller that would maintain temperature and flow setpoint control despite changes in the loop characteristics.

7.0 REFERENCES

1. G. A. Cordes, "FY 1989 ERDP No. 2702 Process Control Using Neural Computation," EG&G Idaho, Inc., letter report to C. R. Toole, GAC-31-89, October 9, 1989.
2. D. C. Mumme and D. R. Chick, "Design of a Neural-Network Control System," EG&G Idaho, Inc., Informal Report EGG-EE-8256, September 1988.
3. N. E. Pace, "ATR Loops Specs Revision; Operational Information Transmittal," EG&G Idaho, Inc., letter report to distribution, NEP-19-89, June 20, 1989.
4. D. E. Rumelhart and J. L. McClelland et al, Parallel Distributed Processing, Volumes 1 and 2, MIT Press, 1986.
5. C. L. Bowen, "Documentation of an ATR Experiment Loop Simulation," EG&G Idaho, Inc., Engineering Design File (EDF) report, December 1988 (unpublished).
6. T. Bishop, Dan Greenwood, K. Shew and F. Stevenson, "Space Shuttle Main Engine Fault Detection Using Neural Networks," Proceedings of the 1990 Second Joint Technology Workshop on Neural Networks and Fuzzy Logic, Houston TX, April 1990.
7. N. K. Perugini and W. E. Engeler, "Neural Network Learning Time: Effects of Network and Training Set Size," Proceedings of the IJCNN International Joint Conference on Neural Networks, Volume II, Washington D.C., June 1989, pp. 395-401.

8. R. Hecht-Nielsen, "Theory of the Backpropagation Neural Network," Proceedings of the IJCNN International Joint Conference on Neural Networks, Volume I, Washington D.C., June 1989, pp. 593-605.
9. M. Wann, T. Hediger and N. N. Greenbaun, "The Influence of Training Sets on Generalization in Feed-Forward Neural Networks," Proceedings of the IJCNN International Joint Conference on Neural Networks, Volume III, San Diego CA, June 1990, pp. 137-142.

END

DATE FILMED

12 / 03 / 90