Improved Nuclear Reactor Temperature Control Using Diagonal Recurrent Neural Networks

Chao-Chee Ku, Kwang Y. Lee, and R. M. Edwards

Abstract—A new approach for wide-range optimal reactor temperature control using diagonal recurrent neural networks (DRNN) with adaptive learning rate scheme is presented. The drawback of the usual feedforward neural network (FNN) is that it is a static mapping and requires a large number of neurons and takes a long training time. The usual fixed learning rate based on empirical trial and error scheme is slow and does not guarantee convergence. The DRNN is for dynamic mapping and requires much fewer number of neurons and weights, and thus converges faster than FNN. A dynamic backpropagation algorithm coupled with adaptive learning rate guarantees even faster convergence. The DRNN controller described here includes both a neurocontroller and a neuroidentifier. A reference model which incorporates an optimal control law with improved reactor temperature response is used for training of the neurocontroller and neuroidentifier. Rapid convergence of this DRNN-based control system is demonstrated when applied to improve reactor temperature performance.

I. INTRODUCTION

A n observer-based optimal state feedback control theory has been developed to improve the temperature response of a nuclear reactor [1]–[4]. A controller was designed in a robust manner to account for the uncertainties introduced by linearization, unmodeled dynamics, and significant parameter variations. As an alternative to the model-based controller design, this paper considers artificial neural networks, which not only identify nonlinear plant dynamics under various operating conditions, but also control the plant to give an improved temperature response over a wide range of operation.

Artificial neural networks have been used in many control problems of dynamical systems, and some applications in nuclear power plant control are reported [5]–[8]. In [7], an application of neural network for power plant sensor validation is studied. Their results show that the learning and interpolation abilities of a neural network are extremely good and very promising. While in [8], a power prediction system was developed using an artificial neural network. From their results, the neural network approach was shown that it can precisely predict the thermal power in a nuclear power plant. Most people use the feedforward neural network (FNN), combined with tapped delays, and the error backpropagation training algorithm to deal with the dynamic problems. However, the feedforward network is a static mapping and requires a large number of neurons in representing a dynamic response in the time domain. On the other hand, the recurrent neural network (RNN) is more suitable for dynamic systems than the feedforward network. The fully connected recurrent neural network (FRNN), however, where all neurons are coupled to one another, is difficult to train and to converge in a short time; thus it is not a good candidate for our neurocontroller and neuroidentifier.

Recently, the architecture of diagonal recurrent neural network (DRNN) was developed as a minimal realization of RNN [6], [9]. It is minimal in the sense that it has only one hidden layer of self-recurrent neurons, each feeding back its output only into itself and not to other neurons in the same layer. Since there are no interlinks among neurons, the DRNN has considerably fewer weights than the FRNN. Therefore, training is much faster for DRNN and it can be implemented easily. The comparisons of the DRNN, FNN, and FRNN in term of their mapping characteristics and the requirement of number of neurons and weights can be found in [10].

We demonstrate the robustness and adaptivity of the DRNN in controlling reactor temperature and compare with desired optimal reference responses. The DRNN-based controller is shown to converge very rapidly due to an adaptive dynamic learning algorithm [10]. Simulation results show that it performs very well not only locally for normal operation, but also over a wide range of operation.

This paper is organized as following. Section II presents the DRNN-based control system and the dynamic learning algorithm. Section III gives the concept of adaptive learning rate to enhance convergence property. The DRNN-based control system is applied to a fifth order nuclear reactor model to follow reference models in Section IV. The simulation results for a number of case
studies are discussed in Section V, and conclusions are drawn in Section VI.

II. DIAGONAL RECURRENT NEURAL NETWORK BASED CONTROL

The structure of the diagonal recurrent neural network (DRNN) is shown in Fig. 1 for both control and system identification [9]. The controller network, called a diagonal recurrent neurocontroller (DRNC), uses three layers: input, output, and hidden layers. The hidden layer has a number of self-recurrent neurons, each feeding back its output only into itself and not to other neurons in the same layer. Fig. 2 shows the DRNN based control system. The input to the DRNC consists of reference signal, bias, delayed outputs of the DRNC and the plant. The output of the DRNC is the input signal to the plant. The identifier network, called a diagonal recurrent neuroidentifier (DRNI), also has only one hidden layer. The input to the DRNI consists of the control signal generated from DRNC and the delayed output of the plant.

A. Dynamic Representation of DRNN

The mathematical model for the DRNN in Fig. 1 is shown below:

\[ O(k) = \sum_j W^O_j X_j(k), \quad X_j(k) = f(S_j(k)), \]  

\[ S_j(k) = W^P_j X_j(k-1) + \sum_i W^I_{ij} I_i(k), \]  

where \( I_i(k) \) is the \( i \)-th input to the DRNN, \( S_j(k) \) is the sum of inputs to the \( j \)-th recurrent neuron, \( X_j(k) \) is the output of the \( j \)-th recurrent neuron and \( O(k) \) is the output of the DRNN. Here \( f(\cdot) \) is the usual sigmoid function representing nonlinear threshold function, and \( W^I, W^P, \) and \( W^O \) are input, recurrent, and output weight vectors, respectively, in \( \Re^n, \Re^m, \) and \( \Re^n, \) which are Euclidean spaces with appropriate dimensions.

Let \( r(k) \) and \( y(k) \) be the desired and actual responses of the plant, respectively, then an error function for DRNC can be defined as

\[ E_c = \frac{1}{2}(r(k) - y(k))^2. \]  

(3)

In general, the plant response is a nonlinear mapping \( G(\cdot) \) of input \( u(k), \) i.e., \( y(k) = G(u(i), i \leq k). \) Here, the plant input \( u(k) \) is the output of the DRNC, i.e., \( u(k) = O(k) \) in (1). On the other hand, in the case of the DRNI, the plant input \( u(k) \) is the same as the input to the DRNI.

The error function (3) is also modified for the DRNI by replacing \( r(k) \) and \( y(k) \) with \( y(k) \) and \( y_m(k), \) respectively, where \( y_m(k) \) is the output of the DRNI, i.e.,

\[ E_m = \frac{1}{2}(y(k) - y_m(k))^2, \]  

(4)

where \( y_m(k) = O(k) \) in (1).

The gradient of error in (3) with respect to an arbitrary weight vector \( W \in \Re^n \) is represented by

\[ \frac{\partial E_c}{\partial W} = -e_c(k) \frac{\partial y(k)}{\partial W} = -e_c(k)y_c(k) \frac{\partial u(k)}{\partial W} \]

\[ = -e_c(k)y_c(k) \frac{\partial O(k)}{\partial W}, \]  

(5)

where \( e_c(k) = r(k) - y(k) \) is the error between the desired and output responses of the plant, and the factor \( y_c(k) = \frac{\partial y(k)}{\partial u(k)} \) represents the sensitivity of the plant with respect to its input. Since the plant is normally unknown, the sensitivity needs to be estimated for the DRNC. However, in the case of the DRNI, the gradient of error in (4) simply becomes

\[ \frac{\partial E_m}{\partial W} = -e_m(k) \frac{\partial y_m(k)}{\partial W} = -e_m(k) \frac{\partial O(k)}{\partial W}, \]  

(6)

where \( e_m(k) = y(k) - y_m(k) \) is the error between the plant and the DRNI responses.

The output gradient \( \frac{\partial O(k)}{\partial W} \) is common in (5) and (6), and needs to be computed for both DRNC and DRNI. The gradient with respect to output, recurrent, and input weights, respectively, are computed using the following [9]

\[ \frac{\partial O(k)}{\partial W^O_j} = X_j(k) \]  

(7a)

\[ \frac{\partial O(k)}{\partial W^P_j} = W^P_i P_i(k) \]  

(7b)

\[ \frac{\partial O(k)}{\partial W^I} = W^O_i Q_i(k), \]  

(7c)

where

\[ P_i(k) = \frac{\partial X_i(k)}{\partial W^O_i} \]
and

\[ Q_{ij} = \frac{\partial X_i(k)}{\partial W_{ij}} , \]

and satisfy

\[ P_j(k) = f^r(S_j)(X_j(k - 1) + W_j^D P_j(k - 1)), P_j(0) = 0 \]

(8a)

\[ Q_{ij}(k) = f^r(S_j)(I_j(k) + W_j^D Q_{ij}(k - 1)), Q_{ij}(0) = 0 \]

(8b)

Note (8a) and (8b) are nonlinear dynamic recursive equations for the state gradients \( \partial X_i(k)/\partial W \), and can be solved recursively with given initial conditions. For the usual FNN, the recurrent weight \( W_j^D \) is zero and the equations become algebraic.

**B. Dynamic Backpropagation for DRNI**

From (6), the negative gradient of the error with respect to a weight vector in \( \mathbb{R}^n \) is

\[ -\frac{\partial E_m}{\partial W} = e_m(k) \frac{\partial O(k)}{\partial W} , \]

(9)

where the output gradient is given by (7) and (8), and \( W \) represents \( W^D \), \( W^I \), or \( W^I \) in \( \mathbb{R}^m \), \( \mathbb{R}^n \), or \( \mathbb{R}^n \), respectively.

The weights can now be adjusted following any gradient method such as the steepest descent method, i.e., the update rule of the weights becomes

\[ W(n + 1) = W(n) + \eta \left( -\frac{\partial E_m}{\partial W} \right) + \alpha \Delta W(n) , \]

(10)

where \( \eta \) is a learning rate, \( \alpha \) is a momentum factor, and \( \Delta W(n) \) represents the change in weight in the \( n \)th iteration. The equations (7)–(10) define the dynamic backpropagation algorithm (DBP) for DRNI.

**C. Dynamic Backpropagation for DRNC**

In the case of DRNC, from (5), the negative gradient of the error with respect to a weight vector in \( \mathbb{R}^n \) is

\[ -\frac{\partial E_c}{\partial W} = e_c(k) \frac{\partial O(k)}{\partial W} . \]

(11)

Since the plant is normally unknown, the sensitivity term \( y_c(k) \) is unknown. This unknown value can be identified by using the DRNI. When the DRNI is trained, the dynamic behavior of the DRNI is close to the unknown plant, i.e., \( y(k) = y_m(k) \), where \( y_m(k) \) is the output of the DRNI.

Therefore, the sensitivity was approximated in [9] and shown to be

\[ y_m(k) \approx \frac{\partial y(k)}{\partial u(k)} \approx \frac{\partial y_m(k)}{\partial u(k)} \approx \sum_j W_j^D f^r(S_j(k))W_{ij}^I , \]

(12)

where the variables and weights are those found in DRNI, and \( W_{ij}^I \) represents the input weight of DRNI corresponding to \( u(k) \) as its input. Using the negative gradients in (11), the weights for DRNC can now be adjusted using the update rule similar to (10). The equations (7), (8), (10)–(12) define the dynamic backpropagation algorithm for DRNC.

**III. ADAPTIVE LEARNING RATE FOR TRAINING DRNN**

The speed of convergence depends upon the learning rate \( \eta \) in the dynamic backpropagation algorithm. However, an arbitrary large learning rate causes the algorithm to be unstable. An efficient way of checking for the largest possible learning rate which guarantees the convergence stability was developed [10] and its concept is summarized here.

A discrete-type Lyapunov function can be given by

\[ V(k) = \frac{1}{2} e^2(k) , \]

(13)

where \( e(k) \) represents the error in the learning process.
Thus, the change of the Lyapunov function due to the training process is obtained by
\[
\Delta V(k) = V(k+1) - V(k) = \frac{1}{2} [e^T(k+1) - e^T(k)].
\] (14)

The error difference due to the learning can be represented by
\[
e(k+1) = e(k) + \Delta e(k) = e(k) + \left[ \frac{\partial e(k)}{\partial W} \right]^T \Delta W,
\] (15)
where \( \Delta W \) represents a change in an arbitrary weight vector in \( \mathbb{R}^n \).

A. Convergence of DRNI

From the update rule of (6) and (10) with \( \alpha = 0 \), and since \( e_w(k) = y(k) - y_n(k) \), thus
\[
\frac{\partial e_w(k)}{\partial W_i} = -\eta e_w(k) \frac{\partial \Delta e_w(k)}{\partial W_i} = -\eta e_w(k) \frac{\partial O(k)}{\partial W_i},
\] (16)
where \( W_i \) and \( \eta \) respectively represent an arbitrary weight and the corresponding learning rate in DRNI, and \( O(k) \) is the output of DRNI. Then we have the following general convergence theorem [10]:

**Theorem 1**: Let \( \eta \), the learning rate for the weights of DRNI, satisfy \( \eta = \eta_l / g_{l,\text{max}} \) with \( 0 < \eta < 2 \), and \( g_{l,\text{max}} \) defined as \( g_{l,\text{max}} = \max_k \|g_i(k)\| \), where \( g_i(k) = \partial O(k)/\partial W_i \), and \( \| \cdot \| \) is the usual Euclidean norm in \( \mathbb{R}^n \). Then the convergence is guaranteed if \( \eta \) is chosen as
\[
0 < \eta < \eta_l / g_{l,\text{max}}.
\] (17)

B. Convergence of DRNC

From the update rule of (10) and (11) with \( \alpha = 0 \), and following the same procedure as in Section A,
\[
\Delta W_c = -\eta e_c(k) \frac{\partial e_c(k)}{\partial W_c} = -\eta e_c(k) y_n(k) \frac{\partial u(k)}{\partial W_c} = -\eta e_c(k) y_n(k) \frac{\partial O(k)}{\partial W_c},
\]
where \( \partial u(k)/\partial u(k) = y_n(k) \) is the plant sensitivity, \( W_c \) and \( \eta \), respectively, represent an arbitrary weight and the corresponding learning rate in DRNC, and \( O(k) \) is the output of DRNC. Then we have the following general convergence theorem [10]:

**Theorem 2**: Let \( \eta \), the learning rate for the weights of DRNC, satisfy \( \eta = \eta_e / g_{e,\text{max}} \) \( S_{e,\text{max}} \) with \( 0 < \eta < 2 \), \( g_{e,\text{max}} \) defined as \( g_{e,\text{max}} = \max_k \|g_e(k)\| \), where \( g_e(k) = \partial O(k)/\partial W_c \), and \( S_{e,\text{max}} = h_l W_l^{1/2} / \| W_l \|^{1/2} \), where \( h_l \) is the number of neurons in the hidden layer. Then the convergence is guaranteed if \( \eta \) is chosen as
\[
0 < \eta < \frac{2}{S_{e,\text{max}}^2 g_{e,\text{max}}^2}.
\] (18)

Since (17) and (18) define the upper bound on the learning rate for an arbitrary weight vector, they can be used individually to compute the largest possible learning rate for input, hidden, or output weight vector [10].

IV. Application to Reactor Control

The proposed DRNN based control system is applied to a nuclear reactor model to improve reactor temperature response. As shown in Fig. 2, the inputs to the neurocontroller DRNC are the desired power level \( n' \), the control rod reactivity worth gain \( G_r \), the delayed control signal \( u(k-1) \) and delayed output of the plant \( y(k-1) \). The desired power load \( n' \) is split into two parts \( n' = n'_l + \delta n_r \). The term \( n'_l \) represents an equilibrium power level in region \( i \) for which a time varying linear reference model is defined; \( \delta n_r \) represents a deviation from \( n'_l \). Inputs to the neuroidentifier DRNI are the control signal generated from DRNC \( u(k) \) and the delayed output of the plant \( y(k-1) \). The time varying linear reference model is approximated by a set of nine time invariant models, each representing different operating point \( (G_r, n'_l) \) for region \( i \), and the DRNN was trained with the reference model in each region.

A. The Reactor Power Plant Modeling

To demonstrate the proposed DRNN based controller, a simplified model of a pressurized water reactor (PWR) was used [11–4]. The model represents the point kinetics with one delayed neutron group and temperature feedback from lumped fuel and coolant temperature calculations. While this model is ideal for an initial comparison of control concepts on an equal basis, more detailed models, analysis, and experiments are required for verification of a system to be implemented on an operating power plant. Notations in the following model are fully explained in Nomenclature.

The point-kinetic equations with one delayed neutron group are
\[
\frac{dn}{dt} = \frac{\delta p - \beta}{\Lambda} n + \lambda c \tag{19}
\]
\[
\frac{dc}{dt} = \frac{\beta}{\Lambda} n - \lambda c, \tag{20}
\]
where \( n \) and \( c \) are neutron and precursor densities, respectively.

For computational purpose, it is convenient to use an equivalent normalized version of (19) and (20):
\[
\frac{dn}{dt} = \frac{\delta p - \beta}{\Lambda} n + \beta c, \tag{21}
\]
\[
\frac{dc}{dt} = \lambda n - \lambda c. \tag{22}
\]
Reactor temperatures change as a function of power generation and heat transfer from (or to) the system. Using the normalized point-kinetics equations for \( n_r \), reactor power \( P_e \) can be represented as

\[
P_e(t) = P_{ao} n_r(t).
\]

The following thermal-hydraulic model is based on early work of Schultz [2] and represents a two-temperature feedback mechanism for a PWR. First, the time-dependent heat transfer rate from fuel to coolant \( P_c \) and the net heat removal rate from the coolant \( P_e \) are

\[
P_e(t) = \Omega (T_f - T_c),
\]

\[
P_c(t) = M (T_i - T_c),
\]

where \( T_f \) and \( T_c \) are average reactor fuel and coolant temperatures, respectively, and \( T_i \) and \( T_c \) are respective temperatures for coolant leaving and entering the reactor. The differential equation formulations for the lumped fuel and coolant temperature are then

\[
f_f P_e(t) = \frac{d}{dt} T_f + P_e(t)
\]

\[
(1 - f_f) P_e(t) + P_e(t) = \mu_c \frac{d}{dt} T_c + P_e(t).
\]

Reactivity \( \delta \rho \) has several components, and control rod position and temperature feedback effects are represented by

\[
\delta \rho = \delta \rho_r + \delta \rho_v = \alpha_f (T_f - T_{f0}) + \alpha_c (T_c - T_{c0})
\]

\[
\frac{d}{dt} \delta \rho = G_r z_r,
\]

where \( G_r \) represents the control rod worth per unit length. This control rod gain is a time-varying parameter depending on power level, burnup, and physical location of the rod in the reactor core.

The fifth order reactor model is nonlinear because reactivity \( \delta \rho \) multiplies the relative reactor power state variable \( n_r \), as seen in (21). Reactivity includes the control rod reactivity state \( [\delta \rho_r] \) and feedback from the reactor temperatures states through (27).

If the nonlinear plant is linearized using a perturbation theory, it is valid only for \( \delta n_r \ll n_{r0} \), where \( n_{r0} \) is an equilibrium relative neutron density, and \( \delta n_r \) is a deviation from the equilibrium. Therefore, in this paper the nonlinear neutronic model described in (21)–(28) is directly used to avoid the problem resulting from the linearization.

### B. Linear Reference Models

The goal of this paper is to improve the temperature response of the nonlinear reactor model, and reactor safety operation dictates the specification of desired temperature responses. Although the DRNN based controller is capable of tracking any reasonably specified reference response, this paper chooses the model-based optimal controller [2]–[5] as a good candidate for generating reference training responses. The reference model is constituted by nine different time invariant linear reference models, as defined by the regions in Table I, and the optimal state feedback for each reference model was used to obtain desired temperature response. By using the \( \delta \) symbol prefaced to a state variable \( (n_r, \alpha, T_f, T_i, \rho) \) to indicate a deviation about an equilibrium condition, the linearized version of (21), when \( \delta \rho \) times \( \delta n_r \) is negligible, is

\[
\frac{d}{dt} \delta n_r = -\frac{\beta}{\Lambda} \delta n_r + \frac{\beta}{\Lambda} \delta \alpha + \frac{\delta \rho}{\Lambda} n_{r0},
\]

where \( n_{r0} \) is an equilibrium relative neutron density. Linear versions of the remaining equations are similarly represented by using the \( \delta \) symbol. The state, control, and output vectors in the state space representation are therefore

\[
\dot{x} = Ax + Bu
\]

\[
y = Cx,
\]

where

\[
x = \begin{pmatrix} \delta n_r \\ \delta \alpha \\ \delta T_f \\ \delta T_i \\ \delta \rho \end{pmatrix}, \quad y = [\delta n_r], \quad u = [z_r].
\]

The corresponding \( A, B, \) and \( C \) matrices are therefore

\[
A = \begin{pmatrix}
-\frac{\beta}{\Lambda} & \frac{\beta}{\Lambda} & n_{r0} \alpha_f/\Lambda & n_{r0} \alpha_c/2\Lambda & n_{r0}/\Lambda \\
\frac{\lambda}{\Lambda} & -\lambda & 0 & 0 & 0 \\
-f_f P_{ao}/\mu_f & -\Omega/\mu_f & \Omega/2\mu_f & 0 & 0 \\
(1 - f_f) P_{ao}/\mu_c & 0 & \Omega/\mu_c & -(2M + \Omega)/2\mu_c & 0 \\
0 & 0 & 0 & 0 & 0
\end{pmatrix}, \quad B = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ G_f \end{pmatrix}, \quad C = [1, 0, 0, 0, 0].
\]
Table 1: The Classification of Operation Regions

<table>
<thead>
<tr>
<th>(G_e \setminus n_{r0} )</th>
<th>0.1</th>
<th>0.5</th>
<th>1.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0290</td>
<td>Region 3</td>
<td>Region 4</td>
<td>Region 5</td>
</tr>
<tr>
<td>0.0145</td>
<td>Region 2</td>
<td>Region 1</td>
<td>Region 6</td>
</tr>
<tr>
<td>0.0070</td>
<td>Region 9</td>
<td>Region 8</td>
<td>Region 7</td>
</tr>
</tbody>
</table>

where \(n_{r0}'\) is the power level and \(G_e^i\) is the control rod worth for region \(i\).

The linear model (31) is defined at each operating point \((n_{r0}, G_e)\) and the following parameters in the system matrices are also functions of the operating points [12]:

\[
\alpha_f(n_{r0}) = (n_{r0} - 4.24) \times 10^{-3} \frac{\delta k}{k} / ^\circ C
\]

\[
\alpha_c(n_{r0}) = (-4.0n_{r0} - 17.3) \times 10^{-1} \frac{\delta k}{k} / ^\circ C
\]

\[
\mu_c(n_{r0}) = \left( \frac{160}{9} n_{r0} + 54.022 \right) MW/ ^\circ C
\]

\[
\Omega(n_{r0}) = \left( \frac{5}{3} n_{r0} + 4.9333 \right) MW/ ^\circ C
\]

\[
M(n_{r0}) = (28.0n_{r0} + 74.0) MW/ ^\circ C
\]

The remaining constant parameters in the system matrices are shown in Table II.

As mentioned above, the reference training response is generated by the model-based optimal controller. An optimal state feedback control was designed [2]–[3] as \(u = Fx\), where the optimal feedback gain matrix is \(F = [f_1, f_2, f_3, f_4, f_5]\) and \(v\) is a normalization factor.

The optimal feedback gains \((F_1, v)\) were computed for the nine different regions defined in Table I using a region dependent \(J(n_{r0})\) performance index. The resulting gains are tabulated in Table III and IV. Using the optimal feedback gains for each region the reference model was implemented to generate a desired response for training the DRNN in the regions.

V. SIMULATION RESULTS

The number of inputs to DRNC and DRNI are denoted by \(n_c\) and \(n_f\), respectively, and \(h_c\) and \(h_f\) denote the number of neurons in the hidden layer for DRNC and DRNI, respectively. The number of neurons in the hidden layer is normally determined by trial and error. However, the minimum number for the usual FNN is given in [11], which is also chosen here for DRNN as \(h_c = 2n_c + 1\) or \(h_f = 2n_f + 1\). While the neurons in the hidden layer for DRNC and DRNI are all nonlinear threshold neurons, the neurons in the output layer are linear neurons, which then accommodate the output values in a wide range. Thus, it can be seen that the total number of neurons and weights for the DRNN-based control system are \(N_T = h_c + h_f + 2\), and \(W_T = (n_c + 3)h_c + (n_f + 3)h_f\), respectively.

In the following studies, the set of inputs are \(P_c = (G_e', n_{r0}', u(k-1), y(k-1))\) and \(P_I = (u(k), y(k-1))\), thus \(n_c = 4, n_f = 2, N_T = 16,\) and \(W_T = 88\). Here, \(u(k)\) and \(y(k)\) are the input and output of the plant. First, all nine regions are trained sequentially (from region 1 to region 9) 2 times to track their respective reference models (global training), each with \(\pm 10\%\) changes in power level. Then it is followed by training only the region 6, four times (local training), which is the normal 100% power operating region with the average control rod worth

Table II: The Constant Parameters in the System Matrices

<table>
<thead>
<tr>
<th>(\lambda)</th>
<th>0.15 s(^{-1})</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\beta)</td>
<td>0.000019</td>
</tr>
<tr>
<td>(f_1)</td>
<td>0.92</td>
</tr>
<tr>
<td>(\lambda)</td>
<td>0.0001</td>
</tr>
<tr>
<td>(f_2)</td>
<td>26.3 MW/(^\circ C)</td>
</tr>
<tr>
<td>(P_{eq})</td>
<td>2500.0 MW</td>
</tr>
</tbody>
</table>

Table III: The Normalization Factors

<table>
<thead>
<tr>
<th>(n_{r0})</th>
<th>0.1</th>
<th>0.5</th>
<th>1.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>(v)</td>
<td>1.03044</td>
<td>0.61210</td>
<td>1.16680</td>
</tr>
</tbody>
</table>

Table IV: The Optimal Feedback Gains

<table>
<thead>
<tr>
<th>(n_{r0})</th>
<th>(G_e)</th>
<th>(f_1)</th>
<th>(f_2)</th>
<th>(f_3)</th>
<th>(f_4)</th>
<th>(f_5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>0.0070</td>
<td>-0.00146</td>
<td>-0.50896</td>
<td>-0.00013</td>
<td>-0.00015</td>
<td>-20.4681</td>
</tr>
<tr>
<td>0.1</td>
<td>0.0145</td>
<td>-0.00178</td>
<td>-0.50988</td>
<td>-0.00034</td>
<td>-0.00036</td>
<td>-15.6953</td>
</tr>
<tr>
<td>0.1</td>
<td>0.0290</td>
<td>-0.00205</td>
<td>-0.46649</td>
<td>-0.00058</td>
<td>-0.00074</td>
<td>-11.9126</td>
</tr>
<tr>
<td>0.5</td>
<td>0.0070</td>
<td>-0.00042</td>
<td>-0.14689</td>
<td>-0.00038</td>
<td>-0.00002</td>
<td>-24.6451</td>
</tr>
<tr>
<td>0.5</td>
<td>0.0145</td>
<td>-0.00064</td>
<td>-0.16159</td>
<td>-0.00015</td>
<td>-0.00002</td>
<td>-20.9979</td>
</tr>
<tr>
<td>0.5</td>
<td>0.0290</td>
<td>-0.00087</td>
<td>-0.15632</td>
<td>-0.00032</td>
<td>-0.00031</td>
<td>-17.3267</td>
</tr>
<tr>
<td>1.0</td>
<td>0.0070</td>
<td>-0.00112</td>
<td>-0.24069</td>
<td>-0.00037</td>
<td>0.00076</td>
<td>-56.5567</td>
</tr>
<tr>
<td>1.0</td>
<td>0.0145</td>
<td>-0.00165</td>
<td>-0.23305</td>
<td>-0.00075</td>
<td>0.00096</td>
<td>-47.7479</td>
</tr>
<tr>
<td>1.0</td>
<td>0.0290</td>
<td>-0.00221</td>
<td>-0.20579</td>
<td>-0.00118</td>
<td>0.00105</td>
<td>-39.0032</td>
</tr>
</tbody>
</table>
(Gc = 0.0145). In the training process, the reference model for each region is used to generate the desired tracking response, and the error between the desired reference the response and plant output is used to adjust the weights of the neurocontroller. Due to the interpolation abilities, the neural network can perform as a universal controller which can drive the nonlinear plant to exhibit the response of any of the nine desired reference models for an arbitrary untrained input and acts like a gain scheduling controller.

Once the training procedure is completed, the following four groups of cases are tested for verification and validation of the DRNN-based controller:

Case A: Local control
1. 100% → 90% → 100% power level changes in region 6

Case B: Global operation
1. 100% → 90% → 100% power level changes in region 7
2. 100% → 90% → 100% power level changes in region 5.
3. 40% → 50% → 40% power level changes in region 1.
4. 20% → 10% → 20% power level changes in region 9.
5. 20% → 10% → 20% power level changes in region 3.

Case C: Emergency operation
1. 100% → 25% huge step down from region 5 to region 3.

Case D: Shut-down/Start-up
1. 100% → 10% ramp down from region 5 to region 3, followed by 10% → 100% ramp up with 15% per minute rate.

A. Local Control

After training is completed (global training followed by a local training in region 6), the neural network is tested again in region 6 with the power level changing in steps: 100% → 90% → 100%. The responses of reactor power, exit temperature, and control rod speed are shown in Fig. 3. Since this is a local control, the performance of the neural network is very good as expected. Similar testing was performed for all nine regions for local control, where the network was trained locally in each region following the global training. The responses of the reference model and the plant matched very closely in each region.

B. Global Operation

In this study, the controller whose training was completed in Region 6 (normal operation) is tested in five other regions. To verify a global operation, the center and four extreme corners (Table I) are selected for testing: regions 1, 3, 5, 7, and 9. In all cases a step change of
Fig. 6. Case B3: Global operation in region 1 for 40% → 50% → 40% power level change. (a) Relative reactor power. (b) Exit temperature (unit: °C). (c) Control rod speed (unit: fraction of core length per second) (Solid line: reference. Dotted line: plant response).

Fig. 7. Case B4: Global operation in region 9 for 20% → 10% → 20% power level change. (a) Relative reactor power. (b) Exit temperature (unit: °C). (c) Control rod speed (unit: fraction of core length per second) (Solid line: reference. Dotted line: plant response).

Fig. 8. Case B5: Global operation in region 3 for 20% → 10% → 20% power level change. (a) Relative reactor power. (b) Exit temperature (unit: °C). (c) Control rod speed (unit: fraction of core length per second) (Solid line: reference. Dotted line: plant response).
±10% in power level was applied. In cases 1 and 2, since they are in high power region, as expected, the results for both cases are very good. The responses of reactor power, exit temperature, and control rod speed for both cases are shown in Figs. 4 and 5. When case 3, which is the mid-power region, is tested, the results for reactor power, exit temperature, and control rod speed are shown in Fig. 6. In this case, the results are still good. For cases 4 and 5, which are low power regions, the simulation results of reactor power, exit temperature, and control rod speed responses are shown in Figs. 7 and 8, respectively. In low power regions, the performances are fair compared to the reference model responses. The comparison of the exit temperatures from high power regions are shown in Fig. 9. As can be seen in these results, the regions with higher control rod worth \( G_c \) have faster temperature responses.

C. Emergency Operation

This case demonstrates the validity of the controller for most stressed operation. The power level demand is changed from 100% to 25%, from region 5 to region 3, where both regions have high control rod worth \( G_c = 0.029 \). The results of power and temperature responses are shown in Fig. 10.

D. Shut-down / Start-up

This case demonstrates the validity of the controller for shut-down/start-up operation with a relatively fast ramp, 15% per minute. The system was operating in region 5 (100% power, 0.029 control rod worth) and the input demand signal to the system is the fast ramp from 100% → 10% → 100% while maintaining the high control rod worth \( G_c = 0.029 \). The responses of the reactor power and temperature are shown in Fig. 11(a). The results of the power and control rod speed responses are shown in Fig. 11(b). The difference between reference model and plant output is not perceptible.

Remarks: In the training process, the global training is terminated when the average error \( e^*_g \) is less than \( e_1 \). Here \( e^*_g \) is defined as

\[
e^*_g = \frac{1}{2250} \sum_{i=1}^{9} \sum_{k=1}^{2250} |y^*_i(k) - y^i(k)|,
\]

where \( y^*_i(k) \) is the reference output of the \( k^{th} \) sampling point in region \( i \) and \( y^i(k) \) is the corresponding nonlinear plant output. Here 9 is the total number of regions and 2250 is the number of time steps for each response. The average error \( e_1 \) is chosen as 0.06 in this simulation. For global training, after 2 training iterations, the average error is less than the defined error \( e_1 \), thus the global training is terminated after only 2 training iterations. In the global training, the neurocontroller DRNC is tuned roughly to be valid for all nine regions. For the local training, an error \( e_2 \), which is much smaller than \( e_1 \), is used as a stopping criterion. When the local average error \( e_l \) is less than \( e_2 \), the local training is terminated. Here \( e_2 \) is chosen as 0.003, and since local training is based on region 6, thus \( e_l \) is defined as

\[
e_l = \frac{1}{2250} \sum_{k=1}^{2250} |y^*_6(k) - y^6(k)|,
\]

where \( y^*_6 \) and \( y^6(k) \) are the outputs of reference model and nonlinear plant in region 6, respectively. After only 4 training iterations in region 6, the error \( e_l \) is less than \( e_2 \), thus the training is terminated. In the local training, the neurocontroller DRNC is fine-tuned as a local controller.

VI. DISCUSSION AND CONCLUSION

This paper described the diagonal recurrent neural network (DRNN) based control system, which includes the diagonal recurrent neurocontroller (DRNC) and neuroidentifier (DRNI). The DRNN architecture was shown to have a very good dynamic mapping characteristic. Moreover, it requires much fewer neurons and weights.
the power demand. The response speed is not as quick as in the case of 10% change, but the performance is still good. In tracking the fast ramp command, the proposed DRNN based control system follows the reference model very closely.

Fuzzy logic has been used in many dynamic control problems recently. The advantage of the fuzzy logic approach is that it is simple in the sense that once the membership functions and rules are defined only the elementary arithmetic operations are needed in the computation of the desired control signal. However, it is not an easy job to find the membership functions and rules, and usually a trial-and-error method is used. One way to deal with this problem is to introduce fuzzy neural networks such that suitable membership functions and rules can be obtained by the help of neural networks [13]–[14].

**NOMENCLATURE**

\[ n \]  Neutron density (\( n/\text{cm}^3 \)).

\[ c \]  (neutron) precursor density (atom/\text{cm}^3).

\[ \lambda \]  Effective precursor radioactive decay constant (s\(^{-1}\)).

\[ \Lambda \]  Effective prompt neutron lifetime (s).

\[ \beta \]  Fraction of delayed fission neutrons.

\[ k = k_{\text{eff}} \]  Effective neutron multiplication factor.

\[ \delta p = \frac{k - 1}{k} \]  reactivity (since \( k \approx 1.0, \delta p \approx k - 1 \); at steady state \( k = 1, \delta p = 0 \)).

\[ n_0 \]  Equilibrium neutron density at rated power.

\[ c_0 \]  Equilibrium precursor density at rated power.

\[ n_r \]  \( n/n_0 \), neutron density relative to density at rated condition.

\[ c_r \]  \( c/c_0 \), precursor density relative to density at rated condition.

\[ P_e \]  Power transferred from fuel to coolant (\( MW \)).

\[ P_r \]  Power removed from the coolant (\( MW \)).

\[ P_{0a} \]  Rated power level (\( MW \)).

\[ P_a \]  Power generated in (\( MW \)).

\[ f_f \]  Fraction of reactor power deposited in fuel.

\[ \mu_f \]  Heat capacity of the fuel (\( MW.s./^\circ\text{C} \)).

\[ \mu_c \]  Heat capacity of the coolant (\( MW.s./^\circ\text{C} \)).

\[ \Omega \]  Heat transfer coefficient between fuel and coolant (\( MW/^\circ\text{C} \)).

\[ M \]  Mass flow rate times heat capacity of the water (\( MW/^\circ\text{C} \)).

\[ \delta p_r \]  Reactivity due to the control rod.

\[ z_r \]  Control input, control rod speed (fraction of core length per second).

\[ G_r \]  Reactivity worth of the rod per unit length (with rod speed in units of fraction of core length per second, \( G_r \) is the

when compared with the feedforward network. The above features allow the DRNN to be considered for possible on-line applications. Also, the use of adaptive learning rates guarantees convergence for any stable plant, in which case the learning rates are adjusted in each step so that they are within the corresponding bounds.

Several case studied, which include local and global control, huge change, and ramp tracking, have been investigated for verification and validation of the proposed DRNN-based controller. The results show that after the global training procedure for neural networks is completed, not only can the neurocontroller be used as an effective local controller but also as a global controller for an arbitrary untrained input. Although in low power regions, far from the last training region, the plant output does not track the reference model very well; however, the performance is acceptable. The proposed controller can also handle the case where a huge change is made in
total reactivity of the rod.

\( \alpha_f \)
Fuel temperature reactivity coefficient.

\( \alpha_c \)
Coolant temperature reactivity coefficient.

\( T_f \)
Average reactor fuel temperature (°C).

\( T_{f0} \)
Initial equilibrium (steady-state) fuel temperature.

\( T_{c0} \)
Initial equilibrium (steady-state) coolant temperature.

REFERENCES


