Lecture Series on

Intelligent Control

Lecture 13

Neural Networks in
Control System Applications

Kwang Y. Lee
Professor of Electrical & Computer Engineering
Baylor University
Waco, TX 76706, USA
Kwang_Y_Lee@baylor.edu

1

OVERVIEW

- **Introduction**
- **Background**
- **Neural Network Architectures for Modeling and Control**
- **Supervised Neural Network Structures**
- **Diagonal Recurrent Neural Network-Based Control System**
- **Convergence and Stability**
- **Nuclear Reactor Control**
- **Conclusion**

2

2

BACKGROUND

Widrow and Smith (1987):

ADAptive LINear Element (ADALINE): control of an inverted pendulum

Albus (197x):

Cerebella Model Articulation Controller (CMAC) - to control robotic manipulators

Modified ADALINE - Perceptron's architecture combined with ADALINE for binary encoding of the input space

3

3

Power System Control and Identification

Chow & Thomas (1989) - machine modeling
Santoso & Tan (1990) - capacitor control in distribution system
Weerasooriya & El-Sharkawi (1991) - identification and control of dc motor
Wu *et al.* (1992) - NN regulator for turbogenerator
Cho *et al.* (1992) - neuro-fuzzy controller for an induction machine
Hsu & Chen (1991), Saitoh *et al.* (1991a,b), Zhang *et al.* (1992,93,94) - PSS

4

4

Power System Control and Identification

Hiyama *et al.* (1995) - tracking controller for photovoltaic system
Beaufays & Widrow (1993) - load-frequency control
Neily *et al.* (1992) - joint var controller
Djukanovic *et al.* (1995) - coordinated stabilization control of exciter and governor
Park, Choi, & Lee (1994) - decentralized control for PSS
Ku, Lee, & Edwards (1992) - on-line control of nuclear reactor

5

5

Representation of Plants

Continuous-Time Representation:

$$\underline{\dot{x}}(t) = \underline{f}(\underline{x}(t), u(t))$$
$$y(t) = g(\underline{x}(t))$$

Discrete-Time Representation:

$$\underline{x}(t+1) = \underline{f}(\underline{x}(t), u(t))$$
$$y(t) = g(\underline{x}(t))$$

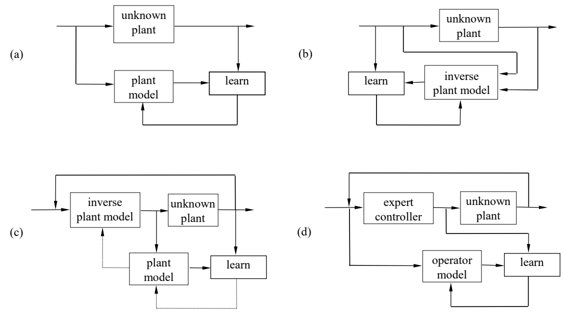
NARMA Representation:

$$y(t) = h(\underline{y}(t-1), \underline{u}(t))$$
$$\underline{y}(t-1) = [y(t-1), y(t-2), \ldots, y(t-n)]^T$$
$$\underline{u}(t) = [u(t), u(t-1), \ldots, u(t-m)]^T$$

6

6

Learning Modeling Architectures



Four learning modeling architectures: (a) forward plant modeling, (b) inverse plant modeling, (c) specialized inverse plant modeling, (d) operator modeling.

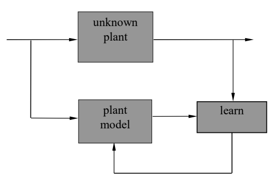
7

7

Learning Modeling Architectures

Forward Plant Modeling:

- **to predict the plant performance**
- **for fault diagnosis**



Minimize the Mean Square Output error:

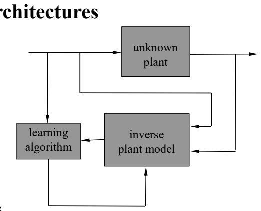
$$J_y = E(\varepsilon_y^2(t))$$
$$\varepsilon_y(t) = \hat{y}(t) - y(t)$$

8

8

Learning Modeling Architectures

Direct Inverse Plant Modeling:



- **the overall controller/plant architecture has a unity transfer function**
- **modeling error perturbs the transfer function away from unity**
- **combined with feedback control provides good performance**
- **minimizes the MSE in the control space rather than the output space**

$$J_u = E(\varepsilon_u^2(t))$$
$$\varepsilon_u(t) = \hat{u}(t) - u(t)$$

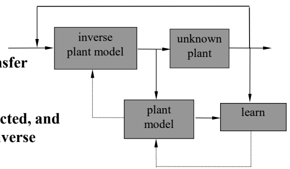
9

9

Learning Modeling Architectures

Specialized Inverse Plant Modeling:

- **inverse model/plant has a unity transfer function**
- **forward plant model is first constructed, and error is back propagated to tune the inverse model**
- **goal driven - plant error causes the inverse model to move into previously unexplored regions of input space**
- **not as robust as alternative learning controllers due to lack of feedback information**



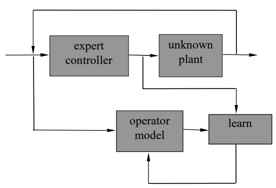
10

10

Learning Modeling architectures

Operator Modeling:

- **learning from an expert**
- **signal contains a large amount of noise due to the operator using different actions for similar inputs**
- **signal have to be filtered before learning algorithm can be applied**



11

11

Supervised Control Architectures

- **Fixed Stabilizing Controllers**
- **Predictive Learning Control Scheme**
- **Model Reference Adaptive Control**
- **Internal Model Control**

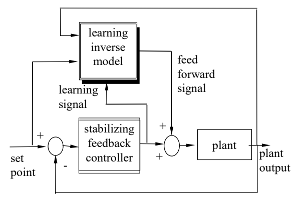
12

12

Supervised Control Architectures

Fixed Stabilizing Controllers:

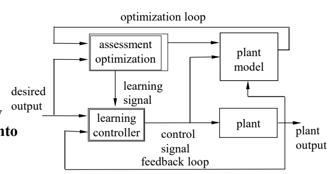
- **direct learning control scheme**
- **the closed-loop system is stable in every operating region**
- **the learning controller builds up a nonlinear model of the desired control surface**



Supervised Control Architectures

Predictive Learning Control Scheme:

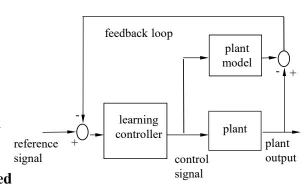
- **formulate a control strategy by assessing the affect of its action into the future and select the *optimal* control action**
- **learning control**
- **excellent closed-loop control for good plant model with proper performance function and search strategy**



Supervised Control Architectures

Internal Model Control:

- **model the process directly**
- **error between the model and the plant output is used as a feedback signal**
- **internal model controller is designed to be an inverse plant model**
- **stability results are available, with assumptions on the open-loop stability, exact modeling and/or inverse modeling**



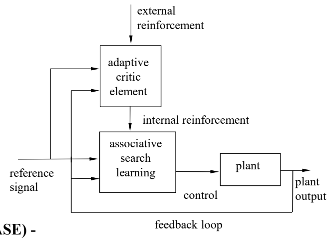
Reinforcement Learning Systems

Reinforcement Schemes:

- **learning with a critic**
- **minimally supervised learning algorithms**

Two adaptive elements:

- **Associative Search Element (ASE) - produces the optimal control signal**
- **Adaptive Critic Element (ACE) - provides a reinforcement signal**



16

16

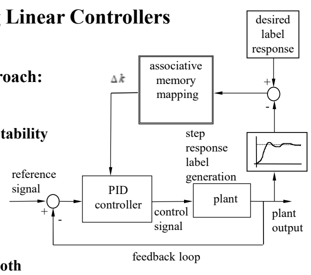
Parameterizing Linear Controllers

Intelligent Gain-Scheduling Approach:

- **linear feedback controllers**
- **known results on robustness and stability**
- **low implementation cost**

Neurocontrollers:

- **calculate control parameters for both off-line and on-line control**
- **adapt to time-varying processes**



17

17

SUPERVISED NEURAL NETWORK STRUCTURES

Multi-layer Feedforward Networks:

- **Input signal is propagated *forward* through several processing layers**
- **FNN is a *static mapping***
- **FNN with the aid of *tapped delays* represents *dynamic mapping***

Radial Basis Function Networks:

- **biological paradigm in favor of topology, simpler and more amenable to training**
- **a single layer of hidden nodes with *radially symmetric* basis activation function**
- **RBN is locally responsive**

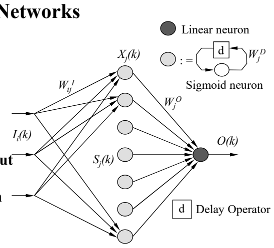
18

18

Recurrent Neural Networks

RNA:

- **FNN with some *feedbacks***
- **nonlinear *dynamic* network**
- **has attractor dynamics and store information for later use**
- **can deal with time-varying input/output**
- **dynamic mapping**
- **no or fewer external feedback through tapped delays**



Diagonal Recurrent Neural Networks:

- **minimal RNN**
- **no interlinks or cross talks**
- **fewer weights than FRNN**
- **real-time implementation**

19

19

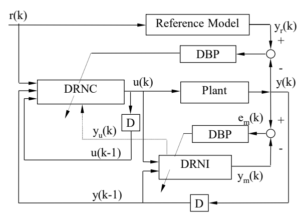
Architecture for DRNN-Based Control System

Neuroidentifier (DRNI):
- **models the plant and provide the sensitivity to DRNC**

Neurocontroller (DRNC):
- **controls the plant to follow a reference model**

Dynamic Backpropagation (DBP) Algorithm

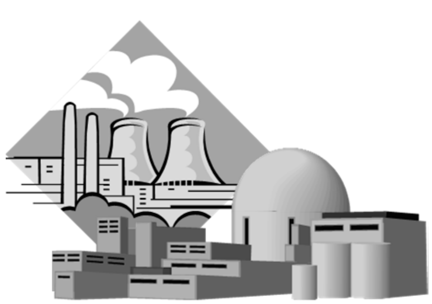
Adaptive Learning Rate: Convergence & Stability



20

20

Nuclear Reactor Control



21

21

NUCLEAR REACTOR CONTROL

The Reactor Power Plant Modeling

$$\frac{d}{dt}n = \frac{\delta\rho - \beta}{\Lambda}n + \lambda c \qquad \frac{d}{dt}n_r = \frac{\delta\rho - \beta}{\Lambda}n_r + \frac{\beta}{\Lambda}c_r$$

$$\frac{d}{dt}c = \frac{\beta}{\Lambda}n - \lambda c, \qquad \frac{d}{dt}c_r = \lambda n_r - \lambda c_r.$$

$$P_a(t) = P_{0a}n_r(t), \qquad P_c(t) = \Omega(T_f - T_c) \qquad P_e(t) = M(T_l - T_e),$$

$$f_f P_a(t) = \mu_f \frac{d}{dt}T_f + P_c(t) \qquad (1 - f_f)P_a(t) + P_c(t) = \mu_c \frac{d}{dt}T_l + P_e(t),$$

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

$$\frac{d}{dt}\delta\rho_r = G_r z_r.$$

22

22

Simulation Results

Operation Regions

G_r \ n_{r0}	0.1	0.5	1.0
0.0290	Region 3	Region 4	Region 5
0.0145	Region 2	Region 1	Region 6
0.0070	Region 9	Region 8	Region 7

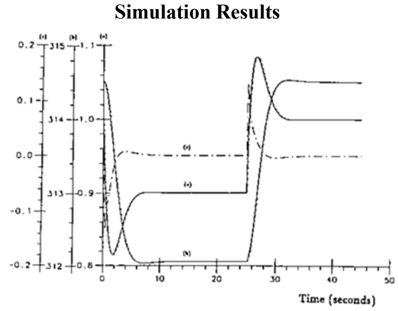
Test Case Studies:

Case A: Local control
100% → 90% → 100% power level changes in Region 6.
Case B: Global operation
40% → 50% → 40% power level changes in Region 1.
Case C: Emergency operation
100% → 25% huge step down from Region 5 to Region 3.
Case D: Shut-down/Start-up
100% → 10% → 100% ramp down and ramp up from Region 5 to Region 3.

23

23

Simulation Results

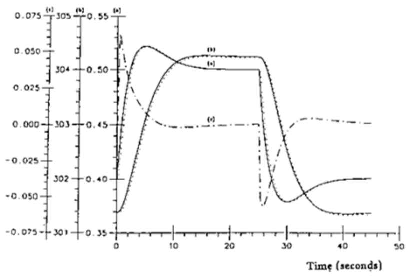


Case A: Local control in Region 6 for 100% → 90% → 100% power change. (a) Relative reactor power. (b) Exit temperature. (c) Control rod speed. (Solid line: reference. Dotted line: plant response).

24

24

Simulation Results

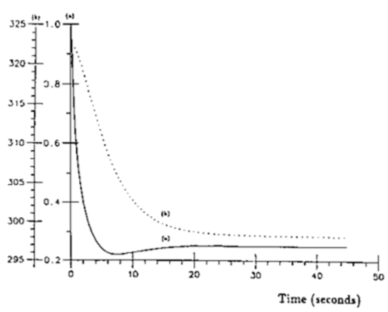


Case B: Global operation in Region 1 for 40% → 50% → 40% power change. (a) Relative reactor power. (b) Exit temperature. (c) Control rod speed. (Solid line: reference. Dotted line: plant response).

25

25

Simulation Results

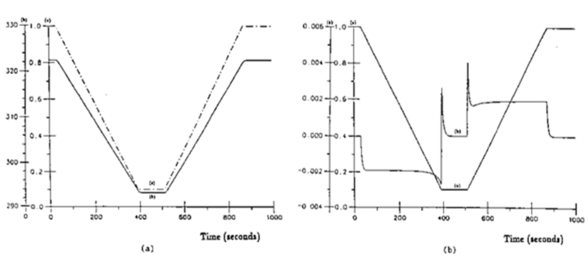


Case C: Emergency operation from Region 3 for 100% → 25% power change. (a) Relative power. (b) temperature response.

26

26

Simulation Results



Case D: Start-up/shut-down for 100% → 10% → 100% power change. (a) Relative reactor power (dot-dash line). (b) temperature (solid line) responses. (c) control rod speed responses.

27

27

CONCLUSIONS

- The use of neural networks for controlling dynamic systems
- Neural network architectures for modeling and control
- Neural network paradigms for neuromodeling and neurocontrol
- Diagonal recurrent neural networks
- DRNN-based control architecture for on-line implementation
- Nuclear reactor control

28

28