A Comparison of DHP Based Antecedent Parameter Tuning Strategies for Fuzzy Control

Alec M. Rogers, Thaddeus T. Shannon and George G. Lendaris
Northwest Computational Intelligence Laboratory
Portland, Oregon USA
alec@arborrhythms.com, tads@sysc.pdx.edu, lendaris@sysc.pdx.edu

Abstract

In the context of fuzzy control, antecedent parameters are used to provide a segmentation of the state space so that different regions can be modeled appropriately. In Adaptive Critic methodologies, two modules (the critic and the controller) must properly segment the state space to insure good performance. In this paper, we explore the effects of tuning antecedent parameters that are shared between these controller and critic networks (as opposed to tuning separate parameter sets). The results indicate that training shared antecedent parameters can be as effective as training separate antecedent parameters.

1. Introduction

Fuzzy logic allows an expert to create a controller that is well suited to a plant by using linguistic characterizations of the plant's state. This characterization defines which rules (or consequent parameters) should be used to control the plant. In some cases, however, no expert may be available who can adequately characterize the appropriate regions of the input space. In these cases, one might divide the input space into a great number of regions, or use an adaptive methodology to tune the region boundaries appropriately. This paper follows the latter approach.

In previous work within the Dual Heuristic Programming (DHP) paradigm [3][6], the same parameters are used as input to both controller and critic modules, each of which is allowed to independently alter its consequent parameters. This research [6] has shown that highly nonlinear plants can be controlled efficiently using fuzzy implementations of the controller and critic within the DHP methodology. Other research within the context of reinforcement learning indicates that tuning the antecedent parameters can yield better performance than using predefined antecedent parameters [1]. This paper addresses the issue of combining these approaches in a fruitful manner.

In this paper, we explore the implications of training antecedent parameters that are shared between the controller and the critic by using the following techniques: training antecedent parameters with the critic only, training with the controller only, training using both the critic and the controller, and not training the antecedent parameters. We also explore the possibility that antecedent parameters, even though they are partitioning the same state space, should be tuned separately by the controller and the critic.

The rationale behind these inquiries is that, for both the critic and controller, success depends on partitioning the state space so that nonlinearities of the plant can be modeled by local linear models. Given that the same state space must be characterized by both modules, it would seem that a common state space partitioning could be achieved more effectively with contributions from both the controller and the critic.

Intuitively, a good state space segmentation for estimating the optimal 'cost to go' function should also be a good segmentation for estimating the associated optimal control law. Certain forms of estimators might negate this relationship for some plants, but in general we would expect a significant correspondence.

2. Background

2.1 Adaptive Critic methodology

The experiments presented in this paper train a fuzzy controller using the Dual Heuristic Programming (DHP) methodology, which falls under the more general category of Adaptive Critic Approximate Dynamic Programming (which is in turn subsumed by reinforcement learning). These technologies use a critic whose role is to evaluate the performance of the controller. This evaluation is then used to train the parameters of the controller. We will only touch on several relevant details here; the interested reader is referred to [3] or [6] for a more complete discussion of the issues involved.

In classical Dynamic Programming, the secondary (temporally extensive) utility, *J*, relies on an exponentially weighted sum of future (instantaneous) utilities, *U*. This equation (known as the Bellman equation) is shown below, where γ represents the discount factor with respect to future terms:

$$J(t) = \sum_{k=0}^{\infty} \gamma^k U(t+k)$$

Approximate Dynamic Programming relies on the Bellman Recursion to train the critic to approximate the secondary utility function by evaluating the consistency between successive estimates. There is a tremendous savings in this approach as opposed to Dynamic Programming, which is computationally infeasible for most significant problems. This recursion is expressed as follows:

$$J(t) = U(t) + \gamma J(t+1)$$

Finally, in the DHP approach, the critic network comes to approximate a function that expresses the derivative of this secondary utility with respect to the state vector.

2.2 Fuzzy Models

For the critic and controller modules, we use the multiple output ANFIS (Adaptive Network-based Fuzzy Inference System) architecture [2]. This architecture corresponds directly to a Takagi-Sugeno-Kang (TSK) fuzzy model, which is represented as a feedforward network composed of small computational units.

The antecedent membership function parameters (c,σ) define the center and width of Gaussian membership functions in order to partition the state space, as follows:

$$\mu_{A_i}(x) = e^{-0.5\left(\frac{x-c_i}{\sigma_i}\right)^2}$$

The product of these membership values was used to perform the fuzzy AND operation, which was normalized to determine the final firing strength of each rule:

$$w_i = \mu_{A_i}(x)\mu_{B_i}(y)$$

$$\overline{w}_i = \frac{w_i}{\sum_j w_j}$$

The consequent parameters of the critic and controller define first-order (linear) models. The output of the ANFIS structure is the sum of the product of each rule and its associated firing strength:

$$y = \sum_j \overline{w}_j (m_{1,j} x_j + m_{2,j} y_j + b_j)$$

One requirement for the use of fuzzy critics within the DHP methodology is the ability to compute the Jacobian of the controller, which is needed in order to train the critic network. The ANFIS architecture lends itself easily to this computation; there are only several additional computations involved in this procedure compared to those required to compute the gradient for antecedent parameter tuning.

2.3 The Plant

The plant used in this experiment was first proposed by Narendra and Mukhopadhyay [5]. It is a nonlinear, discrete-time MIMO plant that requires the controller to track a two-variable reference signal. Previous work [4][7] using the DHP methodology has demonstrated excellent control of this plant using neural network implementations (Multi-Layer Perceptrons) of the controller and the critic.

The plant has three state variables (*x*) and a control vector of dimension two (*u*). The plant state equations are expressed as follows:

$$\begin{aligned} x_1(t+1) &= 0.9x_1(t)\sin(x_2(t)) + \\ &\quad \left[2 + 1.5\frac{x_1(t)u_1(t)}{1+x_1^2(t)u_1^2(t)}\right]u_1(t) + \\ &\quad \left[x_1(t) + \frac{2x_1(t)}{1+x_1^2(t)}\right]u_2(t) \\ x_2(t+1) &= x_3[1+\sin(4x_3(t))] + \frac{x_3(t)}{1+x_3^2(t)} \\ x_3(t+1) &= [3+\sin(2x_1(t))]u_2(t) \end{aligned}$$

The reference signals used to test performance were complex sinusoids:

$$\tilde{x}_1(t) = 0.75\sin\left(\frac{2\pi t}{50}\right) + 0.75\sin\left(\frac{2\pi t}{10}\right)$$

$$\tilde{x}_2(t) = 0.75\sin\left(\frac{2\pi t}{30}\right) + 0.75\sin\left(\frac{2\pi t}{20}\right)$$

The goal of controlling this plant is to make the first two state variables track the reference trajectory. This leads to a simple utility (error) function:

$$U(t) = (\tilde{x}_1 - x_1)^2 + (\tilde{x}_2 - x_2)^2$$

In this experiment, the state provided to the controller and the critic consisted of all the plant state variables and the reference trajectory at the next time step.

Finally, the Jacobian of the plant is necessary in order to train the critic (as DHP is a model-dependant methodology). Although these derivatives can be estimated without the use of the plant equations, this approach was not taken here. The non-zero terms of the Jacobian are as follows:

$$\frac{\partial}{\partial x_1(t)} x_1(t+1) = 0.9 \sin x_2(t) + \left[1 + \frac{2\left(1 - x_1^2(t)\right)}{\left(1 - x_1^2(t)\right)^2}\right] u_2(t) + 1.5\left[\frac{1 - x_1^2(t) u_1^2(t)}{\left(1 + x_1^2(t) u_1^2(t)\right)^2}\right] u_1^2(t)$$

$$\frac{\partial}{\partial x_2(t)} x_1(t+1) = 0.9 x_1(t) \cos(x_2(t))$$

$$\frac{\partial}{\partial x_3(t)} x_2(t+1) = 1 + \sin(4x_3(t)) + 4x_3(t)\cos(4x_3(t)) + \frac{1 - x_3^2(t)}{\left(1 + x_3^2(t)\right)^2}$$

$$\frac{\partial}{\partial x_1(t)} x_3(t+1) = 2u_1(t)\cos(2x_1(t))$$

3. Methods

Two primary sets of experiments were conducted: one in which the controller and critic share antecedent parameters, and one in which they had independent sets of parameters. In both instances, there were four possible scenarios: no antecedent training, antecedent training using the critic, antecedent training using the controller, and training using both the controller and the critic. Thus there were eight cases, for each of which eighteen separate trials were conducted.

In order to train the ANFIS networks, a reference signal was chosen which remained at a given value for 40 time steps, then changed to another random value within the range [-1.5, 1.5]. This training was carried out for 240,000 trials. Every 4,000 trials during this period, training was halted to measure performance on the reference signal task. No learning occurred during this interval.

For all our experiments, both the controller and critic structures were trained simultaneously (i.e. on every trial), as this has been shown to have a relatively rapid convergence [4]. The discount factor used in the Bellman recursion was set at 1.0.

Three membership functions were used for each state variable. For the first set of experiments, these were arranged so that their centers had a uniform distribution [-1,0,1]. Their widths were computed such that the adjacent membership functions overlapped at the 0.5 level (Figure 1).

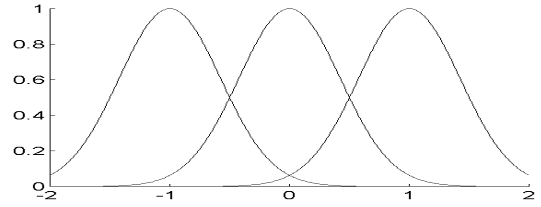

Figure 1: Membership functions for basic comparison

For a second set of experiments, initial membership functions were deliberately chosen to be inadequate. The centers were set at [-2,0,2] and the overlap was set at 0.004 (Figure 2), so that it was of greater importance to tune the antecedent parameters. The same eight cases were used, for each of which nine separate trials were conducted.

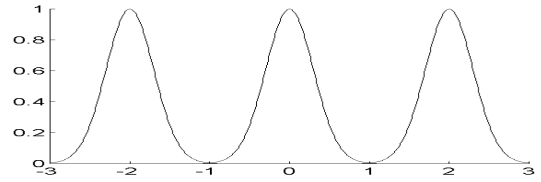

Figure 2: Poorly chosen membership functions

In order to train the critic and controller consequent parameters, learning rates of 0.005 and 0.002 were used (respectively). The learning rates of the Gaussian antecedent parameters (both the center and the width) for the primary results reported in this paper were 0.0002 for the controller and 0.000005 for the critic. The justification of these values is taken up subsequently.

The results refer to the eight cases by number, as follows:

Shared Antecedent Parameters

	Critic not used in training	Critic used in training
Controller not used in training	1	2
Controller used	3	4

in training		

Separate Antecedent Parameters

	Critic not used in training	Critic used in training
Controller not used in training	5	6
Controller used in training	7	8

Note that cases 1 and 5 are identical, in that no antecedent learning is performed (thus separate trials were not conducted for these two cases).

4. Results

4.1 Comparison of Cases

Training shared antecedent parameters offered a slight advantage over training separate parameters. In Figure 3, the averaged results of the eighteen trials for each condition are displayed for the cases in which both the critic and the controller trained their antecedent parameters (cases 4, 8). Training that used the controller and the critic tended to result in the best performance in both the shared and separate antecedent parameter conditions.

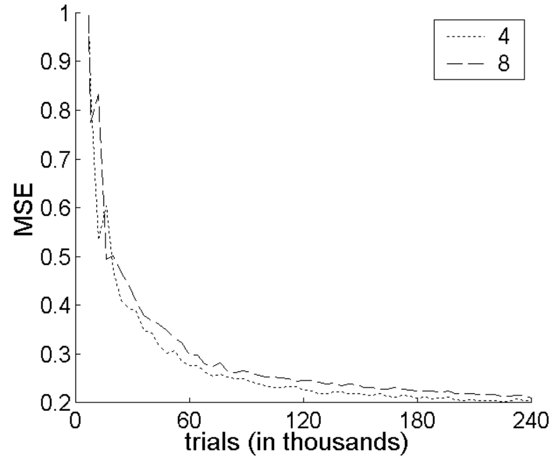

Figure 3: MSE for shared vs. separate parameter cases (4, 8)

The results of sharing antecedent parameters between the controller and critic had differential effects depending on which of these two modules was used to train these parameters. The fastest convergence occurred when either the controller or both the critic and the controller were used to train the antecedents (cases 3,4, Figure 4). Training the antecedents using a learning rate for the critic which was comparable to that of the controller often resulted in instability. At a lower critic learning rate (0.000005), results tended to be only slightly better than those that did not use the critic for antecedent training.

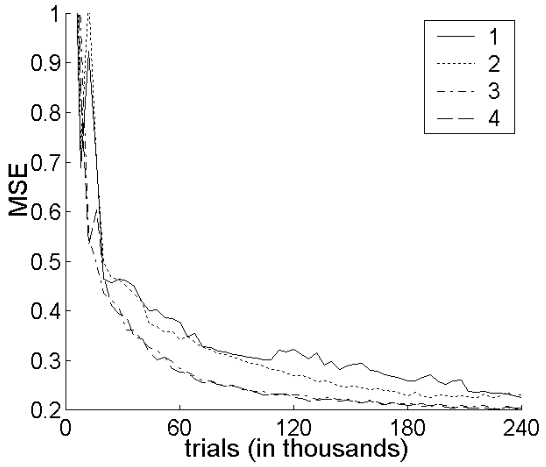

Figure 4: MSE for shared antecedent parameter cases (1,2,3,4)

The results for tuning separate antecedent parameters (Figure 5) were very similar to those obtained by using shared antecedent parameters. Performance, however, was slightly worse.

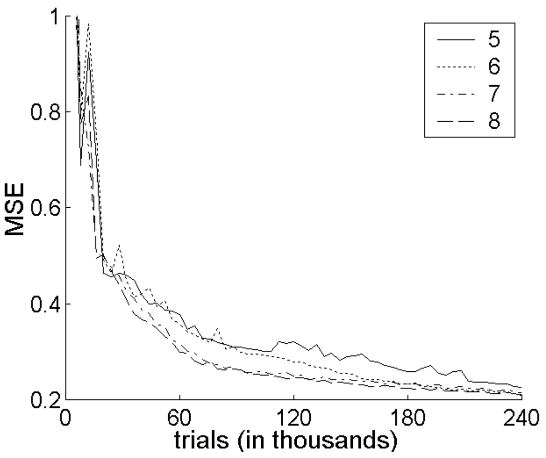

Figure 5: MSE for separate antecedent parameter cases (5,6,7,8)

After 240,000 trials, a comparison of all eight cases (Figure 6) shows that shared parameter cases that used the controller to train the antecedents (cases 3,4) had the best performance. The condition in which separate antecedent parameters were used all performed roughly equivalently, although the case that trained only the controller's parameters (case 7) was slower to reach this value (as can be seen in Figure 5).

In general, the effects mentioned were more pronounced during the initial stages of learning, and decreased with the number of trials. The final difference in MSE was not very large.

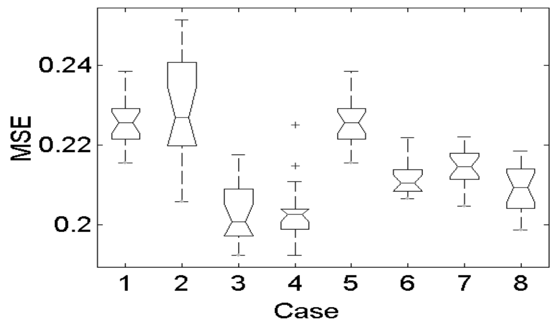

Figure 6: Boxplot of MSE for all groups after 240,000 presentations

4.2 Poorly Chosen Antecedents

The set of membership functions in this experiment produced a very unstable controller. Nine trials were performed for each case. None of the cases which trained their antecedent parameters were stable for all nine trials. The chart below indicates the number of trials (out of nine) for each case that were able to run to completion in a stable manner.

Number of stable trials

Case #	1	2	3	4	5	6	7	8
Number of stable trials	9	8	0	8	9	8	0	1

Training antecedents using only the controller caused instability (cases 3,7). Only two cases converged to values similar to those obtained in the previous experiment, both of which used the controller and critic to train the antecedents. Of these two, the shared parameter case (4) was stable in a greater number of trials (8 as opposed to 1). The results for this experiment are shown in Figure 7.

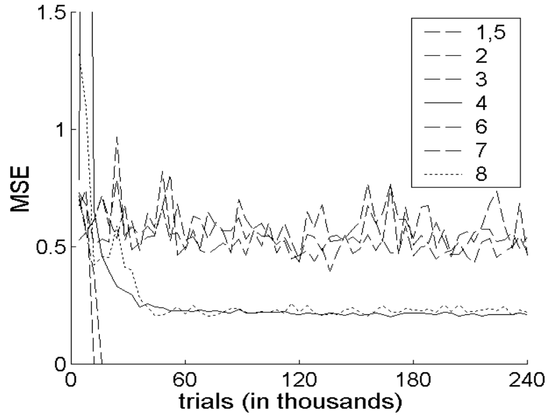

Figure 7: Results for poorly chosen antecedents

5. Conclusion

Shared parameter cases were found to perform as well as, or better than, the corresponding separate parameter cases. This suggests that independent sets of antecedent parameters are unnecessary, even when these parameters are tuned by different controller and critic modules.

Similarly, using both the critic and the controller to train the antecedents was beneficial. This was especially true of the trials using poorly chosen antecedents. In these trials, the only case that consistently learned the control task satisfactorily used both controller and critic training of a shared set of parameters.

One of the primary difficulties encountered in this study derives from the interplay between the various learning rates that are involved. In Adaptive Critic methodologies, the learning rates for the controller and the critic need to be balanced with respect to one another, because each of these modules depends on the performance of the other module in order to accomplish its own task. This coordination of learning rates (which is generally done by a process of trial and error) is aggravated by the balance that needs to be struck between the learning rates of the antecedent and consequent parameters.

Given that training shared antecedent parameters is beneficial, we suggest that Adaptive Critic methodologies that need to train their antecedents should use shared parameters. It not only reduces the number of adjustable parameters of the model, but is also of benefit to the performance of the controller.

References

[1] Hougen, Dean F., Gini, Maria & Slagle, James, "Partitioning Input Space for Reinforcement Learning for Control", *Proceedings of the International Conference on Neural Networks*, 1997, Vol 2 pp 755-760.

[2] Jang, J.-S.R., Sun,C.-T., & Mizutani, E., *Neuro-Fuzzy and Soft Computing*, Prentice Hall, Upper Saddle River, NJ, 1997.

[3] Lendaris, G., Shannon, Thaddeus T., et al., "Dual Heuristic Programming for Fuzzy Control", *Proceedings of 20th NAFIPS International Conference*, Vancouver, BC, Canada, July, 2001.

[4] Lendaris, G., Shannon, Thaddeus T., Rustan, A., "A Comparison of Training Algorithms for DHP Adaptive Critic Neuro-control", *Proceedings of IJCNN 1999*, Washington DC, July 1999.

[5] Narendra, Kumpati S. & Mukhopadhyay, Snehasis, “Adaptive control of Nonlinear Multivariable Systems Using Neural Networks”, *Neural Networks*, Vol 7, issue 5, pp 737-752.

[6] Shannon, Thaddeus T. & Lendaris, George G., “Adaptive Critic Based Dynamic Programming for Tuning Fuzzy Controllers”, *Proceedings of IEEE – Fuzz 2000*, San Antonio, TX, May 2000.

[7] Visnevski, N., & Prokhorov, D. “Control of a Nonlinear Multivariable System with Adaptive Critic Designs”, in C. Dagli, et al (eds) *Intelligent Engineering Systems Through Neural Networks*, (Proceedings conference of Artificial Neural Networks in Engineering) Vol 6 pp 559-565, New York ASME press, 1996.

[8] Werbos, Paul J., “Designs for Reinforcement Learning”, in Miller, T.W., Sutton, R.S., & Werbos, P.J. eds. *Neural Networks for Control*, the MIT Press, Cambridge, MA, 1990, pp 67-95.

[9] Yen, John & Langari, Reza, *Fuzzy Logic: Intelligence, Control, and Information*, Prentice Hall, Upper Saddle River, NJ, 1999.