Pergamon

Mathl. Comput. Modelling Vol. 21, No. 1/2, pp. 109–118, 1995

0895-7177/95 $9.50 + 0.00

0895-7177(94)00198-7

A Comparison of Adaptive Critic and Chemotaxis Methods in Adaptive Control

D. L. STYER
Biomedical Engineering Graduate Group
University of California, Davis, CA 95616, U.S.A.
styer@eecs.ucdavis.edu

V. VEMURI
Department of Applied Science and Lawrence Livermore National Laboratory
University of California, P.O. Box 808, L-794, Livermore, CA 94550, U.S.A.
VEMURI@icdc.llnl.gov

Abstract—Adaptive critic and chemotaxis algorithms are used to control a cart-pole system. Performance of these two methods are compared with earlier results obtained by using the functional link outerproduct method, as well as two other variations of the classical adaptive critic. This work is expected to shed light on biologically plausible adaptive control methods used to control and maintain the postural stability of the human musculo-skeletal system.

INTRODUCTION

Balance and posture of the body is essential to most human locomotion. Because humans are bipeds with two-thirds of their body mass located at about two-thirds of their height from the ground, the mechanism controlling their posture is critical [1]. As the average life expectancy of our population continues to increase, afflictions of the aged are becoming increasingly more important. The balance control degenerates with age and the fear of falling is a major deterrent to the mobility of the elderly. Thus, there has been a heightened interest in postural control and its disorders in the elderly [2]. People suffering from Parkinson's disease are also severely afflicted with postural control problems.

Extensive biomechanical work has shown that modeling the human body as a linked group of rigid bodies is a reasonably accurate first approximation for kinesiological research. The simplest mechanical analog belonging to this category is the problem of balancing an inverted pendulum. Progressively complex models belonging to this family are single-link inverted pendulum, a pendulum mounted on a cart, the so-called cart-pole or broom balance problem, and the double-link inverted pendulum. The relation between the inverted pendulum problem and postural control was recognized early. By constraining an inverted pendulum with abstract springs and dash pots, it is possible to model cartilage and ligamental behavior [3]. Here, we investigate various artificial neural network controllers to be used in the complete (mechanical and control components) postural model of the human. This work is expected to shed light on biologically plausible adaptive control methods used to control and maintain the postural stability of the human musculo-skeletal system.

This paper reports some preliminary results of our experiments with a variety of neural network based adaptive control algorithms on two models of human posture. Specifically, the focus of this paper is on two adaptive control algorithms that rely on methods that fall outside the realm of supervised learning. This line of research is motivated by the observation that many biologically

plausible learning and control mechanisms do not rely on the existence of a teacher. The learning schemes often found in biological systems are either of the self-organizing or reinforcement type. Although it is hard to put the two methods under consideration in strict categories, it is safe to say that one of the methods, chemotaxis, has some self-organizing characteristics, whereas the other method, adaptive critic, is believed to be a reinforcement learning method.

Here, we compare the performance of artificial neural networks (ANNs) that use the adaptive critic and chemotaxis algorithms to learn and maintain control of a dynamic system. The methods discussed here have direct bearing on a wide range of problems involving the balance and stability of flexible structures, such as space-borne structures, two-legged walking robots, aiming of rocket thrusters, and so on. The power of classical adaptive control techniques to solve these problems is limited because they work best when the system parameters are known and when the dynamic equations are linear and deterministic. Recent studies indicate that ANNs can play an important role when:

(a) the system dynamics are nonlinear,
(b) the system is operating in a nondeterministic environment, and
(c) the system parameters are hard to estimate [4].

ANN control implementations (neurocontrollers) fall into one of the following categories: supervised control, direct inverse control, neural adaptive control, and adaptive critic control [5].

The Adaptive Critic (AC), proposed and promoted by Barto [6] belongs to a category that is somewhere between supervised learning and unsupervised (or self-organized) learning. In this method, a teacher or an oracle inspects the system's (or the network's) response and simply provides a qualitative evaluation of that response with such comments like "good" or "bad." Based on this type of qualitative judgmental feedback, the controller makes temporally local decisions in order to optimize some temporally global objective (or utility) function.

Chemotaxis, a method similar to that used by primitive organisms to search for food, has been proposed by Bremmerman and Anderson [7] as a potential learning method. It uses no teacher. This algorithm essentially implements a biased random search in weight space. The search direction is biased (i.e., not truly random) because favorable directions are rewarded. In general, weight changes that improve performance are kept, while others are discarded. The act of rewarding favorable directions may be construed as a reinforcement signal. As in the adaptive critic, the utility of a decision taken at a given time step may not become evident until a sequence of steps (or control actions) have been taken. In spite of these apparent similarities, fundamental differences exist between these two methods. A more detailed description of chemotaxis is provided below. This paper summarizes efforts to understand the operational similarities and differences between these methods.

THE ADAPTIVE CRITIC ALGORITHM

The family of Adaptive Critic (AC) methods evaluated here implement direct adaptive control [8]; they determine the control rule without forming an explicit model of the plant. There are more complex AC methods [5] that rely on a system identification procedure to form an explicit model of the system and determine the control rule from the model (indirect control).

The adaptive critic method has its origins in the associative search network [6]. The associative search network (ASN) is capable of performing pattern recognition tasks. In doing so, this network conducts a search for the output that maximizes a reinforcement signal based on context input from the environment. The ASN combines two types of learning: stochastic optimization (or hill-climbing) that solves the pattern recognition problem, and a stochastic automaton that performs a search to maximize the reinforcement. Both the stochastic hill climbing and stochastic automata searches are iterative procedures [9]. The stochastic nature of the ASN is provided by a probabilistic neuron [10]. The adaptation of the weights in the ASN is proportional to the change in reinforcement and the change in its action. This update algorithm is not capable

of solving an associative search problem unless the reinforcement function implemented by the environment varies smoothly over time [6]. This is due to the problem of context transitions that occur in the pattern recognition phase when the pattern changes. The problem is exacerbated by the quantization of state space when the algorithm is used in a control task. One of the methods used to alleviate this problem was to introduce an element that would predict the reinforcement [6]. The element was aptly called a predictor. When a predictor is included, adaptation of the ASN weights is proportional to the change in the action and the difference between predicted reinforcement and actual reinforcement. Weight changes in the predictor are proportional to the difference between the predicted reinforcement and the actual reinforcement. The predictor element and the ASN are the precursors of the adaptive critic element and adaptive search element used in adaptive critic methods [11].

The AC method uses two functionally different elements: the Associative Search Element (ASE) and the Adaptive Critic Element (ACE). The ASE is the decision maker; it constructs associations between the input and the control output, under the influence of a reinforcement signal. The decisions are stochastic in nature; they are determined based on a probability function of the weighted inputs. Therefore, the ASE is functionally the same as the previously discussed ASN.

However, the output of the ACE is much more than merely a prediction of the reinforcement, as in the "predictor" element. The ACE implements a variation of the temporal-difference method [12]; its output is called secondary (internal) reinforcement $\hat{r}$. It is the sum of the current primary reinforcement r and the difference between the discounted current prediction $\gamma\, p(t)$ and the previous prediction $p(t-1)$ of reinforcement

$$\hat{r}(t) = r(t) + \gamma p(t) - p(t-1). \tag{1}$$

Thus, the ACE constructs an evaluation function of performance that is more informative than a simple reinforcement (failure indicator) signal.

The adaptations of the weights in both the ACE and ASE are a function of this secondary reinforcement (proportional to the secondary reinforcement and an eligibility function). Anderson's work [13] represents further evolution of the AC method and appears more complex than the implementation of Barto *et al.* [11]. Yet, Anderson's method is essentially the same with the exception that the inputs are analog, not binary valued. The adaptation of the weights in the ACE is essentially the same in both algorithms. However, the adaptation of the weights in Anderson's ASE includes a term not used by Barto *et al.* This term is the difference between the expected action and the action taken. Thus, the adaptation of the ASE weights is proportional to the secondary reinforcement and the difference between the action and the expected action. The i^{th} weight elements of both the ACE and ASE are updated using a rule of the following form:

$$w_i(t+1) = w_i(t) + \alpha \hat{r}(t) e_i(t), \tag{2}$$

where α is the learning rate and $e_i(t)$ is the eligibility at time t. Equations defining the eligibility function and the trace function used in the ACE and ASE updates, respectively, are defined in Barto *et al.* [11]. These functions are used as a method to solve the "temporal credit problem." The temporal credit problem is stated as follows. "What was responsible for the current situation (i.e., what control actions were responsible for the failure?)."

THE CHEMOTAXIS ALGORITHM

In the chemotaxis algorithm, the initial weights $\mathbf{W}_0$ are assigned random values between -0.1 and 0.1. The performance $J(\mathbf{W}_0)$ of the weight vector is determined. A random change vector $\mathbf{W}_{\text{rc}}$ is drawn from a multivariate Gaussian distribution with zero mean and unit standard deviation. A new weight vector $\mathbf{W}_1$ is formed by adding the random change vector to the current

weight vector:

$$\mathbf{W}_1 = \mathbf{W}_0 + h\mathbf{W}_{\text{rc}}, \tag{3}$$

where h is the step size parameter (adjustable during learning). The performance of the new weight vector $J(\mathbf{W}_1)$ is then calculated. If the performance of the new configuration is better (smaller error) than the original configuration, then the new weight vector is retained. The values of $\mathbf{W}_1$ are assigned to $\mathbf{W}_0$ and an additional step is taken along the random change vector $\mathbf{W}_{\text{rc}}$. The step parameter h is then increased. The weight vector changes will continue along this random change vector until performance improvement ceases. Therefore, the search will follow a declining slope, if it finds one. If the performance of the new configuration is worse (larger error) than the original configuration, the new weight vector is discarded. A new random change vector is chosen and the process is repeated ($\mathbf{W}_1$ is formed and the performance is evaluated, etc.). If after "several" trials, a successful direction is not found, then learning has stalled; the step size parameter h is reduced and the search is continued. The parameters used in the implementation of chemotaxis are:

$$h_{\text{hstart}} = 0.08, \qquad h_{\text{hstall}} = 0.75h, \qquad h_{\text{update}} = 1.50h.$$

CONTROLLER IMPLEMENTATIONS

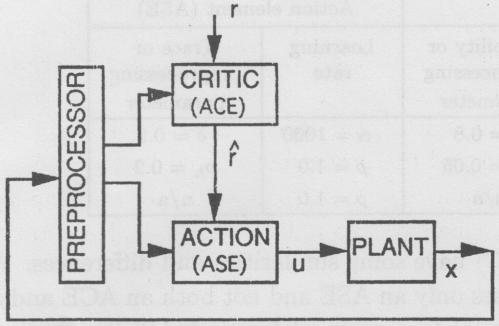
The ultimate goal of this research is to develop an artificial neural network controller that is capable of learning to maintain erect posture of a musculo-skeletal model of a standing human. The family of AC methods and chemotaxis have been chosen for investigation because of their biological plausibility in effecting the necessary control signal.

The cart-pole system has been chosen as the vehicle to test these methods; many researchers have studied ANN implemented control using it. Some used supervised learning [14], others genetic algorithms [15], while others implemented variations on the adaptive critic method [11,13,16–19]. The equations of motion and system parameters used are those detailed by Anderson [13]. In our earlier paper [18], the same system was used to study the properties of two versions of the adaptive critic and the Functional Link Outerproduct (FLO) proposed by Klassen *et al.* [20]. In all these studies, the problem solved is the same, but the methods are different. Briefly, the problem involves a cart that is free to move back and forth along a finite-length (2.4 meter) one-dimensional track. A pole attached to the cart is free to rotate within a specific range (±12 degrees) in the vertical plane of the cart and track. The control problem involves keeping the cart and pole within their bounds while exerting a fixed force (10 newtons), at discrete instants of time (0.02 seconds), either to the left or the right on the cart. The output of the problem generator is four state variables (position of the cart on the track x, angle of the pole with the vertical θ, and their respective velocities) and a failure indicator r (used to signal if the cart or pole had exceeded its bounds). The state variables are taken as inputs to the Artificial Neural Network (ANN) controller. They are used in the determination of the control action applied to the cart-pole system at each time increment. In the AC and Chemotaxis methods the inputs to the ANN controller are the state variables X, and the primary reinforcement r; the output is the control variable u. The primary reinforcement is defined:

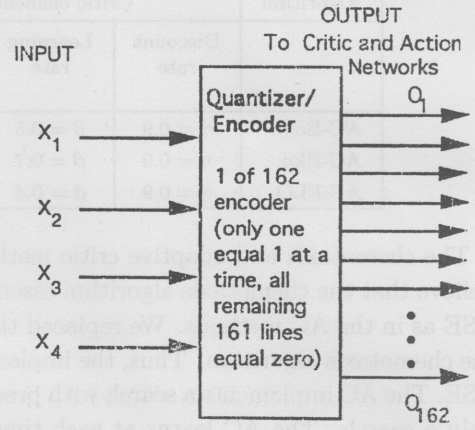
$$r = \begin{cases} 0, & \text{if } |\theta| \le 12^\circ \text{ and } |x| \le 2.4m, \\ -1, & \text{otherwise.} \end{cases} \tag{4}$$

In the AC method, the ASE uses a stochastic processing element (probabilistic neuron) in the determination of the control action on the environment. The probabilistic nature of the ASE is provided in the following manner. The state representation vector is processed in the "normal" manner of ANN (summed and passed through a sigmoid squashing function) and mapped into a range from zero to one. A random number is generated between zero and one, and the two values

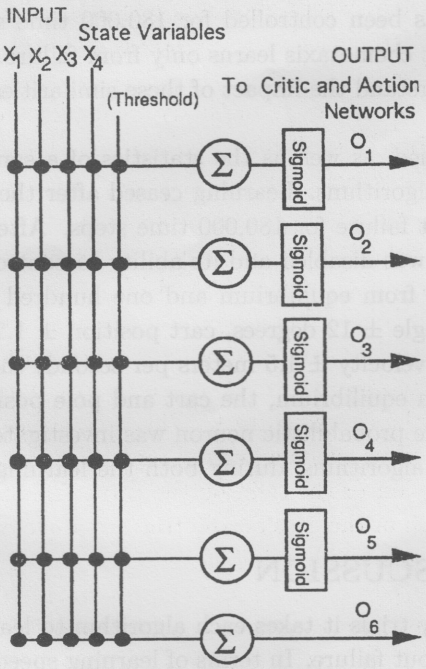
are compared. If the random number is less than the mapped number, the fixed force is applied to the left, otherwise it is applied to the right. In a probabilistic neuron, the weighted input determines the probability of a specific control action output u. In a deterministic neuron, the mapped value is compared to a threshold value, here 0.5, instead of a random number. Thus, the weighted input determines the output. The importance of the probabilistic nature of the output neuron will be illustrated.



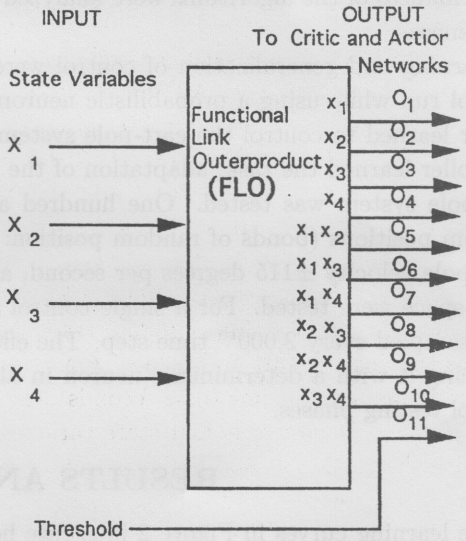
(a) A schematic of the adaptive critic.



(b) AC-Box preprocessor.



(c) AC-Bkp preprocessor.



(d) AC-FLO preprocessor.

Figure 1. Schematic of adaptive critic. (Preprocessor—quantizer, neural network layer or functional link outerproduct layer).

The AC methods used are essentially the same, differing primarily in how the state variables are preprocessed before being applied to the functional elements of the AC (Figure 1a). Barto *et al.*'s [6] method (referred to as AC-Box) preprocesses the state variables so that the state space is partitioned (quantized) into 162 discrete boxes (see Figure 1b). The state of the system is represented by a binary vector whose components are all zero except the one that describes the position of the box that contains the state vector. Reducing the amount of *a priori* knowledge required, Anderson's [13] method (referred to as AC-Bkp) preprocesses the state variables in an ANN layer that uses backpropagation (see Figure 1c). Finally, the Functional Link

Outerproduct (FLO) method (referred to as AC-FLO) preprocesses the state variables by adding pairwise combinations of the state variables to the representation space (see Figure 1d [18,20]). The learning equations used in these AC methods are recorded in the references [6,13]. The parameters used are summarized below in Table 1.

Table 1. Adaptive critic algorithm parameters.

Algorithm	Critic element (ACE)			Action element (ASE)	
	Discount rate	Learning rate	Eligibility or preprocessing parameter	Learning rate	Trace or preprocessing parameter
AC-Box:	$\gamma = 0.9$	$\beta = 0.5$	$\lambda = 0.8$	$\alpha = 1000$	$\delta = 0.9$
AC-Bkp:	$\gamma = 0.9$	$\beta = 0.7$	$\beta_h = 0.05$	$\rho = 1.0$	$\rho_h = 0.2$
AC-FLO	$\gamma = 0.9$	$\beta = 0.4$	n/a	$\rho = 1.0$	n/a

The chemotaxis and adaptive critic methods (AC) have some similarities and differences. We believe that the chemotaxis algorithm essentially uses only an ASE and not both an ACE and an ASE as in the AC methods. We replaced the deterministic neuron with a probabilistic neuron in the chemotaxis algorithm. Thus, the implemented chemotaxis method more closely resembles the ASE. The AC implements a search with prediction, whereas the chemotaxis algorithm implements only a search. The AC learns at each time step; yet, learning takes place only after each trial in the chemotaxis algorithm. A trial begins with the cart-pole system in starting position and ends with a failure or when the cart-pole system has been controlled for 180,000 time steps. Therefore, the AC learns from success and failure, but chemotaxis learns only from failure. The performances of the algorithms were analyzed to understand the impact of these similarities and differences.

Learning and generalization of control were examined, as well as the statistics of a sampled control run while using a probabilistic neuron in all algorithms. Learning ceased after the controller learned to control the cart-pole system without failure for 180,000 time steps. After the controller learned the task, adaptation of the weights was disabled and its ability to control the cart-pole system was tested. One hundred attempts from equilibrium and one hundred from random positions (bonds of random position: pole angle $\pm$ 12 degrees, cart position $\pm$ 1.2 meters, pole velocity $\pm$115 degrees per second, and cart velocity $\pm$ 1.5 meters per second) than in state space were tested. For a single control run from equilibrium, the cart and pole positions were sampled every $2{,}000^{\text{th}}$ time step. The effect of the probabilistic neuron was investigated by replacing it with a deterministic neuron in all of the algorithms, during both the learning and control testing phases.

RESULTS AND DISCUSSION

The learning curves in Figure 2 illustrate how many trials it takes each algorithm to learn to control the cart-pole system for 18000 time steps without failure. In terms of learning speed, the AC-Box topped the list, followed by chemotaxis, AC-FLO, and AC-Bkp. In terms of ability of the algorithms to control (after learning was ceased), the cart-pole system from an equilibrium starting point, there was no significant difference in the performance of the four methods (see Figure 3). However, the ability of the controllers to generalize the learned control rule (control from random starting positions) differed. The chemotaxis algorithm generalized control the best and the AC-Box method generalized control the worst. The performance of the other two methods fell in between.

The goal of the control task was only to keep the cart and pole between the allowed bounds. However, it was noticed that the chemotaxis algorithm maintains the cart closest to the equilibrium position and has the second largest variability in control (see Table 2). In AC-Bkp the average cart position deviates the furthest from equilibrium and has the largest variability

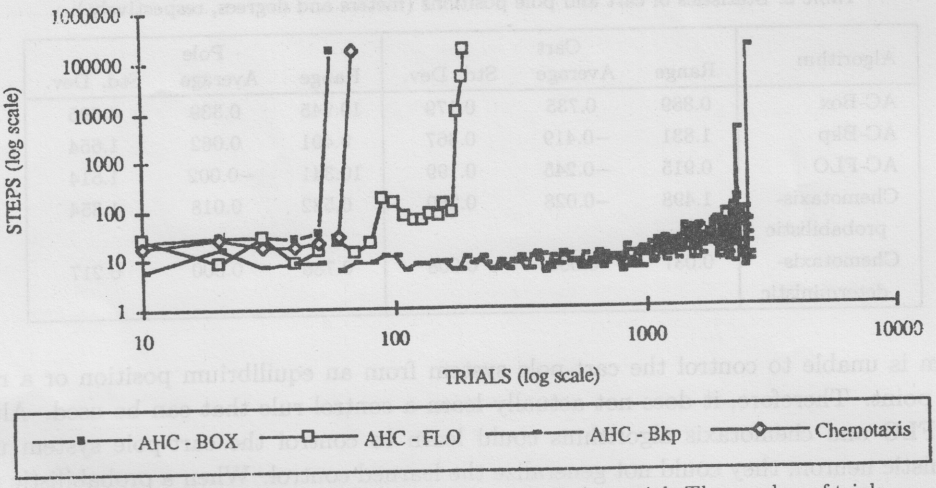


Figure 2. Learning curves. Number of steps controlled vs. trial. The number of trials required to control cart-pole for 18000 time steps are: 58 trials, AC-Box, 2257 trials, AC-Bkp, 192 trials, AC-FLO, 69 trials, chemotaxis.

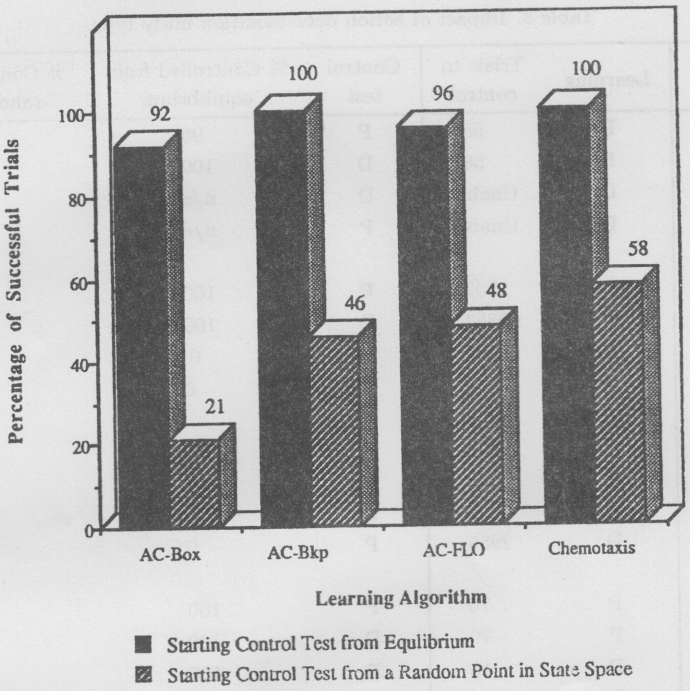


Figure 3. Percentage of successful trials vs. learning algorithm.

in control. The variability of the position of the pole is less in the AC-FLO and chemotaxis methods than in the other two methods. Although, the chemotaxis method had the smallest standard deviation in pole position the AC-FLO method, on the average, kept the pole closest to equilibrium.

Substitution of a deterministic neuron for the probabilistic neuron in the ASE during the learning phase is detrimental for the AC and chemotaxis algorithms (see Table 3). The AC-Box method gets stuck in a local optimum and is unable to learn after 6000 trials using the deterministic neuron. The number of trials required to learn increased in the AC-Bkp (2557 to 5629) algorithm and dramatically increased in the AC-FLO (192 to 2985) algorithm. This substitution improves the learning rate (decreases the number of trials from 70 to 15) of the chemotaxis algorithm. After apparently learning a control rule, with learning ceased the AC-Bkp

Table 2. Statistics of cart and pole positions (meters and degrees, respectively.)

Algorithm	Cart			Pole		
	Range	Average	Std. Dev.	Range	Average	Std. Dev.
AC-Box	0.889	0.735	0.179	19.945	0.339	4.695
AC-Bkp	1.831	−0.419	0.367	9.491	0.062	1.654
AC-FLO	0.915	−0.245	0.199	10.341	−0.002	1.514
Chemotaxis-probabilistic	1.498	−0.028	0.280	6.582	0.018	1.354
Chemotaxis-deterministic	0.037	0.000	0.008	0.786	0.000	0.217

algorithm is unable to control the cart-pole system from an equilibrium position or a random starting point. Therefore, it does not actually learn a control rule that can be used. Although the AC-FLO and chemotaxis algorithms could learn to control the cart-pole system using a deterministic neuron, they could not generalize the learned control. When a probabilistic neuron is used in the learning phase, there were only minor differences in the control test phase between the use of a probabilistic and a deterministic neuron in the AC and chemotaxis methods.

Table 3. Impact of action determination unity type.

Algorithm	Learning	Trials to control	Control test	% Controlled from equilibrium	% Controlled from random point
AC-Box	P	58	P	95	21
	P	58	D	100	18
	D	Unable	D	n/a	n/a
	D	Unable	P	n/a	n/a
AC-Bkp	P	2557	P	100	46
	P	2557	D	100	41
	D	5629	D	0	0
	D	5629	P	0	0
AC-FLO	P	192	P	96	48
	P	192	D	100	52
	D	2985	D	100	0
	D	2985	P	0	0
Chemotaxis	P	70	P	100	58
	P	70	D	100	62
	D	15	D	100	1
	D	15	P	0	0

Deterministic (D)—Weighted sum determines output action.

Probabilistic (P)—Weighted sum determines probability of output action.

The substitution of the deterministic neuron in the associative search element of the AC algorithm during learning exposes the importance of the probabilistic neuron in the method. This reveals that the algorithm is primarily a stochastic automaton; it searches probability (action) space for appropriate control actions. The original AC algorithms learn to maximize the probability of action (probability close to 1 or 0). This was revealed by substitution of the deterministic neuron for the probabilistic neuron during the control testing (see Table 3). Initially, either action is equally probable. As more of the state space is searched the probability of a particular action is found to increase for a given point in state space. Thus, the probabilistic nature of control gradually evolves to resemble deterministic control as learning progresses. These characteristics were also seen in the chemotaxis algorithm that uses a probabilistic neuron.

The use of a probabilistic neuron, inclusion of uncertainty, in the chemotaxis algorithm provides a mechanism for the algorithm to avoid local minima. Thus, we have a method that explores both parameter (weight) space and probability (action) space simultaneously. Inclusion of the probabilistic nature in the algorithm makes it more closely parallel to the biological neuron. The biological neuron activation is influenced by other cellular mechanisms besides the net level of excitation.

All the above tested ANN algorithms have been used in the control of both a single and double-link inverted pendulum simulation of a standing human subjected to a postural disturbance. In scaling from the simplest to a more difficult control task, the time required to learn control increased by more than 450% from the one-link inverted pendulum to the cart-pole system except for the AC-FLO which increased 186%. Likewise, the time required to learn increased by more than 100% from the cart-pole to the two-link inverted pendulum except for the AC-FLO which had an increase of 63%. The smaller increases required by the AC-FLO could be due to the explicit nonlinear nature of the signals applied to the ASE network. It should be noted that in our two-link inverted pendulum simulations, two control signals were used, one for each joint. We are conducting further analysis to evaluate the quality of the controller (control law learned) in the one and two link simulations.

REFERENCES

1. D.A. Winter, *Biomechanics and Motor Control of Human Movement,* 2nd ed., John Wiley & Sons, New York, (1990).
2. V.P. Panzer, T.A. Zeffiro and M. Hallett, Kinematics of standing posture associated with aging and Parkinson's Disease, In *Disorders of Posture and Gait,* (Edited by T. Brandt *et al.*) pp. 390–393, G. T. Verlag, New York, (1990).
3. B. Bavarian, B.F. Wyman and H. Hemami, Control of the constrained planar simple pendulum, *International Journal Control* **37** (4), 741–753 (1983).
4. K.S. Narendra and K. Parthasarathy, Identification and control of dynamic systems using neural networks, *IEEE Trans. Neural Networks* **1** (1), 4–27 (1990).
5. P.J. Werbos, Neurocontrol and related techniques, In *Handbook of Neural Computing Applications,* (Edited by A. Maren, C. Harston and R. Pap) pp. 345–380, Academic Press, San Diego, CA, (1990).
6. A.G. Barto, R.S. Sutton and P.S. Brouwer, Associative search network: A reinforcement learning associative memory, *Biol. Cybern.* **40**, 201–211 (1981).
7. H.J. Bremermann and R.W. Anderson, How the brain adjusts synapses—maybe, In *Automated Reasoning: Essays in Honor of Woody Bledsoe,* (Edited by R.S. Boyer) pp. 119–147, Kluwer Academic Publishers, Boston, (1991).
8. R.S. Sutton, A.G Barto and R.J. Williams, Reinforcement learning is direct adaptive optimal control, *IEEE Control Systems Magazine* **12** (2), 19–22 (1992).
9. K.S. Narendra and M.A.L. Thathachar, Learning automata—A survey, *IEEE Transactions on Systems, Man, and Cybernetics* **4** (4), 323–334 (1974).
10. R.J. Williams, Reinforcement learning in connectionist networks: A mathematical analysis, Technical Report ICS 8605, Institute for Cognitive Science, University of California at San Diego, La Jolla, CA (1986).
11. A.G. Barto, R.S. Sutton and C.W. Anderson, Neuronlike adaptive elements that can solve difficult learning control problems, *IEEE Transactions on Systems, Man, and Cybernetics,* **13** (5), 834–846 (1983).
12. R.S. Sutton, Learning to predict by methods of temporal difference, *Machine Learning* **3**, 9–44 (1988).
13. C.W. Anderson, Learning to control an inverted pendulum using neural networks, *IEEE Control Systems Magazine* **9** (3), 31–37 (1989); MIT Press, Cambridge, MA, 5-58.
14. A. Guez and J. Selinsky, A trained neuromorphic controller, *Journal of Robotic Systems* **5** (4), 363–388 (1988).
15. A.P. Wieland, Evolving controls for unstable systems, In *Connectionist Models: Proc. 1990 Summer School,* (Edited by D. Touretzky) pp. 91–102, Morgan Kaufmann, San Mateo, CA, (1991).
16. C. Lin and H. Kim, CMAC-based adaptive critic self-learning control, *IEEE Transactions on Neural Networks* **2** (5), 530–533 (1991).
17. B.E. Rosen, J.M. Goodwin and J.J. Vidal, Adaptive range coding, In *Neural Information Processing Systems 3,* (Edited by R.P. Lippmann *et al.*) pp. 486–492, Morgan Kaufmann, San Mateo, CA, (1991).
18. D.L. Styer and V. Vemuri, Preprocessing inputs for adaptive critic control, In *Proceedings of 3rd IFAC International Workshop on Artificial Intelligence in Real Time Control,* September 23–25, 1991, Napa, CA.

19. B. Widrow, N.K. Gupta and S. Maitra, Punish/reward: Learning with a critic in adaptive threshold systems, *IEEE Transactions on Systems, Man, and Cybernetics* **3** (5), 455–465 (1973).
20. M.S. Klassen, Y.H. Pao and V. Chen, Characteristics of the functional-link net: A higher order delta rule net, In *IEEE Proceedings of 2nd Annual International Conference on Neural Networks,* June, 1988, San Diego, CA.