An Analysis of Actor/Critic Algorithms using Eligibility Traces: Reinforcement Learning with Imperfect Value Functions

Hajime Kimura* 
Tokyo Institute of Technology 
gen@fe.dis.titech.ac.jp

Shigenobu Kobayashi 
Tokyo Institute of Technology 
kobayasi@dis.titech.ac.jp

Abstract

We present an analysis of actor/critic algorithms, in which the actor updates its policy using eligibility traces of the policy parameters. Most of the theoretical results for eligibility traces have been for only critic’s value iteration algorithms. This paper investigates what the actor’s eligibility trace does. The results show that the algorithm is an extension of Williams’ REINFORCE algorithms for infinite horizon reinforcement tasks, and then the critic provides an appropriate reinforcement baseline for the actor. Thanks to the actor’s eligibility trace, the actor improves its policy by using a gradient of actual return, not by using a gradient of the estimated return in the critic. It enables the agent to learn a fairly good policy under the condition that the approximated value function in the critic is hopelessly inaccurate for conventional actor/critic algorithms. Also, if an accurate value function is estimated by the critic, the actor’s learning is dramatically accelerated in our test cases. The behavior of the algorithm is demonstrated through simulations of a linear quadratic control problem and a pole balancing problem.

1 Introduction

Actor/critic architecture is an adaptive version of policy iteration [Kaelbling et al.96]. In general, policy iteration alternates two phases: a policy evaluation phase and a policy improvement phase. The actor implements a stochastic policy that maps from a representation of a state to a probability distribution over actions. The critic attempts to estimate the evaluation function for the current policy. The actor improves its control policy using critic’s temporal difference (TD) as an effective reinforcement. In many cases, the policy improvement is executed concurrently with the policy evaluation, because it is not feasible to wait for the policy evaluation to converge.

The actor/critic algorithms have been successfully applied to a variety of delayed reinforcement tasks; ASE/ACE architecture for a pole balancing [Barto et al. 83] [Gullapalli 92], RFALCON for a pole balancing and for control of a ball-beam system [Lin et al. 96], a cart-pole swing-up task [Doya 96]. Although convergence proofs for the actor/critic algorithms are less than value-iteration based algorithms such as Q-learning [Watkins et al. 92], the actor/critic algorithms have the following practical advantages.

- It is easy to implement multidimensional continuous action, that is often mixed with discrete action [Gullapalli 92]. Because the actor selects action by its stochastic policy, therefore problems of action selection like in Q-learning does not exist. The Q-learning needs to estimate returns for all state-action pairs, but the critic would estimate only the return of each state.

- Memory-less stochastic policies can be considerably better than memory-less deterministic policies in the case of partially observable Markov decision processes (POMDP’s) [Singh 94] [Jaakkola 94] or multi-player games [Littman 94].

- It is easy to incorporate an expert’s knowledge into the learning system by applying conventional supervised learning techniques to the actor [Clouse et al. 92].

Eligibility traces are a fundamental mechanism that has been widely used to handle delayed reward [Singh 96]. Also the traces are often used to overcome non-Markovian effects [Sutton 95],

* Interdisciplinary Graduate School of Science and Engineering, Tokyo Institute of Technology, 4259 Nagatsuta Midori-ku Yokohama 223-8502 JAPAN.
[Pendrith et al. 96]. In Barto, Sutton and Anderson’s ASE/ACE architecture, both the critic and the actor make use of the eligibility trace. Theoretical results of eligibility traces in the context of $TD(\lambda)$ [Sutton 88] have been obtained. But, in actor/critic algorithms, the effect of the actor’s trace has not been investigated. This paper presents an analysis of an actor/critic algorithm, in which the actor improves its policy using eligibility traces of the policy parameters. This may be the first analysis of the actor’s eligibility traces.

2 Discounted Reward Criteria

At each discrete time $t$, the agent observes $x_t$ containing information about its current state, select action $a_t$, and then receives an instantaneous reward $r_t$ resulting from state transition in the environment. In general, the reward and the next state may be random, but their probability distributions are assumed to depend only on $x_t$ and $a_t$ in Markov decision processes (MDPs), in which many reinforcement learning algorithms are studied. The objective of reinforcement learning is to construct a policy that maximizes the agent’s performance. A natural performance measure for infinite horizon tasks is the cumulative discounted reward:

$$V_\pi = \sum_{k=0}^{\infty} \gamma^k r_{t+k},$$  

(1)

where the discount factor, $0 \leq \gamma < 1$ specifies the importance of future rewards. $V_\pi$ is called the actual return, that specifies how good the reward sequence after time $t$ is. By this notation, the goal of the learning is to maximize the expected return. In MDPs, the expected return can be defined for all states as:

$$V^\pi(x) = E_x \sum_{k=0}^{\infty} \gamma^k r_k | x_0 = x,$$

(2)

where $E_x$ denotes the expectation assuming the agent always uses stationary policy $\pi$. $V^\pi(x)$ is called the value function, that specifies how good the given state $x$ is. In MDPs, the goal of the learning is to find an optimal policy that maximizes the value of each state $x$ defined by Equation 2. Although similar value functions can be given in POMDPs, difficulties to define the optimum have pointed out in [Singh 94].

3 Actor/Critic Algorithms

Figure 1 and 2 give an overview of actor/critic algorithms [Sutton 90] [Crites et al. 94]. There are many ways to implement the policy and its updating scheme in the actor. The algorithms for the critic are mostly TD methods. We should notice the following two points; one is the actor implements stochastic policy, the other is the actor improves its policy using TD-error. This paper especially investigates an algorithm for the actor.

1. The agent observes $x_t$ in the environment, and the actor executes action $a_t$ according to the current stochastic policy $\pi$.
2. The critic receives the immediate reward $r_t$, and then observes the resulting next state $x_{t+1}$. The critic provides TD error as an useful reinforcement feedback to the actor, according to:

$$\text{TD-error} = (r_t + \gamma \hat{V}(x_{t+1}) - \hat{V}(x_t)),$$

where $0 \leq \gamma < 1$ is the discount factor, $\hat{V}(x)$ is an estimated value function by the critic.
3. The actor updates the stochastic policy using the TD-error. If (TD-error) > 0, action $a_t$ performed relatively good and its probability should be increased. If (TD-error) < 0, action $a_t$ performed relatively poorly and its probability should be decreased.
4. The critic updates estimated value function $\hat{V}(x)$ according to TD methods. e.g., $TD(0)$ algorithm adjusts

$$\hat{V}(x_t) = \hat{V}(x_t) + \alpha \text{TD-error},$$

where $\alpha$ is the learning rate.
5. Go to step 1.

Figure 2: Main loop of the generic actor/critic algorithm.
4 Adding Eligibility Trace to the Actor

4.1 Function Approximation for Stochastic Policies

In this paper, \(\pi(a, W, x)\) denotes probability of selecting action \(a\) under the policy \(\pi\) in the observation \(x\). The \(\pi(a, W, X)\) is taken to be a probability density function when the set of possible actions is continuous. The policy is represented by a parametric function approximator using the internal variable vector \(W\). The agent can improve the policy \(\pi\) by modifying \(W\). For example, \(W\) corresponds to synaptic weights where the action selecting probability is represented by neural networks, or \(W\) means weight of rules in classifier systems. The advantage of using the notation of the parametric function \(\pi()\) is that computational restriction and mechanisms of the agent can be specified simply by a form of the function, and then we can provide a sound theory of learning algorithms for arbitrary types of the actor.

4.2 Details of the Algorithm

Figure 3 specifies the actor/critic algorithm that uses the eligibility trace in the actor. The ASE/ACE system configured for pole-balancing [Barto et al. 83] is just an instance of this algorithm. The agent's eligibility in step 3 is the same variable defined in Williams' REINFORCE algorithms [Williams 92]. The eligibility \(\epsilon_i(t)\) specifies a correlation between the associated policy parameter \(w_i\) and the executed action \(a_i\). The eligibility trace \(D_i(t)\) is a discounted running average of eligibility. It accumulates the agent's history. When a positive reinforcement is given, the agent updates \(W\) so that the probability of actions recorded in the history is increased. It means the TD-error at the time \(t\) affects not only the action \(a_t\) but also \(a_{t+1}, a_{t+2}, \cdots\). At first glance, this idea is senseless for improving the policy, but it has very interesting features given in detail later. Note that the algorithm shown in Figure 3 is identical to a stochastic gradient ascent for discounted reward [Kimura et al. 97] when the agent's discount factor \(\beta = \gamma\) and the \(\hat{V}(x)\) in the critic equals a constant \(\delta\) for all observations.

The actor requires a memory to implement \(W\) for the policy and to implement \(D_i\) for the eligibility trace. The amount of the memory for \(D_i\) is equal to \(W\)’s.

4.3 An Analysis of the Algorithm

Assume that the agent's discount factor \(\beta\) equals \(\gamma\), and for all \(t < 0\), \(D_i(t) = 0\), then the algorithm shown

1. The agent observes \(x_t\), and the actor executes action \(a_t\) with probability \(\pi(a_t, W, x_t)\).
2. The critic receives the immediate reward \(r_t\), and then observes the resulting next state \(x_{t+1}\). The critic provides TD error to the actor according to

\[
\text{(TD-error)} = r_t + \gamma \hat{V}(x_{t+1}) - \hat{V}(x_t),
\]

where \(0 \leq \gamma < 1\) is the discount factor, \(\hat{V}(x)\) is an estimated value function by the critic.
3. The actor updates the stochastic policy using the TD-error according to:

\[
\epsilon_i(t) = \frac{\partial}{\partial w_i} \ln(\pi(a_t, W, x_t))
\]

Eligibility Trace:

\[
D_i(t) = \epsilon_i(t) + \beta D_i(t - 1),
\]

\[
\Delta w_i(t) = \text{(TD-error)} D_i(t)
\]

\(W \leftarrow W + \alpha_p \Delta W(t)\),

where \(w_i\) denotes the \(i\)th component of \(W\), \(\epsilon_i\) and \(D_i\) are the associated eligibility and eligibility trace respectively, \(\beta (0 \leq \beta < 1)\) is a discount factor for the eligibility trace, \(\alpha_p\) is the learning rate for the actor.
4. The critic updates estimated value function \(\hat{V}(x)\) according to TD methods. e.g., TD(0) algorithm adjusts \(\hat{V}(x) \leftarrow \hat{V}(x) + \alpha \text{(TD-error)}\), where \(\alpha\) is the learning rate.
5. Go to step 1.

Figure 3: The actor/critic algorithm adding the eligibility trace to the actor.

in Figure 3 updates the policy parameters as:

\[
\sum_{t=0}^{\infty} \Delta w_i(t) = \sum_{t=0}^{\infty} \left( r_t + \gamma \hat{V}(x_{t+1}) - \hat{V}(x_t) \right) D_i(t)
\]

\[
= \sum_{t=0}^{\infty} \left( r_t + \gamma \hat{V}(x_{t+1}) - \hat{V}(x_t) \right) \left( \sum_{t=0}^{\infty} \gamma^{t-t} \epsilon_j(t) \right)
\]

\[
= \sum_{t=0}^{\infty} \epsilon_j(t) \left( \sum_{t=0}^{\infty} \gamma^{t-t} r_t + \gamma \hat{V}(x_{t+1}) - \hat{V}(x_t) \right)
\]

\[
= \sum_{t=0}^{\infty} \epsilon_j(t) \left( \sum_{t=0}^{\infty} \gamma^{t-t} r_t - \hat{V}(x_t) \right)
\]

\[
= \sum_{t=0}^{\infty} \epsilon_j(t) \left( V_t - \hat{V}(x_t) \right)
\]

Equation 5 is given by Equation 1 and 4. Here we assume that the statistics of the random variable \(V_t\) depends only on the current policy parameter. It means \(E\{V_t\}\) is a deterministic function of \(W\), where \(E\) de-
notes the expectation operator. This assumption may be right if the policy is converged to an equilibrium point. The critic’s estimation \( V(x_t) \) is obviously independent of the action at the time \( t \). From the theory of Williams’ REINFORCE algorithm [Williams 92], the value \( V_t \) and \( V(x_t) \) in Equation 5 can be seen as a reinforcement signal and a reinforcement baseline respectively, then we have \( E\{\epsilon_i(t)(V_t - V(x_t))\} = (\partial/\partial x_t)E\{V_t\} \). It says that the algorithm updates policy parameters statistically in a direction for increasing the actual return \( V_t \), not in a direction of a gradient of estimated value function in the critic. Also it can be seen as an extension of reinforcement comparison methods [Sutton et al. 98], then \( V(x_t) \) corresponds to the reference reward.

From the above analysis and Figure 3, we can explain what the actor’s eligibility trace does. At the time \( t \), the algorithm reinforces \( a_t \) using TD error \( r_t + V(x_{t+1}) - V(x_t) \) as a temporary expedient, thereafter the actor’s eligibility trace replaces \( V(x_{t+1}) \) with the actual return \( (r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots) \) in order.

The critic does not affect the direction of the average update vector, because the critic works as a reinforcement baseline. Therefore, the actor can improve its policy, whether the critic is able to learn the value function or not. If the critic approximates the value function well, the actor’s learning would be accelerated.

The above results are under the special condition \( \beta = \gamma \). If \( \beta = 0 \), the actor updates \( W \) in the direction of the gradient of the approximated value function in the critic. The \( \beta \) \((0 < \beta < \gamma)\) interpolates between the above two limiting cases. The characteristics of the \( \beta \) are similar to the \( \lambda \) in TD(\( \lambda \)) [Sutton 88] and Q(\( \lambda \))-learning [Peng et al. 94].

5 Preliminary Experiments

This section demonstrates the performance of the algorithm applying to a simple linear control problem.

5.1 A Linear Quadratic Regulator (LQR)

The following linear control problem can serve as a benchmark of delayed reinforcement tasks [Baird 94]. At a given discrete-time \( t \), the state of the environment is the real value \( x_t \). The agent chooses a control action \( a_t \), that is also real value. The dynamics of the environment is:

\[ x_{t+1} = x_t + a_t + \text{noise}, \]

where the \text{noise} is the normal distribution that follows the standard deviation \( \sigma_{\text{noise}} = 0.5 \). The immediate reward is given by

\[ r_t = -x_t^2 - a_t^2. \]  \( (7) \)

The goal is to maximize the total discounted reward, defined by Equation 1 or 2 for all \( x \). Because the task is a linear quadratic regulator (LQR) problem, it is possible to calculate the optimal control rule. From the discrete-time Riccati equation, the optimum regulator is given by

\[ a_t = -k_1 x_t, \quad \text{where} \quad k_1 = 1 - \frac{2}{1 + 2\gamma + 4\gamma^2 + 1}. \]  \( (8) \)

The optimum value function is given by \( V^*(x_t) = -k_2 x_t^2 \), where \( k_2 \) is a some positive constant. In this experiment, the set of possible states is constrained to lie in the range \([-4, 4]\). When the state transition given by Equation 6 does not result in the range \([-4, 4]\), the \( x_t \) is truncated. When the agent chooses an action that is not lie in the range \([-4, 4]\), the action executed in the environment is also truncated.

5.2 Implementation for the LQR Problem

5.2.1 The Actor

Remember the policy \( \pi(a, W, X) \) is a probability density function when the set of possible actions is continuous. The normal distribution is a simple multiparameter distribution for a continuous random variable. It has two parameters, the mean \( \mu \) and the standard deviation \( \sigma \). When the policy function \( \pi \) is given by the equation 9, the eligibility of \( \mu \) and \( \sigma \) are

\[ \pi(a, \mu, \sigma) = \frac{1}{\sigma \sqrt{2\pi}} \exp\left(-\frac{(a - \mu)^2}{2\sigma^2}\right) \]  \( (9) \)

\[ \epsilon_\mu = \frac{a_t - \mu}{\sigma^2} \]  \( (10) \)

\[ \epsilon_\sigma = \frac{(a_t - \mu)^2 - \sigma^2}{\sigma^2} \]  \( (11) \)

One useful feature of such a Gaussian unit [Williams 92] is that the agent has a potential to control its degree of exploratory behavior. We must draw attention to the fact that the eligibility is to divergent when \( \sigma \) goes close to 0, because the parameter \( \sigma \) is occupying the denominators of Equation 10 and 11. The divergence of the eligibility has a bad influence on the algorithm. One way to overcome this problem is to control the step size of the update parameter vector using \( \sigma \). It is obtained by setting the learning rate parameter proportional to \( \sigma^2 \), then the eligibility can be be seen as

\[ \epsilon_\mu = a_t - \mu, \quad \epsilon_\sigma = \frac{(a_t - \mu)^2 - \sigma^2}{\sigma}. \]  \( (12) \)

The actor would first compute \( \mu \) and \( \sigma \) deterministically and then draw its output from the normal distribution that follows mean equal to \( \mu \) and standard
deviation equal to σ. The actor has two internal variables, \( w_1 \) and \( w_2 \), and computes the values of \( \mu \) and \( \sigma \) according to

\[
\mu = w_1 x_t, \quad \sigma = \frac{1}{1 + \exp(-w_2)}.
\]

(13)

Then, \( w_1 \) can be seen as a feedback gain. The reason for this calculation of \( \sigma \) is to guarantee the \( \sigma \) to keep positive. The \( e_1 \) and \( e_2 \) are the characteristic eligibilities of \( w_1 \) and \( w_2 \) respectively. From Equation 12, \( e_1 \) and \( e_2 \) are given by

\[
e_1 = e_p \frac{\partial}{\partial w_1} \mu = (a_t - \mu) x_t,
\]

(14)

\[
e_2 = e_e \frac{\partial}{\partial w_2} \sigma = ((a_t - \mu)^2 - \sigma^2)(1 - \sigma).
\]

(15)

The \( w_1 \) is initialized to 0.35 ± 0.15, and \( w_2 = 0 \), i.e., \( \sigma = 0.5 \). The learning rate \( \alpha_p \) is fixed to 0.001.

5.2.2 The Critic

The critic quantizes the continuous state-space \((-4 \leq x \leq 4)\) into an array of boxes. We have tried two types of the quantizing: one is discretizing \( x \) evenly into 3 boxes, the other is 10 boxes. The critic attempts to store in each box a prediction of the value \( V \) by using TD(0) [Sutton 88]. The learning rate \( \alpha \) for TD(0) is fixed to 0.2.

5.3 Simulation Results

Figure 4, 5, 6, 7 and 8 show the performance of 100 trials in the LQR problem with the discount rate \( \gamma = 0.9 \).

Figure 4 shows the performance of the algorithm, in which the critic uses 3 boxes, the actor does not use eligibility traces, i.e., \( \beta = 0 \). Figure 6 shows the performance where the critic uses 10 boxes, the actor does not use the traces. The algorithm in Figure 6 converged close to the optimum feedback gain. In contrast, Figure 4 didn’t. The reason for this is that the ability of the function approximation (3 boxes) is insufficient for learning policy without the trace.

Figure 5 shows the performance where the critic uses 3 boxes, the actor uses the trace, \( \beta = \gamma = 0.9 \). It achieved much better results in terms of both the learning efficiency and the quality of the mean value of the converged policy than the algorithm in Figure 4 or 5. Obviously, the actor’s eligibility trace relates these two advantages. The reason for the learning efficiency in this case may be that the actor’s trace accelerates propagating information. The better quality of the policy is clearly owing to the property that the actor improves its policy by using a gradient of actual return, shown in Section 4.3. Therefore, the algorithm using the trace was not influenced by the critic’s ability in terms of the quality of the mean of the policy. We can also see this property in Figure 8, but its deviation is considerably large. Figure 9 shows the value function that is defined by Equation 1 and 7 over the parameter space \( \mu \) and \( \sigma \). The value of performance is fairly flat around the optimal solution. This is the reason that the deviation of the policy is large in Figure 8. This example makes it clear that the critic controls step-size of the actor’s backups so that the step-size is taken to be smaller around the local maximum.

The algorithm in Figure 7 achieved best results in terms of both the mean and the deviation of the policy. The reason for this may be owing to the critic’s perfect value estimation.

In this preliminary experiment, we can see that the algorithm using the actor’s eligibility trace performed better than the algorithm without using the trace in the same computational resources.

Here we presented the results of the actor-critic that use only TD(0) in the critic, but we have also experimented on TD(\( \lambda \)) where \( 0 < \lambda \leq 1 \). Roughly speaking, we have poor performance when the \( \lambda \) approaches close to 1. It follows from this that the eligibility trace in the critic cannot make up for the critic’s poor ability of function approximation. The details of the experiments using TD(\( \lambda \)) will appear in other papers.

Figure 4: The average performance of 100 trials without the actor’s eligibility trace (\( \beta = 0 \)). The critic uses 3 boxes.
Figure 5: The average performance of 100 trials using the actor’s trace $\beta = 0.9$. The critic uses 3 boxes.

Figure 6: The average performance of 100 trials without the actor’s trace ($\beta = 0$). The critic uses 10 boxes.

Figure 7: The average performance of 100 trials using the actor’s trace $\beta = 0.9$. The critic uses 10 boxes.

Figure 8: The average performance of 100 trials. $\beta = 0.9$. The agent learns without the critic, i.e., the critic provides $V(x) = 0$ for all $x$. 
6 Applying to a Cart-Pole Problem

The behavior of this algorithm is demonstrated through a computer simulation of a cart-pole control task, that is a multi-dimensional nonlinear non-quadratic problem. We modified the cart-pole problem described in [Barto et al. 83] so that the action is taken to be continuous.

6.1 Problem Formulation

The dynamics of the cart-pole system is modeled by

\[
\ddot{\theta} = \frac{g \sin \theta + \cos \theta \left( -\frac{F - m \ell \dot{\theta}^2 \sin \theta + \mu \dot{\theta}}{M + m} \right) - \frac{\mu \dot{\theta}}{m}}{\ell},
\]

\[
\ddot{x} = \frac{F + ml \left( \dot{\theta}^2 \sin \theta - \ddot{\theta} \cos \theta \right) - \mu \dot{\theta} \text{sgn}(\dot{x})}{M + m},
\]

where \( M = 1.0 \) (kg) denotes mass of the cart, \( m = 0.1 \) (kg) is mass of the pole, \( 2\ell = 1 \) (m) is a length of the pole, \( g = 9.8 \) (m/sec\(^2\)) is the acceleration of gravity, \( F \) (N) denotes the force applied to cart’s center of mass, \( \mu_c = 0.0005 \) is a coefficient of friction of cart, \( \mu_p = 0.000002 \) is a coefficient of friction of pole. In this simulation, we use discrete-time system to approximate these equations, where \( \Delta t = 0.02 \) sec. At each discrete time step, the agent observes \((x, \dot{x}, \theta, \dot{\theta})\), and controls the force \( F \). The agent can execute action in arbitrary range, but the possible action in the cart-pole system is constrained to lie in the range \([-20, 20]\) (N). When the agent chooses an action which is not lie in that range, the action executed in the system is truncated. The system begins with \((x, \dot{x}, \theta, \dot{\theta}) = (0, 0, 0, 0)\). The system fails and receives a reward (penalty) signal of \(-1\) when the pole falls over \( \pm 12\) degrees or the cart runs over the bounds of its track \((-2.4 \leq x \leq 2.4)\), then the cart-pole system is reset to the initial state.

6.2 Details of the Agent

In this experiment, the actor adopts similar implementation shown in Equation 9 and 12. The state space is constrained in the range \((x, \dot{x}, \theta, \dot{\theta}) = (\pm 2.4 \text{ m}, \pm 2 \text{ m/sec}, \pm \pi \times 12/180 \text{ rad}, \pm 1.5 \text{ rad/sec})\). The actor has five internal variables \(w_1 \cdots w_5\), and computes the \(\mu\) and \(\sigma\) according to

\[
\mu = \frac{w_1 x_1 + w_2 \dot{x}_1 + w_3 \frac{\theta_1}{12\pi/180} + w_4 \dot{\theta}_1}{1.5},
\]

\[
\sigma = 0.1 + \frac{1}{1 + \exp(-w_5)}. \tag{16}
\]

Similarly to Equation 14 and 15, the eligibilities \(e_1 \cdots e_5\) are given by

\[
e_1 = (a_1 - \mu) x_1, \quad e_2 = (a_1 - \mu) \dot{x}_1, \quad e_3 = (a_1 - \mu) \theta_1, \quad e_4 = (a_1 - \mu) \dot{\theta}_1, \quad e_5 = ((a_1 - \mu)^2 - \sigma^2)(1 + 0.1 - \sigma).
\]

The critic discretizes the normalized state space evenly into \(3 \times 3 \times 3 \times 3 = 81\) boxes, and attempts to store in each box \(V\) by using TD(0) algorithm [Sutton 88]. The parameters are set to \(\gamma = 0.95\), \(\alpha = 0.5\), \(\alpha_p = 0.001\).

6.3 Simulation Results

Figure 11 shows the performance of three learning algorithms in which the policy representation is the
same. The actor/critic algorithm using the actor’s trace achieved best results. In contrast, the algorithm without using the trace couldn’t learn the control policy because of the poor ability of function approximation in the critic.

To Overcome non-Markovian: There are many ways to implement the critic’s learning scheme. [Peng et al. 94] and [Sutton 95] pointed out that increasing $\lambda$ makes $TD(\lambda)$ less sensitive to non-Markovian effect. The actor’s eligibility traces are also useful in getting over non-Markovian problems [Kimura et al. 97]. Therefore, the combination of $TD(\lambda)$ and the actor’s eligibility trace will be robust to non-Markovian problems.

Combining with Efficient DP-based Methods: If the hidden state is relatively small in the state space, the agent may perform good in which efficient DP-based algorithms are adopted for the critic. The DP-based algorithms accelerate the actor’s learning in completely observable states, and the actor’s stochastic policy and its trace ($\beta = \gamma$) would make up for the non-Markovian effects owing to the hidden state or function approximation.

8 Conclusions

This paper presented an analysis of actor/critic algorithms in which the actor updates its policy using the eligibility trace of the policy parameters. The results show that when the discount rate of the value function equals the discount factor of the actor’s trace, the actor improves its policy by using a gradient of actual return, not by using a gradient of the estimated return in the critic. Then, the critic provides an adaptive reinforcement baseline to the actor controlling the step-size of the actor’s backups. It enables the agent to learn a fairly good policy under the condition that the approximated value function in the critic is hopelessly imperfect. The behavior is demonstrated through simulations showing that the trace contributes the learning efficiency and the suppression of undesirable drifts of the policy. Analysis of the algorithm in non-Markovian environments is a future work.

Acknowledgements

We would like to thank Andrew Barto, Jing Peng, Jeff Schneider, Satinder Singh, Richard Sutton, and reviewers for many helpful comments and suggestions.

References
