A Comparison of Eligibility Trace and Momentum on SARSA in Continuous State- and Action-Space

Barry D. Nichols
School of Science and Technology
Middlesex University
London, England
Email: b.nichols@mdx.ac.uk

Abstract—**Here the Newton's Method direct action selection approach to continuous action-space reinforcement learning is extended to use an eligibility trace. This is then compared to the momentum term approach from the literature in terms of the update equations and also the success rate and number of trials required to train on two variants of the simulated Cart-Pole benchmark problem. The eligibility trace approach achieves a higher success rate with a far wider range of parameter values than the momentum approach and also trains in fewer trials on the Cart-Pole problem.**

I. Introduction

Reinforcement learning (RL) [1] has had many successes, however, when the action-space is continuous it becomes difficult to select the greedy action. One approach to overcoming this problem is by directly optimising the value function [2], [3].

A popular approach to accelerating training in RL is to apply an eligibility trace, but in [2], [3] a momentum term was applied to the update of the value function. When the update equations of these two techniques are compared they are very similar. Here, the update equations of both eligibility and momentum based approached are compared, then both approaches are applied to the well known simulated Cart-Pole benchmark problem and the double pole variant for comparison, both in terms of trials to train and sensitivity to parameter values.

The rest of this paper is organised as follows. Section II gives the background about reinforcement learning and the use of eligibility in continuous action-space. The details of the approach applied here are described in Section III. Then, Section IV explains the details of the experiment carried out, followed by the results (Section V). Finally, the conclusion and future work are in Section VI.

II. Background

A. Reinforcement Learning

An RL agent learns to perform a task by experimenting on that task and receiving feedback based on its current performance. At any given time-step, the agent is in state $s_t \in \mathcal{S}$ and must select an action $a_t \in \mathcal{A}$. After applying a_t to the environment, the agent receives a reward $r_{t+1} \in \mathbb{R}$ from the environment and transitions to $s_{t+1} \in \mathcal{S}$.

In order to improve its performance on the task, the RL agent seeks to maximise the sum of long-term discounted rewards it receives

$$\sum_{t=0}^{T-1} \gamma^t r_{t+1}, \tag{1}$$

where t is the time-step; r_{t+1} is the reward received in time-step $t+1$, after taking a_t from s_t; and $\gamma \in (0, 1]$ is the discount rate. Normally $\gamma < 1$ to ensure the sum of rewards remains finite and to assign more weight to rewards received sooner.

The optimisation of the sum of discounted rewards is achieved by the construction of a value function $Q : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, which approximates the expected sum of long term rewards when taking a from s and then following the learned policy $\pi : \mathcal{S} \to \mathcal{A}$. The learned policy is a mapping of which action to take from a given state. The Q function is defined as the sum of the immediate reward and the expected sum of rewards which will be received resulting from taking the next action from the next state and can be recursively defined as

$$Q(s_t, a_t) = r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}). \tag{2}$$

The learnt policy involves selecting the greedy action w.r.t. the value function. In small, discrete action-space the agent can select the greedy action a^* using

$$a^* = \arg\max_a Q(s, a), \quad \forall a \in \mathcal{A}. \tag{3}$$

The agent must also perform exploratory actions in order to update the value function in unexplored areas of the state- and action-space, which may lead to an improved policy. There are several exploration methods commonly applied, in the discrete action-space this may involve selecting a random action with a small probability or selecting actions proportionately to their value. These methods are known as ϵ-greedy and Softmax exploration respectively [1].

The value function can be updated on-line by temporal difference (TD) methods, in which the TD error

$$\delta_t = r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t) \tag{4}$$

is minimised. In the SARSA algorithm [1] this is achieved by using δ_t to update the Q using

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \delta_t, \tag{5}$$

where α is a step-size parameter.

B. *Continuous State- and Action-Space*

When the state-space is continuous it is not possible to store Q values in a lookup table; therefore, function approximation is often applied to approximate the Q function. However, when the action-space is also continuous it is impossible to compare the values of all actions, thus alternative action selection methods must be employed.

Possible approaches to overcoming the action selection problem in continuous action-space include applying a function approximator to approximate the policy function $\pi : \mathcal{S} \to \mathcal{A}$, which can be used alone [4] or together with a value function [5]; or by directly optimising the Q function [2], [6].

The approximation of the Q function can be achieved by an artificial neural network (ANN). When an ANN is employed to approximate the value function, the update is then applied to the parameters of the ANN (the synaptic weights) rather than directly updating the value associated with the specific state and action. The update equation is then

$$\vec{\theta} \leftarrow \vec{\theta} + \alpha \delta_t \nabla_{\vec{\theta}} Q(s_t, a_t), \tag{6}$$

where $\vec{\theta}$ is the parameter vector for the ANN.

In the continuous action-space setting exploration is often applied through Gaussian exploration [5], whereby a Gaussian random variable is added to the greedy action before applying it to the environment. This is more appropriate than ϵ-greedy and Softmax when the action-space is continuous.

C. *Eligibility Trace*

In the standard version of the TD update (5) only the Q values of a single state and action will be updated. In order to also allocate credit to previous states and actions which led to the reward an eligibility trace can be applied [1]. This is achieved by assigning an eligibility value to states and actions as they are encountered which is reduced as time progresses.

When function approximation is applied to the Q function rather than a lookup table the eligibility trace is a vector $\vec{e}$, where each element of $\vec{e}$ relates to one of the function approximator parameters rather than to an entry in a lookup table. The parameter update with eligibility trace is

$$\vec{\theta} \leftarrow \vec{\theta} + \alpha \delta_t \vec{e}_t, \tag{7}$$

where δ_t is as (4),

$$\vec{e}_t = \gamma \lambda \vec{e}_{t-1} + \nabla_{\vec{\theta}} Q(s_t, a_t) \tag{8}$$

and $\vec{e}_0 = \vec{0}$.

III. Approach

The approach used here is based on [2], but uses an eligibility trace rather than a momentum term. As with [2], the Q function was approximated using a multi-layer perceptron, with 7 hidden nodes using a hyperbolic tangent activation function. The initial weights were uniformly selected in the range $[-0.2, 0.2]$. Inputs to the ANN were scaled to be within $[-1, 1]$ and outputs rescaled to be applied to the environment. The value function was updated using the SARSA algorithm with an eligibility trace.

Greedy actions were selected using the iterative Newton's Method approach $a = a - \frac{\nabla_a Q(s,a)}{\nabla_a^2 Q(s,a)}$, which was then constrained to be within $[-1, 1]$. A maximum of 10 iterations (15 for Double Cart-Pole) were applied, but this was terminated early if two iterations result in actions which differ by less than 0.0001. As Newton's Method is unable to distinguish between maxima and minima, this was repeated from several initial actions in $\{-1, -0.5, 0, 0.5, 1\}$. Actions were rescaled to the range of the dynamic system after being selected.

Although the original approach in [2] did not apply an eligibility trace it did apply a momentum term [7] to the ANN update. By rearranging the update equations of both approaches into a similar form we can see the differences and similarities more clearly.

Firstly the momentum update formula, the usual form is

$$\vec{\theta}_{t+1} = \vec{\theta}_t + \Delta \vec{\theta}_{t+1} \tag{9}$$

where

$$\Delta \vec{\theta}_{t+1} = \alpha \delta_t \nabla_{\vec{\theta}_t} Q(s_t, a_t) + \mu \Delta \vec{\theta}_t. \tag{10}$$

Which, taking into account $\Delta \vec{\theta}_0 = \vec{0}$, can be combined to form

$$\vec{\theta}_{t+1} = \vec{\theta}_t + \alpha \sum_{k=0}^{t} \mu^k \delta_{t-k} \nabla_{\vec{\theta}_{t-k}} Q(s_{t-k}, a_{t-k}). \tag{11}$$

Then, by substituting (4) and (8) into (7), and taking into account the fact that $\vec{e}_0 = \vec{0}$, we can arrive at a similar form to the momentum update (11) for the eligibility trace

$$\vec{\theta}_{t+1} = \vec{\theta}_t + \alpha \sum_{k=0}^{t} \gamma^k \lambda^k \delta_t \nabla_{\vec{\theta}_{t-k}} Q(s_{t-k}, a_{t-k}). \tag{12}$$

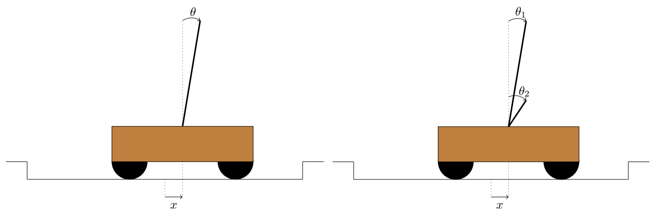
If we set $\mu = \gamma \lambda$ the only difference between eligibility trace (12) and momentum (11) is that eligibility trace uses the current TD error δ_t for each of the terms, whereas momentum uses a different TD error in each term: the TD error from the time-step of the term.

Intuitively, it makes sense to use the most recent knowledge (from rewards) in RL to update the previous actions which led to it, rather than the knowledge the agent had at the time-step after taking those actions. Similar conclusions have been drawn in [8] albeit on adaptive critic designs. Also, here the comparison includes the impact of parameter values on both approaches, which was not included in [8].

Therefore using an eligibility trace would be expected to give improved results over standard SARSA with a momentum term when updating the ANN. In the following experiments the momentum approach is also applied for comparison purposes.

IV. Experiment

The experiments here use two variants of the Cart-Pole problem. The Cart-Pole problem is a very well known control benchmark problem often used by the RL community [1], [9]–[11]. The problem consists of a cart on a limited track with a pole attached, but free to swing. The objective is to maintain the pole in a balanced position without reaching the ends of



(a) Cart-Pole (b) Double Cart-Pole

Fig. 1. The two Cart-Pole variants. The distance of the cart from the centre of the track is x and θ (θ_i) is the angle of the pole(s) from the vertical position.

the track. Often actions are limited to discrete values [11], here, however, continuous ranges are permitted. The double pole variant of the problem is also used, as described in [5].

Both problems were run for 1000 trials. A trial consists of a maximum of $120\,\mathrm{s}$, which is terminated immediately if the agent fails, i.e. the cart reaches the end of the track or the pole (one of the poles) falls. At each $0.02\,\mathrm{s}$ time-step, the agent selects an action, applies it to the environment (which is updated using the Runge Kutta fourth order method) and the agent receives the resulting state and the computed reward. The reward is -1 on failure and zero at all other times. The pole was considered to have fallen if $|\theta| > \pi/15$, and the cart was considered to have reached the edge of the track if $|x| > 2.4$.

Gaussian exploration was applied by selecting a random exploration value from the normal distribution with mean 0 and standard deviation 1. This exploration term was then multiplied by an exploration size parameter and added to the selected action. The exploration size started at 1 at the beginning of each trail and was reduced by 0.001 (0.01 for the Double Cart-Pole) at each time-step until it reached 0.

All experiments are repeated for 100 runs, and the mean average number of trials taken before the agent could successfully balance the pole(s) for the full simulated time was recorded. The percentage of runs where the agent was able to learn to successfully balance the pole was the success rate. This was then repeated for all values in $\{0, 0.1, \ldots, 1\}$ for γ and λ (μ for momentum) and for various values of α.

A. Cart-Pole

The state vector comprises the pole angle, pole angular velocity, cart distance from centre of track and cart velocity $s = [\theta, \dot{\theta}, x, \dot{x}]^{\mathsf{T}}$. The action is the force applied to the cart $a = F \in [-10, 10]\,\mathrm{N}$.

The initial state at the beginning of each episode was $[0 + o, 0, 0, 0]^{\mathsf{T}}$, where o was a uniformly randomly generated offset in the range $[-0.05, 0.05]$. The selected action, including any exploration, was limited to the allowable range before being applied to the simulation. The parameters used for the Cart-Pole simulation are given in Table I. The equations of motion used to update the environment, the same as those used in [11],

TABLE I
CART POLE PARAMETERS

Parameter	Value
Cart mass (m_c)	$1\,\mathrm{kg}$
Pole mass (m)	$0.1\,\mathrm{kg}$
Gravitational constant (g)	$9.81\,\mathrm{m/s^2}$
Half pole length (l)	$0.5\,\mathrm{m}$
Cart friction (μ_c)	5×10^{-4}
Pole friction (μ_p)	2×10^{-6}
Time increment (Δ_t)	$0.02\,\mathrm{s}$
Maximum force (F_{max})	$10\,\mathrm{N}$

TABLE II
DOUBLE CART-POLE PARAMETERS

Parameter	Value
Cart mass (m_c)	$1\,\mathrm{kg}$
Pole one mass (m_1)	$0.1\,\mathrm{kg}$
Pole two mass (m_2)	$0.01\,\mathrm{kg}$
Gravitational constant (g)	$9.81\,\mathrm{m/s^2}$
Pole one length (l_1)	$1\,\mathrm{m}$
Pole two length (l_2)	$0.1\,\mathrm{m}$
Cart friction (μ_c)	5×10^{-4}
Pole one friction (μ_1)	2×10^{-6}
Pole two friction (μ_2)	2×10^{-6}
Time increment (Δ_t)	$0.02\,\mathrm{s}$
Maximum force (F_{max})	$40\,\mathrm{N}$

are given by

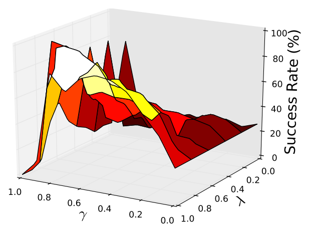
$$\begin{aligned} \phi &= -F - ml\dot{\theta}^2 \sin\theta + \mu_c \operatorname{sgn}(\dot{x}), \\ \ddot{\theta} &= \frac{g\sin\theta + \phi\cos\theta - \frac{\mu_p\dot{\theta}}{ml}}{l\left(\frac{4}{3} - \frac{m\cos^2\theta}{m_c+m}\right)}, \\ \ddot{x} &= \frac{F + ml\left(\dot{\theta}^2\sin\theta - \ddot{\theta}\cos\theta\right) - \mu_c \operatorname{sgn}(\dot{x})}{m_c + m}. \end{aligned} \tag{13}$$

B. Double Cart-Pole

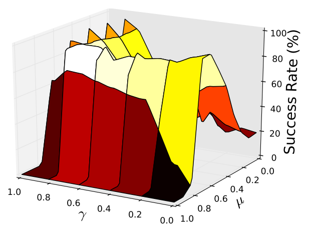
The double Cart-Pole problem is a variation of the standard Cart-Pole problem, whereby the cart has two poles of differing lengths, both of which must be balanced. The parameters used for the double Cart-Pole simulation are given in Table II. The equations of motion in this experiment were the same as that of [5], which are

$$\begin{aligned} \phi &= F - \mu_c \operatorname{sgn}(\dot{x}), \\ \ddot{x} &= \frac{\phi + \sum_{i=1}^{2} 2m_i\theta_i^2 \sin\theta_i + \frac{3}{4}m_i\cos\theta_i\left(2\frac{\mu_i\dot{\theta}_i}{m_i l_i} + g\sin\theta_i\right)}{m_c + \sum_{i=1}^{2} m_i\left(1 - \frac{3}{4}cos^2\theta_i\right)}, \\ \ddot{\theta}_i &= -\frac{3}{8l_i}\left(\ddot{x}\cos\theta_i + g\sin\theta_i + \frac{\mu_i\dot{\theta}_i}{m_i l_i}\right). \end{aligned} \tag{14}$$

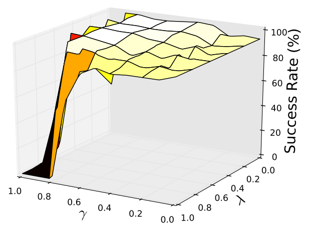
The state vector comprised the angle and angular velocity of each pole; cart distance from centre of track; and cart velocity $s = [\theta_1, \dot{\theta}_1, \theta_2, \dot{\theta}_2, x, \dot{x}]^{\mathsf{T}}$, and the action was the force applied to the cart $a = F \in [-40, 40]\mathrm{N}$. The initial state for each episode was $[\frac{\pi}{180}, 0, 0, 0, 0, 0]^{\mathsf{T}}$ (as [5]).



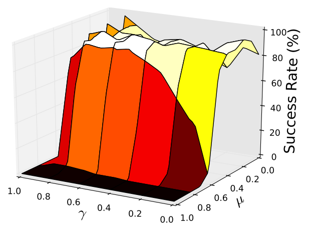
(a) Eligibility Trace $\alpha = 0.1$



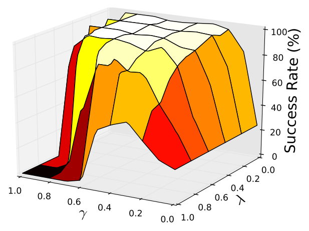
(b) Momentum $\alpha = 0.1$



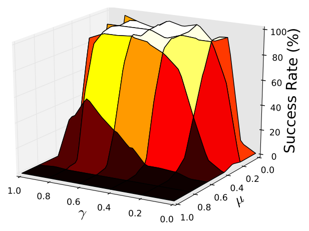
(c) Eligibility Trace $\alpha = 0.2$



(d) Momentum $\alpha = 0.2$

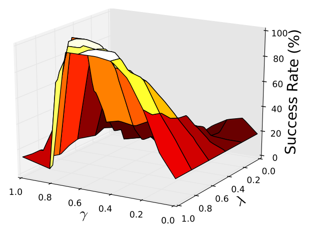


(e) Eligibility Trace $\alpha = 0.3$

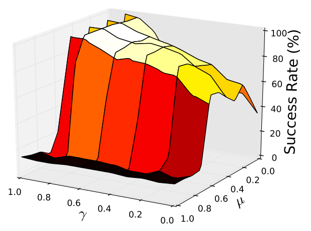


(f) Momentum $\alpha = 0.3$

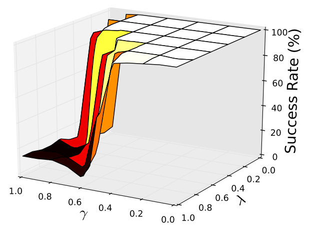
Fig. 2. Cart-Pole success rate for various parameter values.



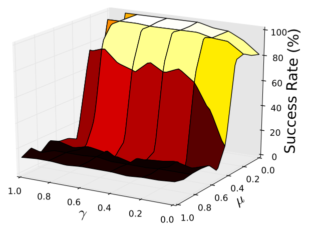
(a) Eligibility Trace $\alpha = 0.1$



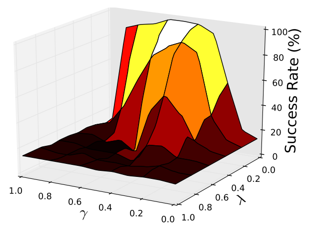
(b) Momentum $\alpha = 0.1$



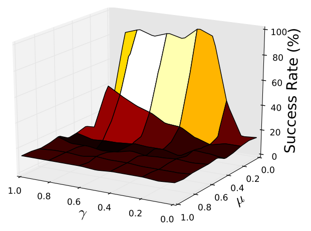
(c) Eligibility Trace $\alpha = 0.2$



(d) Momentum $\alpha = 0.2$



(e) Eligibility Trace $\alpha = 0.3$



(f) Momentum $\alpha = 0.3$

Fig. 3. Double Cart-Pole success rate for various parameter values.

V. Results

Figure 2 shows the success rate on the Cart-Pole problem of both approaches for $\alpha \in \{0.1, 0.2, 0.3\}$. When $\alpha \geq 0.4$ the region with high success rate continued to shrink. Each subplot shows the success rate as a function of two other parameters: in the eligibility trace approach this was γ and λ, and in the momentum approach was γ and μ. Figure 3 shows the same information for the double Cart-Pole problem. In both problems $\alpha = 0.2$ had a high success rate for the widest range of parameter values, with the exception of the Cart-Pole problem with momentum where $\alpha = 0.3$ provided a higher success rate.

Table III shows, for each approach on each problem, the percentage of runs in which the RL agent succeeded, i.e. was able to balance the pole(s) for the full 120 s, and the mean average number trials it took the agent to succeed. The parameter values used to produce these results for the Cart-Pole were eligibility trace: $\alpha = 0.2$, $\gamma = 0.9$ and $\lambda = 0.5$; momentum: $\alpha = 0.3$, $\gamma = 0.8$ and $\mu = 0.3$. For the Double Cart-Pole they were eligibility trace: $\alpha = 0.2$, $\gamma = 0.6$ and $\lambda = 0.3$; momentum: $\alpha = 0.2$, $\gamma = 0.9$ and $\mu = 0.2$.

The eligibility trace approach achieved a higher success rate for a wider range of parameter values than the momentum approach. Also, on the Cart-Pole problem eligibility trace achieved training in fewer trials. On the double Cart-Pole the momentum approach did achieve training in less trials than with an eligibility trace, but as with the Cart-Pole problem, the range of parameters for which a high success rate could be achieved was significantly smaller than with the eligibility trace.

TABLE III
Results

Problem	Approach	Success Rate	Trials to Train
Cart-Pole	Eligibility Trace	100%	49.56
Cart-Pole	Momentum	98%	67.398
Double Cart-Pole	Eligibility Trace	100%	94.4
Double Cart-Pole	Momentum	100%	55

VI. Conclusion

Here the SARSA(λ) algorithm with direct action selection using Newton's Method was applied to the single and double pole variants of the well known Cart-Pole problem. The success rates are compared for various parameter values, and the approach is compared to a previous approach using a momentum term. The eligibility trace approach succeeds more reliably for a wider range of parameter values. Moreover, it achieves training in less trials on the Cart-Pole problem, which requires generalisation (due to the random starting position).

Future work should apply this approach to a wider range of problems and should also apply noise to the selected actions.

References

[1] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, Massachusetts: MIT Press, 1998.

[2] B. D. Nichols and D. C. Dracopoulos, "Application of Newton's Method to action selection in continuous state-and action-space reinforcement learning," in *European Symposium on Artificial Neural Networks*, 2014, pp. 141–146.

[3] B. D. Nichols, "A comparison of action selection methods for implicit policy method reinforcement learning in continuous action-space," in *Neural Networks (IJCNN), 2016 International Joint Conference on*. IEEE, 2016, pp. 3785–3792.

[4] I. Grondman, L. Busoniu, G. A. D. Lopes, and R. Babuska, "A survey of actor-critic reinforcement learning: Standard and natural policy gradients," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 42, no. 6, pp. 1291–1307, 2012.

[5] H. van Hasselt, "Reinforcement learning in continuous state and action spaces," in *Reinforcement Learning*, ser. Adaptation, Learning, and Optimization, M. Wiering and M. Otterlo, Eds. Springer Berlin Heidelberg, 2012, vol. 12, pp. 207–251.

[6] J. C. Santamarí, R. S. Sutton, and A. Ram, "Experiments with reinforcement learning in problems with continuous state and action spaces," *Adaptive behavior*, vol. 6, no. 2, pp. 163–217, 1997.

[7] S. Haykin, *Neural Networks and Learning Machines*, 3rd ed. Prentice Hall, November 2009.

[8] J. Xu, F.-M. Liang, and W.-S. Yu, "Learning with eligibility traces in adaptive critic designs," in *Vehicular Electronics and Safety, 2006. ICVES 2006. IEEE International Conference on*. IEEE, 2006, pp. 309–313.

[9] M. Riedmiller, J. Peters, and S. Schaal, "Evaluation of policy gradient methods and variants on the cart-pole benchmark," in *Approximate Dynamic Programming and Reinforcement Learning, 2007. ADPRL 2007. IEEE International Symposium on*, April 2007, pp. 254–261.

[10] L. C. Baird and H. Klopf, "Reinforcement learning with high-dimensional, continuous actions," Wright Laboratory, Tech. Rep., 1993.

[11] J. Si and Y.-T. Wang, "Online learning control by association and reinforcement," *Neural Networks, IEEE Transactions on*, vol. 12, no. 2, pp. 264–276, March 2001.