Temporal Difference Updating without a Learning Rate

ANU

Marcus Hutter

Canberra, ACT, 0200, Australia

http://www.hutter1.net/

NICTA

Summary

- Reinforcement learning TD update: $V_s^{t+1} = V_s^t + E_s^t \beta_t(s, s_{t+1})(r_t + \gamma V_{s_{t+1}}^t - V_{s_t}^t)\ \forall s$
- We derived learning rate β_t and eligibility trace E_s^t from statistical principles.
- In every setting that we have tested, superior performance & fewer parameters to tune.

IDSIA

Shane Legg

6900, Lugano, Switzerland

http://www.vetta.org/shane/

IDSIA SUPSI USI

Introduction & Background

Abstract

In the field of reinforcement learning, temporal difference (TD) learning is perhaps the most popular way to estimate the future discounted reward of states. We derive an equation for TD learning from statistical principles. Specifically, we start with the variational principle and then bootstrap to produce an updating rule for discounted state value estimates. The resulting equation is similar to the standard equation for temporal difference learning with eligibility traces, so called TD(λ), however it lacks the parameter α that specifies the learning rate. In the place of this free parameter there is now an equation for the learning rate that is specific to each state transition. We experimentally test this new learning rule against TD(λ) and find that it offers superior performance in various settings. Finally, we combine our update equation with both Watkin's Q(λ) and Sarsa(λ) and find that it again offers superior performance without a learning rate parameter.

Temporal Difference Learning

- s_k=state and r_k=reward in cycle k, $1 > \gamma$=discount.
- Value of state s = expected future discounted reward =

$$\begin{aligned}\overline{V}_s &:= \mathbf{E}\{r_k + \gamma r_{k+1} + \gamma^2 r_{k+2} + \cdots | s_k = s\}\\ &= \mathbf{E}\{r_k + \gamma \overline{V}_{s_{k+1}} | s_k = s\}\end{aligned} \qquad (1)$$

- Given: History of states $s_1, s_2, \ldots, s_t$, and observed rewards $r_1, r_2, \ldots, r_t$.
- Goal: Compute estimate V_s^t of $\overline{V}_s\ \forall s$.
- Equation (1) suggests the following TD(0) learning algorithm

$$V_{s_t}^{t+1} := V_{s_t}^t + \alpha\left(r_t + \gamma V_{s_{t+1}}^t - V_{s_t}^t\right),$$

- where α is a parameter that controls the rate of learning.
- Shortcoming: At each time t, the value of only the last state s_t is updated.

Eligibility Traces

- Improvement: TD with eligibility traces:
- update value of all recently visited states proportional to their ...
- Eligibility: $E_s^t := \gamma\lambda E_s^{t-1} + \delta_{s_t s}$
- λ controls the rate at which the eligibility trace is discounted.
- The TD(λ) update [Sutton 1988] is then, for all states s,

$$V_s^{t+1} := V_s^t + \alpha E_s^t\left(r + \gamma V_{s_{t+1}}^t - V_{s_t}^t\right) \quad \forall s. \qquad (2)$$

- TD makes intuitive sense and has various motivations [Sutton 1988], but is heuristic rather than a theoretically founded.

The New Update Equation

Main Novel Idea

- Idea: Derive a TD rule from statistical principles.
- Empirical future discounted reward of a state s_k at times $k = 1, \ldots, t$ is

$$v_k := \sum_{u=k}^{\infty} \gamma^{u-k} r_u,$$

- v_k is unknowable, since it depends also on unknown future ($u > t$) rewards.
- Goal: Estimate V_s^t close to $\overline{V}_s$.

$\Rightarrow$ we would like V_s to be close to v_k for all k such that $s = s_k$.

- Also: Discount old evidence by $\lambda \in (0, 1]$ in non-stationary environments:

$\Rightarrow$ Principle: minimize loss $L := \frac{1}{2}\sum_{k=1}^{t} \lambda^{t-k}(v_k - V_{s_k}^t)^2$

- For stationary environments we may simply set $\lambda = 1$ a priori.

Derivation of the Value Estimate

$$\frac{\partial L}{\partial V_s^t} = 0 \quad \Leftrightarrow \quad V_s^t N_s^t = \sum_{k=1}^{t} \lambda^{t-k} \delta_{s_k s} v_k \qquad (3)$$

- $N_s^t := \sum_{k=1}^t \lambda^{t-k}\delta_{s_k s}$ = discounted state visit counter.
- v_k has a self-consistency property: $v_k = \sum_{u=k}^{t-1} \gamma^{u-k} r_u + \gamma^{t-k} v_t$.
- Substituting this into (3) we get: $V_s^t N_s^t = R_s^t + E_s^t v_t$ (*)
- $E_s^t := \sum_{k=1}^t (\lambda\gamma)^{t-k}\delta_{s_k s}$ = the eligibility trace of state s,
- $R_s^t := \sum_{u=1}^{t-1} \lambda^{t-u} E_s^u r_u$ = discounted reward with eligibility.
- Bootstrap: Replace unknowable v_t in (*) by known $V_{s_t}^t$ and solve w.r.t. V_s^t:

Value estimate: $$V_s^t = \frac{1}{N_s^t}\left[R_s^t + E_s^t \frac{R_{s_t}^t}{N_{s_t}^t - E_{s_t}^t}\right] \qquad (4)$$

Incremental Update Rule

Rewrite (4) as incremental update rule HL(λ):

- $V_s^{t+1} = V_s^t + E_s^t \beta_t(s, s_{t+1})\left(r_t + \gamma V_{s_{t+1}}^t - V_{s_t}^t\right) \quad \forall s$ (**)
- $\beta_t(s, s_{t+1}) := \dfrac{1}{N_s^t} \cdot \dfrac{1}{1 - \gamma E_{s_{t+1}}^t / N_{s_{t+1}}^t}$ = Learning rate.
- $N_s^{t+1} = \lambda N_s^t + \delta_{s_{t+1}s}, \quad E_s^{t+1} = \lambda\gamma E_s^t + \delta_{s_{t+1}s}$.
- $V_s^0 = N_s^0 = E_s^0 = R_s^0 = 0$.

Explanation:

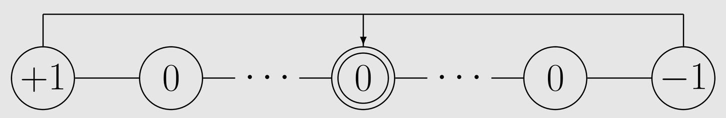
- (**) is usual update equation for TD learning with eligibility trace (2),
- but with heuristic hand-tuned learning rate parameter α
- replaced by exact/automatic/dynamic $\beta_t(s, s_{t+1})$.
- First term in β_t is inversely proportional to the state visit counter N_s^t.
- The second term in β_t has bounded fluctuation between 1 and $\frac{1}{1-\gamma}$. It is "large" for recently visited states. It converges to 1 if and only if $\lambda = 1$.

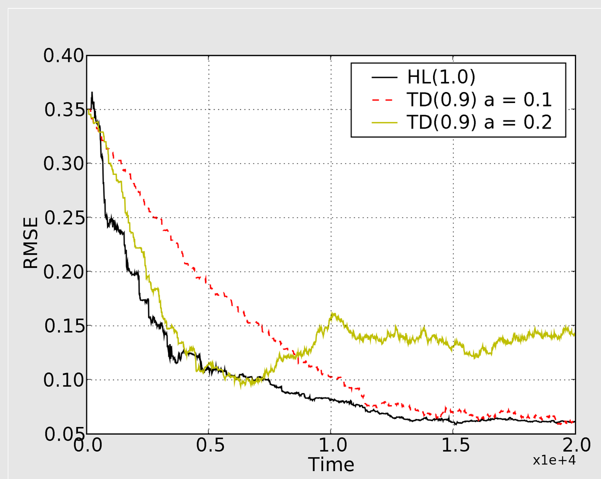
Experimental Evaluation

- Root mean square error (RMSE) between estimate V_s^t and exact value $\overline{V}_s$, averaged over s, are plotted.
- Best value of λ was used.

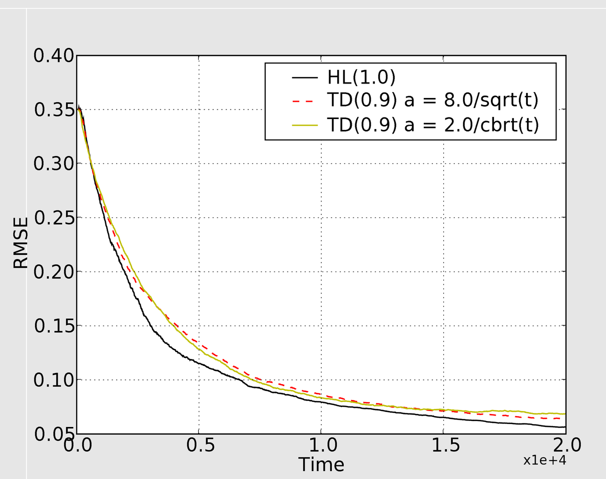
A simple Markov Process

- Linear Markov chain with
- 51 states. Start at $s = 25$.
- No reward, except ...
- Reward $+1$ at $s = 0$, reward -1 at $s = 50$. In both cases reset to $s = 25$.





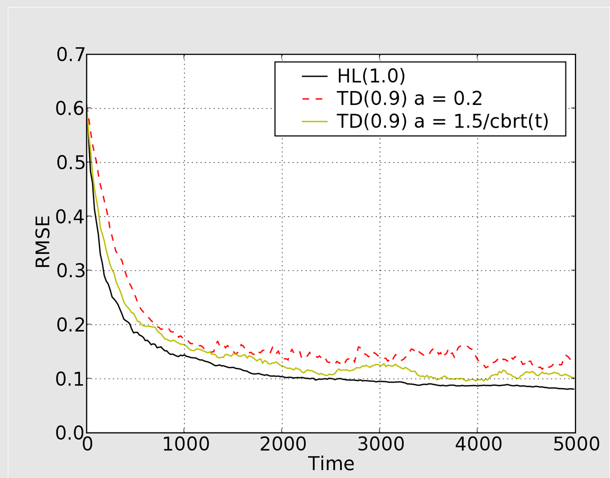
51 states. Discount $\gamma = 0.99$. Constant learning rate $a = \alpha$. HL(λ) signif. outperforms TD(λ).



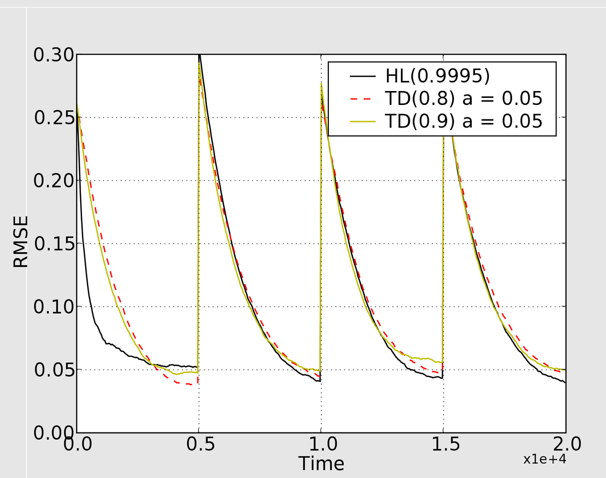
51 states. Variable learning rate α. HL(λ) somewhat better than TD(λ). ($\alpha_t \propto {}^1/_t$ very poor).

- Similar results for larger and smaller Markov chains, and different γ.

Random & Non-Stationary Markov Process



Random 50 state Markov process. Discount $\gamma = 0.9$.



21 state non-stationary Markov process, where reward function changed 5000 steps.

New Reinforcement Learning Algorithm

Algorithm for Markov Decision Processes

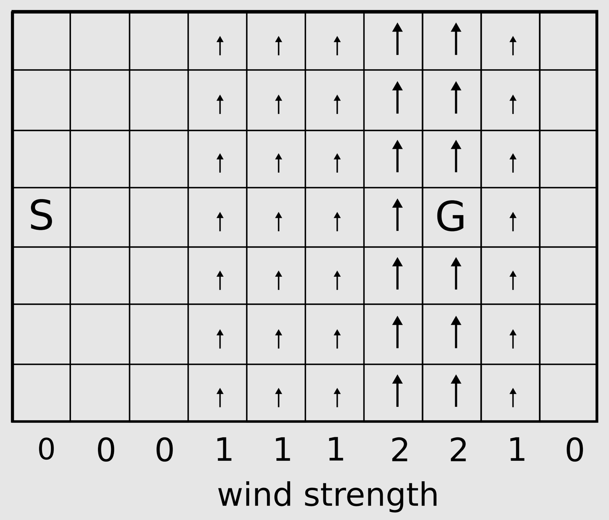
- Include actions: Markov process $\rightsquigarrow$ Markov Decision Process (MDP).
- TD(λ) $\rightsquigarrow$ Sarsa(λ) [Rummery 1994].
- Combining Sarsa(λ) with HL(λ) $\Rightarrow$ our HLS(λ) algorithm.

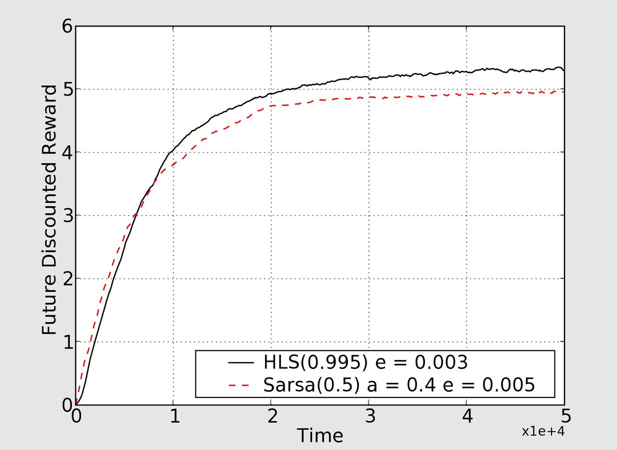
Algorithm HLS(λ)

Initialise $Q(s,a) = 0$, $N(s,a) = 1$ and $E(s,a) = 0$ for all s, a
Initialise s and a
repeat
  Take action a, observed r, s'
  Choose a' by using ϵ-greedy selection on $Q(s', \cdot)$
  $\Delta \leftarrow r + \gamma Q(s', a') - Q(s, a)$
  $E(s,a) \leftarrow E(s,a) + 1$
  $N(s,a) \leftarrow N(s,a) + 1$
  for all s, a **do**
    $\beta((s,a),(s',a')) \leftarrow \frac{1}{N(s,a)} \cdot \frac{1}{1 - \gamma E(s',a')/N(s',a')}$
  end for
  for all s, a **do**
    $Q(s,a) \leftarrow Q(s,a) + \beta\big((s,a),(s',a')\big)E(s,a)\Delta$
    $E(s,a) \leftarrow \gamma\lambda E(s,a)$
    $N(s,a) \leftarrow \lambda N(s,a)$
  end for
  $s \leftarrow s'; a \leftarrow a'$
until end of run

Windy Gridworld

- Move from start state S to goal state G on a windy grid [Sutton 1998].
- Once the goal state G is reached, the agent jumps back S and receives a reward of 1 ($\gamma = 0.99$).





Sarsa(λ) vs. HLS(λ)

- Despite considerable effort in tuning the parameters of Sarsa(λ), HLS(λ) consistently beats Sarsa(λ).
- Similar results for Q(λ) [Watkins 1989] but much smaller gap.

Conclusions

- We have mathematically derived the equation for setting the learning rate in temporal difference learning with eligibility traces.
- The equation replaces the learning rate α, which is a free parameter that normally has to be experimentally tuned by hand.
- In every setting that we have tested [(non)stationary Markov (Decision) processes], our new update equation has produced superior results.

$\Rightarrow$ superior performance and fewer parameters to tune