Temporal Learning - A Bounded Random Walk Problem (Sutton 1988)

Mikey Ling
Georgia Institute of Technology
mling7@gatech.edu

Abstract

TD(λ) is a proven yet misunderstood reinforcement learning algorithm that is capable of out-performing conventional supervised-learning methods and algorithms. This paper describes an experiment involving a bounded random walk problem and the TD(λ) method to attempt to verify performance and accuracy of the algorithm.

I. Bounded Random Walks

A bounded random walk problem is a sequence of states that's created by executing random steps to the right or left until a boundary (terminal state) is reached. For this specific experiment, there is a 50% chance of moving to the right or left. The state space is defined in the figure below (taken from Sutton 1988):

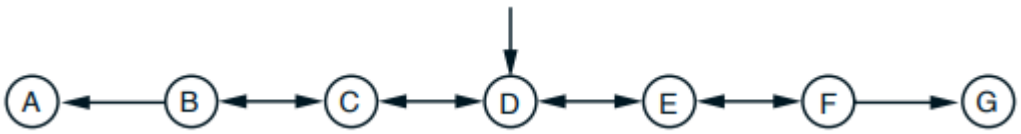

Figure 1- Visual Depiction of the Bounded Random Walk Generator

Every sequence starts in state D and ends in when either state A or G is reached. The TD(λ) learning method was applied to determine the probability the sequence results in a rightside termination (ending in state G). Each state sequence was modeled via binary vectors x_i, where the walk is in state i at time t. In Sutton's paper, state vectors of length five were used (states B through F); however, in our experiment all seven states were represented in these vectors because the addition of the terminal states will not affect the probabilities predicted of the middle five. The true probabilities for the middle five states were given: $\{\frac{1}{6}, \frac{1}{3}, \frac{1}{2}, \frac{2}{3}, \frac{5}{6}\}$, B to F, respectively.

II. Conducting the Experiment

The paramount equation in the experiment is referred to as Equation 4 in Sutton's paper (we, however, will refer to it as Equation 1 for bookkeeping purposes):

$$\Delta w_t = \alpha(P_{t+1} - P_t) \sum_{k=1}^{t} \lambda^{t-k} \nabla_w P_k \qquad (1)$$

P_t is the prediction at time t, w_t is the weight vector, α is the learning rate, and $\nabla_w P_k$ is the partial derivative of P_k with respect to w. However, this experiment can be considered a special case in which P_t is a linear function of x_t and w. So $P_t = w^T x_t = \sum_i w(i)x(i)$. With this mathematical equation for P_t, the partial derivate with respect to w is simply x_t.

We use Equation 1 to calculate Δw_t, the change in weights between temporal episodes. We must use another equation to calculate and update the actual values of the weight vectors:

$$w_t \leftarrow w + \sum_{t=1}^{m} \Delta w_t \tag{2}$$

(2) describes the update rule for the weight vector. Sutton uses this equation in the paper's experiments, but they are used at different times within the learning algorithm. For the first experiment, the weight vector is updated via (2) after the learner experiences an entire training set. For the second experiment, the weight vector is updated after each sequence is presented to the learner.

III. Experiment 1 – Reproducing Figure 3

In this experiment, the weight vector wasn't updated after each sequence presented to the learner; instead, the Δw's were accumulated over all the sequences in the training set and used to update the weights vector when the presentation of an entire training set was complete. The experiment featured 100 training sets, each containing 10 sequences. Each training set was presented to the learner until changes in the weight vector were no longer significant. For a small learning rate, α, the weight vector always converged. Sutton referred to this procedure as the *repeated presentations* training paradigm.

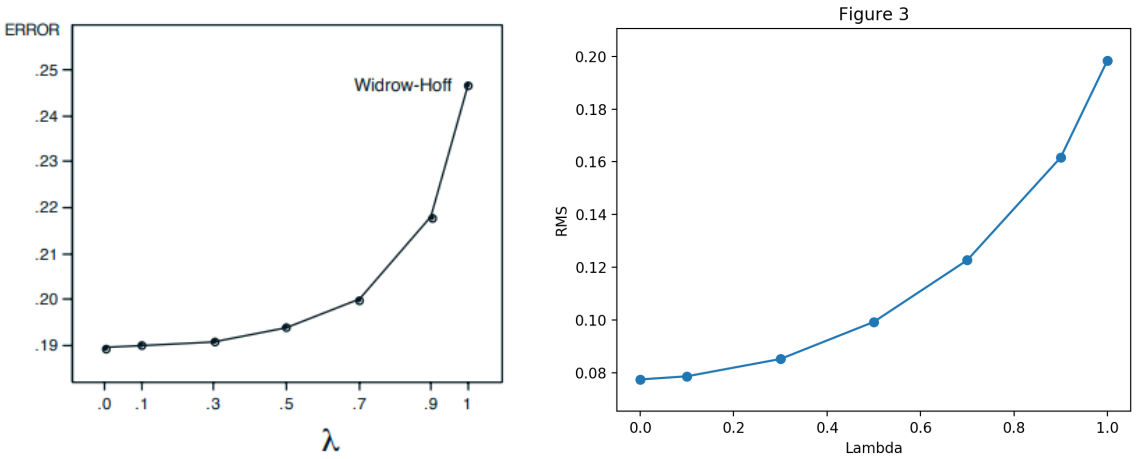

Figure 2 - Comparing Sutton's Original Experiment 1 (on the left) to Our Results (on the right)

Figure 2 shows a cross-comparison of our results and Sutton's for this first experiment. The overall shapes of the graphs are extremely similar; however, it's apparent our results yield, on average, an RMS error approximately 0.11 *lower* than Sutton's. There are, at least, two possible explanations for this increase in accuracy:

1.) The randomly generated state sequences simply led to a more accurate prediction from the learner.

2.) We held our learner to a “stricter” threshold for convergence than Sutton did when he conducted his experiment

The first possibility is less likely than the second possibility because the experiment was conducted multiple times, and, although there was some variance, the results always yielded a lower RMS compared to Sutton’s results. The second possibility is more likely because Sutton does not mention the criteria that had to be met when the weights vector can be considered “converged”. For our experiment, the sum of the absolute value of the predicted and true weight vectors had to be less than 0.001 for the learner to be considered “converged”. Perhaps Sutton had a threshold larger than 0.001, or he used a completely different method of determination.

IV. Experiment 2 – Reproducing Figure 4 and 5

Experiment 2 presented the same data to the learner, but a few procedural changes were introduced:

1.) The weights vector was updated after every presentation of a sequence to the learner, instead of updating after presentations of entire training sets.
2.) Each training set was presented once.
3.) Each learning procedure was applied with a range of values for α: {0.0 to 0.6 | step size = 0.05} and λ: {0, 0.3, 0.8, 1}
4.) The weights vector was initialized to 0.5

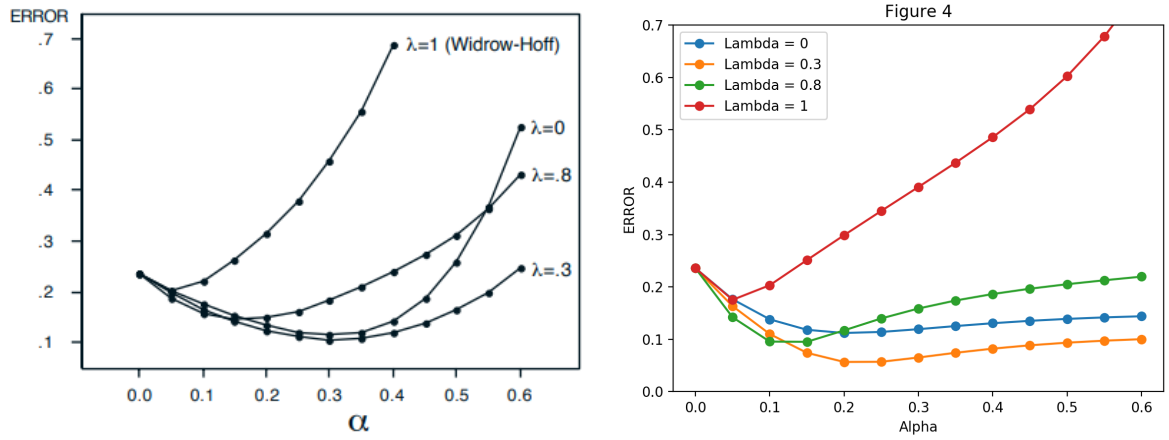

Figure 3 - Comparing Sutton's Figure 4 (on the left) to Ours (on the right)

We plotted multiple curves in Figure 3 with the intention of finding a “best alpha” value that yields the lowest RMS error, and it’s clear the RMS error is lowest when α = 0.25. Sutton’s results show the a “best alpha” around α ≈ 0.2. These alpha values will be used to conduct the second part of Experiment 2, more on this a little farther down. The overall shape of our graph is similar to Sutton’s, but our results are missing some of the concavity featured in Sutton’s graph.

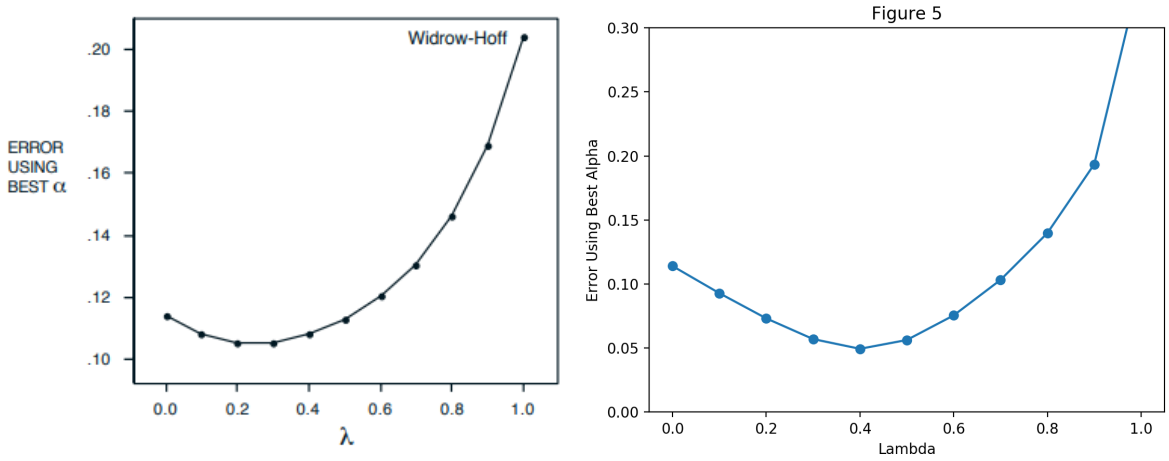

Figure 4 - Comparing Sutton's Figure 5 (on the left) to Ours (on the right)

Figure 4 was produced in a similar way Figure 3 was produced, but only one value for α was used (the "best alpha" determined from Figure 3). Our results are extremely similar to Sutton's. The minimum of our graph occurs at a lambda value that is slightly larger than that of Sutton's; however, the overall shape AND accuracy is very comparable.

V. Conclusion

Overall the results we were able to create were very similar to the results shared by Sutton in his 1988 paper. The most obvious difference between our results and Sutton's is the difference in accuracy. This could be due to different converging criteria and/or a difference in technological capabilities. For example, Sutton could have been using a machine that only supported 32-bit floating-point data types whereas the machine we used to conduct our experiments allow for 64-bit floating-point data types. Though miniscule as far as decimal value goes, this increase in data-resolution could very well contribute to an increase in accuracy across all experiments.

VI. References

Richard S. Sutton. *Learning to Predict by the Methods of Temporal Differences*. "Machine Learning", pages 9–44. Kluwer Academic Publishers, Boston, MA 1988.

VII. Link to Github Repo

https://github.gatech.edu/mling7/CS7642_Project1