Implementation and Analysis of TD Learning Experiments

1. Introduction

In "Learning to Predict by the Methods of Temporal Differences" [1], Sutton shows that TD(λ), which uses temporarily successive prediction, outperforms supervised methods for multi-step predictions. This paper implements Sutton's experiments to better understand TD-learning.

2. Environment and Algorithms

Following experiments uses random walks as environment. There are 7 possible states (A, B, C, D, E, F, G) in random walks. Agent starts at state D at every sequence, and moves stochastically left or right until state A or G is entered, in which sequence ends. Given state A and G have rewards of 0 and 1, TD(λ) is used to estimate probability of right-side termination. Ideal predictions for non-terminal states are 1/6, 1/3, 1/2, 2/3, and 5/6 for states B, C, D, E, and F, respectively.
In TD(λ), learning procedure is summarized as follows:

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

Figure 1. learning procedure formulas [1].

This means that predictions or weight vectors are learned by accumulating changes in weights per each time step. This incremental change weighs successive predictions by value λ, with exponentially higher weight on latest steps. Assuming learner's predictions are linear, P_t and $\nabla_w P_k$ are computed as:

$$P_t = w^T x_t. \qquad \nabla_w P_k = x_t$$

Figure 2. Prediction and its partial derivative formula [1].

At λ = 1 or TD(1), above algorithms simplifies to computations used in supervised methods.

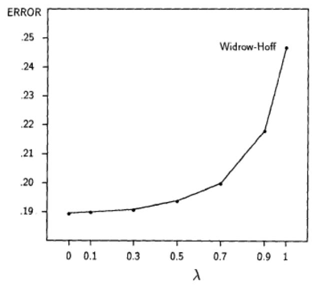
3. Experiment 1 – Repeated Presentation

3.1 Implementation

This experiment computes average RMSE for 100 training sets of 10 sequence data between actual and ideal predictions of random walks. To generate a sequence data, agent is moved randomly left or right with equal probability until terminal state (A or G) is reached. Each state i at step t (e.g. $x_{i,t}$) is represented as a basis vector of length 5 with 1 in at the i^{th} component and 0 otherwise (e.g. x_D = [0, 0, 1, 0, 0]). Representing weight vector (w) with length 5, each for a nonterminal state, prediction P_t is determined as i^{th} component of w.
Learning procedure for this experiment is implemented as follows. At start of each training set, weight vector is initialized to [0.5, 0.5, 0.5, 0.5, 0.5]. Then, for each set's sequences, Δw_t of each time step is determined (as in "Figure 1") by computing $P_{t+1,}$ $P_{t,}$ and $\sum\lambda^{t-k} * x_t$ using $w^T x_{t+1}$, $w^T x_t$, and summation of discounted previous observations, respectively. Here, Δw_t are summed a training set. If total Δw is greater than 0.0001, same training set is used until weights converge. Average RSME over 100 training set for 0<= λ<=1 of 0.1 step is computed between ideal and actual predictions.

3.2 Result

Results are generated as follows (with RSME as average over 100 training set each with 10 sequences):



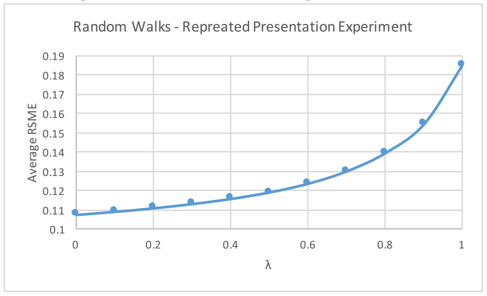


Figure 3. Repeated presentation by Sutton (left) vs. implementation (right)

As shown, result (right) is similar to that of Sutton (left), in which RSME exponentially increases as λ approaches 1. This supports Sutton's conclusion that linear TD(0) converges to optimal predictions, outperforming Widrow-Hoff or TD(1), for future experience.

However, RSMEs computed from implementation is consistently lower (~0.06) than Sutton's for all λs. This drop in error is likely due to higher convergence limit and/or learning rate used for convergence in Sutton's work due to computation limitation at the time.

3.3 Pitfalls

Sutton's paper does not explicitly define convergence condition and learning rate. This complicates implementation as a higher learning rate overflows while higher convergence limit increases error. For example, using convergence of 0.1 in initial implementation produced higher RSME for λ=0 than λ = 1. To overcome divergence and overflow, optimal α was determined by inferring that experiment 2 (below) uses α of 0.1 magnitude. Thus, α, with 10 times weight accumulation, should be 0.1 / 10 = 0.01. Another pitfall is that Sutton does not explicitly define that weight vector should reset at start of a training set. Initially, weight vector was not reset after each training set, which resulted in much lower RSME (~0.04) as weights vector converged much quicker.

4. Experiment 2 – Learning Rate Analysis

4.1 Implementation

Implementation of experiment 2 requires only slight modification to that of experiment 1. Same training data (100 training set each of 10 sequences) and RSME computation is used. However, training set is only presented once at each learning so that there is no search for weight convergence. This means that weights are also updated after each sequence with weight vector initialized similarly to [0.5, 0.5, 0.5, 0.5, 0.5] at start of each training set. Lastly, both learning rate and λs are variables to examine.

4.2 Result

Performing these modifications to experiment 1, results are generated as follows

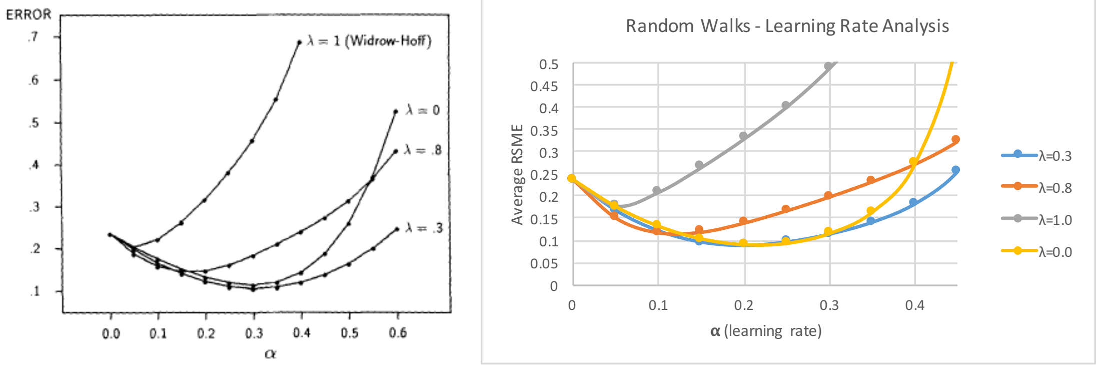


Figure 4. Single presentation with variable learning rate by Sutton (left) vs. implementation (right)

As shown, all RMSE curves from implementation are shifted left from Sutton's work by ~0.1 α, but still maintain general shape and magnitude of error. This agrees with Sutton's finding that TD methods of λ < 1 outperforms TD(1) over range of α's, and that learning rate has significant effect on RSME. Differences in implementation results is potentially due to limitation of training sets used in Sutton's work. For example, a sequence of observation length 20 produces closer predictions than one of length 5 because more of true prediction is propagated in the calculation. Thus, a smaller learning rate or less weight on previous predictions is necessary to approach ideal estimates. This means Sutton could have filtered long-length sequences for training.

4.3 Pitfalls

This experiment assumes linear prediction, which allows partial derivative for prediction to be observation at step t. It also does not place a limit in number of steps a training sequence can take. In addition, weight vector is assumed to be re-initialized at start of each training set as error accounts for changes in weights per set.

5. Experiment 3 – Best alpha Analysis

This experiments examines minimal RSME for each λ using best α determined previously.

5.1 Implementation

This implementation requires only slight modification to experiment 2. Same training data (100 training set with 10 sequence each) is used for single presentation. As previously, weights are updated after each sequence. However, learning rate in this experiment is set to α with minimal error determined in "Figure 4". Thus, for λ from 0 to 10 with 0.1 step, α's = [0.2, 0.2, 0.2, 0.2, 0.2, 0.15, 0.15, 0.15, 0.10, 0.10, 0.05]. Average RSMEs are computed over 100 training sets. In this experiment, weight vector is also initialized to [0.5, 0.5, 0.5, 0.5, 0.5].

5.2 Result

With modifications above, results are generated as follows:

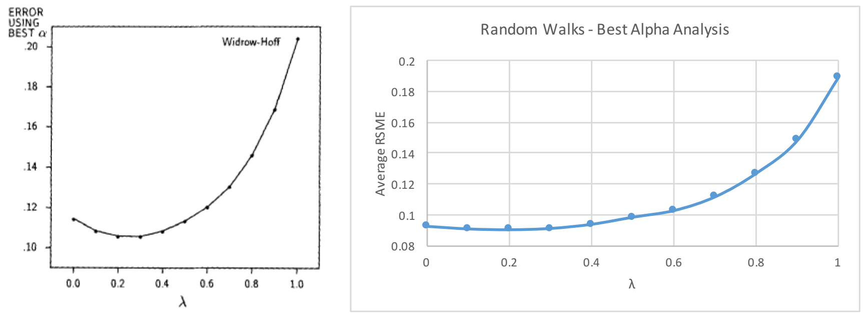


Figure 5. Single presentation with optimal learning rate by Sutton (left) vs. implementation (right)

As shown, implementation produces similar result to Sutton's work with all λ < 1 outperforming TD(1). However, optimal (least error) λ occurs at 0.2 instead of 0.3 as mentioned in Sutton's finding. This difference in best λ is likely due to using slightly different optimal α values for implementation as compared to Sutton's reference. This result, however, still also demonstrates that single presentation has slower learning as best λ is no longer at 0 as found in experiment 1.

5.3 Pitfalls

Sutton does not explicitly mention initial values of weight vector. Thus, implementation assumes same vector values as experiment 2 as this minimizes bias toward either right/left side termination. In addition, as previously, this experiment assumes linear prediction.

6. Conclusion

In conclusion, implementation results are very similar to Sutton's finding, particularly that TD(***λ***) outperforms supervised methods / TD(1) for multi-step predictions. In addition, learning procedure, especially specification of learning rate and weight update procedure, significantly affects performance.

7. Video Presentation

https://youtu.be/kpJ1lCTouP8

8. Reference

[1] R. Sutton. Learning to Predict by the Methods of Temporal Differences. Machine Learning 3: 9-44, 1988.