# CS7642 - Decoding Temporal Difference Learning

5ce6ff97a7bb9920e328ae57c68c47f3d7dfb683 (Git Hash)

Preprint, compiled September 28, 2020

**Sangeet Moy Das (sangeet.das@gatech.edu)**

College of Computing, Georgia Tech

**Abstract**

The objective of this project report is to discuss the fundamentals of Temporal Difference (TD) Learning and provides details of steps in reproducing the results in Sutton's paper "Learning to Predict by the Methods of Temporal Differences". The major focus of this report will be to apply Temporal Difference (TD) learning to the "Random Walk" example discussed by Sutton in the first half of the paper.

## 1 Introduction

The problem of predicting the expected long term future reward of a stochastic system without having the knowledge of the system itself is conventionally done using supervised learning methods where we predict the target output variable using a set of independent variables. In this method of prediction, the learner is expected to associate patterns to rules during the training process and then predict the target variable when only the independent variables or input features are known. In the paper, "Learning to predict by the method of temporal difference", Sutton proposed the concept of Temporal Difference (TD) learning, in which, instead of pairing up observations with the target variables, it takes the temporal relationships between states into consideration and pairs them up with their corresponding predictions in the training sequence. This learning technique focuses more on the errors between temporally successive predictions instead of doing prediction vs. actual, as in the case of supervised learning techniques. One thing to note here is that the TD learning technique assumes that the state of the environment is somewhat continuous and does not change from one point in time to the next. In other words, the environment is predictable and stable. If we accept these assumptions we are provided with three key advantages:

1. Easier to compute because it is incremental in nature
2. Makes accurate predictions by using experience
3. Closer to the actual learning behavior shown by humans

## 2 Background

### 2.1 Supervised Learning

For a prediction problem in the form of observations-outcome sequence: $x_1, x_2, ...xm, z$, where $x_t$ is a state vector at time t and z is a scalar value of the reward/outcome. The prediction is a sequence of estimation of z value associated with each state: $P_1, P_2, ...P_m$, where $P_t$ is a function of all proceeding states including the current state: $x_1, x_2, ...xt-1, x_t$.

### 2.2 Widrow-Hoff Learning Rule

In the late 1950's it was evident that the need for a more flexible and powerful mathematically derived model has arisen and that was when the Widrow & Hoff learning rule or the Least Mean Square (LMS) rule was invented by Widrow and Hoff, which was also called as the Delta Rule. The Delta Rule uses the difference between target activation (i.e., target output values) and obtained activation to drive learning. The Delta Rule employs the error function for what is known as Gradient Descent learning, which involves the 'modification of weights along the most direct path in weight-space to minimize error', so change applied to a given weight is proportional to the negative of the derivative of the error with respect to that weight.

**Algorithm 1:** Widrow-Hoff

initialize $\boldsymbol{w}_1 = \mathbf{0}$;
**for** $t = 1$ **to** $T$ **do**
  get $\boldsymbol{x}_t \in \mathbb{R}^n$;
  predict $\hat{y}_t = \boldsymbol{w}_t \cdot \boldsymbol{x}_t$;
  observe $y_t$;
  incur loss of $(\hat{y}_t - y_t)^2$;
  update $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta(\boldsymbol{w}_t \cdot \boldsymbol{x}_t - y_t)\boldsymbol{x}_t$;
**end**

Algorithm 1: The Widrow-Hoff Algorithm.

## 3 Random Walk & Temporal Difference (TD) Learning

In Sutton's 1988 paper, the method described for making predictions using past experience employs a prediction function with functional dependence on experienced input observations x and a set of learned weights w, and a learning process updates this set of weights incrementally with experience using stepwise increments between predictions P:

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

Where $\lambda$ is a value between 0 and 1 and acts as a discount rate on past observations. By changing the values of lambda, the weighting of past observations relative to recent observations can be controlled. The gradient of $P_k$ with respect to w represents the rate of ascent along with the derivatives of $P(X_t, w)$, and in a linear function, this is simply $X_t$. The value of t represents the number of steps taken and $\alpha$ is the learning rate.

### 3.1 Random Walk

Sutton in his 1988 paper has described an experiment that was designed to evaluate the effectiveness of this learning procedure by varying the value of $\lambda$ while learning the predicted value of each step in a bounded random walk. A Random Walk task is depicted below which has has been taken from figure 2 in Sutton's paper.

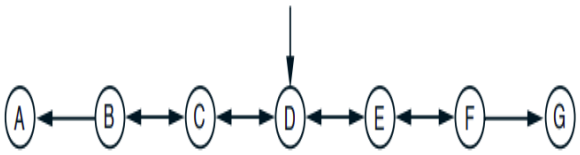


Figure 1: A bounded random walks from Figure 2,Sutton (1988).

Illustrated in figure 1, the random-walk sequences always starts with the initial center state D and then transition left or right with uniformly random steps where P(left) = P(right) = 0.5, untils it arrives at one of the terminal states A or G, which commences the end of the random walk. If the random-walk ends at state A, the reward z = 0; When it ends at state G, the reward z = 1.

In the settings of the bounded random-walk, there are two types of states:

- Non-terminal states: B, C, D, E, F
- Terminal states: A, G

The non-terminal states were represented in the form of a vector, where each state was represented as a column vector with a dimension of (5x1). The terminal states A and G are not one-hot encoded but rather associated with a reward of z = 0 or z = 1. For the sequence illustrated in figure-1 and considering taking the right path with states D, E, D, E, F, E, F, G; it is represented with reward z = 0 and non-terminal state sequence in the form of a matrix of 0's and 1's.

### 3.2 Temporal Difference Learning

Three different experiments were conducted employed towards bounded Random Walk by Sutton which sought to explore the effects of varying both the learning rate $\alpha$ and the parameter $\lambda$, across repeated versus single presentations of training data.

1. In the first experiment, $\alpha$ was held constant and $\lambda$ was varied during repeated presentation of 100 sets of training data, with each set containing 10 bounded random walk sequences. The training data was presented to the learner until the learner's weight update value converged, with convergence defined by a threshold value for the size of the updates. After collecting the weight vectors, the predictions, that the learned weights produced were equal to the weights themselves, as the x values were vectors of zeros with a single one value, resulting in $P(X_t, w)$ as $w^T X_t$. After generating the weights vectors with varied values of $\lambda$, the results were compared against the known actual probabilities from each state, which were $[\frac{1}{6}, \frac{1}{3}, \frac{1}{2}, \frac{2}{3}, \frac{5}{6}]$ with the error measure reported as root mean squared error (RMSE) between the two.
2. The second experiment explored the effect of four $\lambda$ values across all values of $\alpha$, again reporting error as RMSE.
3. The third experiment showed the effect of varied values of $\lambda$ for each value's best value of $\alpha$.

Experiments two and three functioned with only a single presentation of 10 sequences of training data, averaged over 100 trials. The results of each experiment are shown in the original figures from Sutton (1988) below:

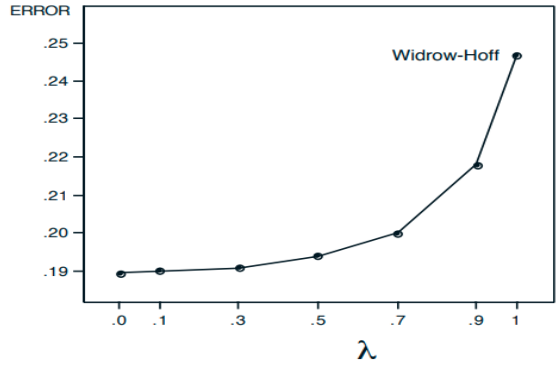


Figure 2: Actual image of experiment 1 from Figure 3, Sutton (1988) showing average error on random walk problem under repeated presentations.

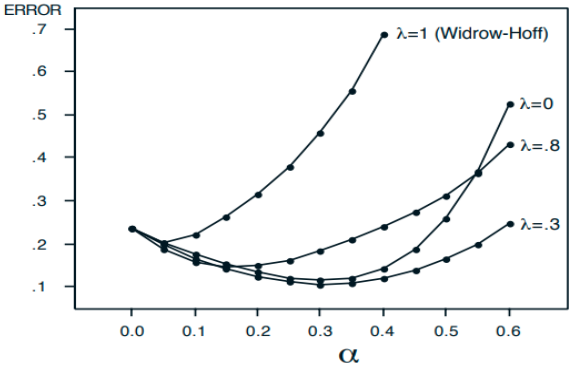


Figure 3: Actual image of experiment 2 from Figure 4, Sutton (1988) showing average error on random walk problem after experiencing 10 sequences.

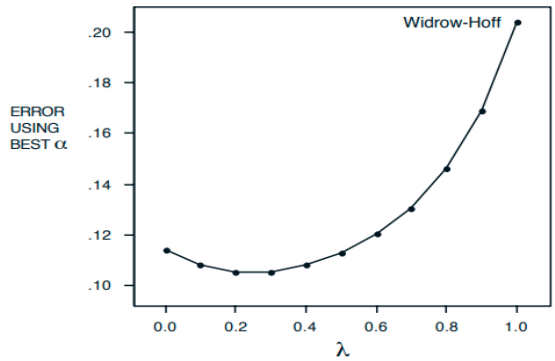


Figure 4: Actual image of experiment 3 from Figure 5, Sutton

(1988) showing average error at best $\alpha$ value on random walk problem.

The goal of this work is to replicate the results from Sutton obtained in the figures shown above. In order to this, a Python3 script was written to generate uniformly random steps for the training data. The TD($\lambda$) procedure was also implemented which performed the updates to the weights vectors. As in Sutton's work, the weights were initialized to a vector with each equal to 0.5. After performing the updates as described in Suttons's work, the figures shown above were replicated.

## 4 EXPERIMENTS & RESULTS

In this section of the report, we will discuss about the reproduction of figure 3, 4 & 5 from Sutton's paper. In order to replicate the results from Sutton's paper, a two step approach was taken:

- Calculating the ideal predictions of non-terminal states in the random walk problem, using the methods mentioned in the paper
- Conducting a two-fold experiment:
  - Repeated presentation of training sequences using different values of $\lambda$;
  - Single presentation of training sequences with unbiased initialization and different combinations of $\lambda$ & learning rate

## 5 PREDICTED WEIGHTS IN RANDOM WALK

Weight w(i) for a non-terminal state sequence represents the expected value of the outcome z given that the sequence of states start from i. In the bounded random walk example from Sutton's paper, Q is the transition probability matrix where $Q_{ij}$ is the transition probability between non-terminal states i and j, whereas h is the transition probability vector between non-terminal states and the terminal state G. The predicted weight of w(i) can be calculated as follows:

$$E\{z|i\} = \left[\sum_{k=0}^{inf} Q^k h\right]_i = \left[(I-Q)^{-1} h\right]$$

The vectorized form of ideal weights for non-terminal states B, C, D, E & F is derived as:

$$E(z) = \left[(I-Q)^{-1} h\right]$$

$$= (1 - \begin{bmatrix} 0 & 0.5 & 0 & 0 & 0 \\ 0.5 & 0 & 0.5 & 0 & 0 \\ 0 & 0.5 & 0 & 0.5 & 0 \\ 0 & 0 & 0.5 & 0 & 0.5 \\ 0 & 0 & 0 & 0.5 & 0 \end{bmatrix})^{-1} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0.5 \end{bmatrix} = \begin{bmatrix} 1/6 \\ 1/3 \\ 1/2 \\ 2/3 \\ 5/6 \end{bmatrix}$$

### 5.1 TD($\lambda$) with repeated presentations

For this experiment 100 randomly generated training sets were taken with each containing 10 bounded random walks sequences. In repeated presentations, for a specific value of $\lambda$, the learner is repeatedly presented with 10 randomly generated random walk sequences in a training set until it converges. The learner accumulates $\Delta$w across these sequences and only performs the weight update after a full epoch of training. This is repeated for 100 training sets and computes the average root mean squared error (RMSE) between the learned weights and ideal weights of non-terminal states: $X_B, X_C, X_D, X_E, and X_F$.

In the below reproduced figure 5, which is a close replication of Figure 3 from Sutton's paper shows that TD(1) (Widrow-Hoff) performs the worst compared to other $\lambda$ values. This is because that, TD(1) process minimizes the error between predictions and actuals in the training sets, however it does not mean that future predictions will be better. TD(0) here has the lowest RMSE as it is only concerned with the most recent state and it is consistent with the maximum-likelihood estimation (MLE) for the markov process.

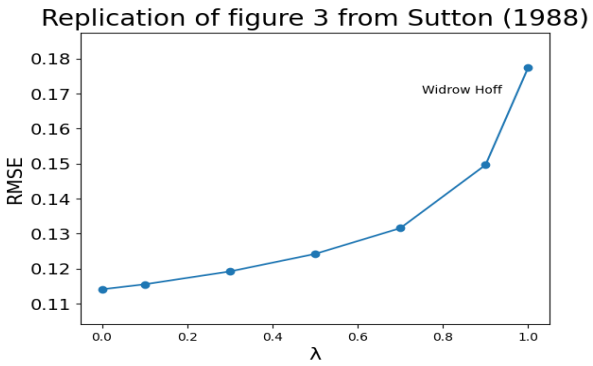


Figure 5: Replicated image of figure 3, Sutton (1988) showing average error on random walk problem under repeated presentations.

The reproduced results of the figure 3 from Sutton's paper seems to have lower RMSE compared to that of Sutton's report. This has been the case for different randomly generated training sets and has been verified.

### 5.2 TD($\lambda$) with single presentations

For this experiment, the same randomly generated 100 training sets were used and the weights are taken as 0.5 for all non-terminal states. In the single presentations paradigm, for a specific combination of $(\lambda, \alpha)$ values, the learner can only learn the 10 random walk sequences for a single time in a training set and weight w is updated by the learner at the end of each random walk sequence. This is repeated for 100 training sets and computes the average RMSE between the learned weights and ideal weights. Combinations of $\lambda$&$\alpha$ are tested to analyze the impacts of learning rate on learning outcomes.

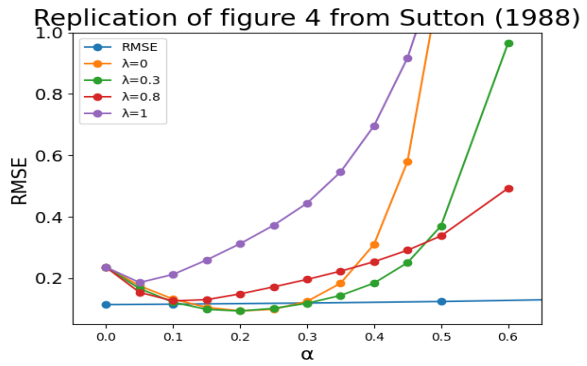


Figure 6: Replicated image of figure 4, Sutton (1988) showing average error on random walk with single presentation for $\lambda$ = 0.0, 0.3, 0.8, 1.0

The above figure 6, which is a close reproduction of figure 4 from Sutton's paper, demonstrates the average RMSE between the learned weights w and ideal weights of $[\frac{1}{6}, \frac{1}{3}, \frac{1}{2}, \frac{2}{3}, \frac{5}{6}]^T$ using different combinations of $\lambda and \alpha$ where each line in the graph represents the RMSE for a given $\lambda$ across different learning rate $\alpha$. Learning rate $\alpha$ has a significant impact on the temporal learning outcome. Lower $\alpha$ values of around 0.2 & 0.3 yield the predictions with the lowest errors. We can notice that a $\alpha$ = 0.2 can achieve good rate of convergence. We can also notice that the TD($\lambda$ = 1) (Widrow-Hoff) produces the worst results regarding of the learning rate, which shows the limitation of the conventional used TD($\lambda$ = 1) (Widrow-Hoff) approach.

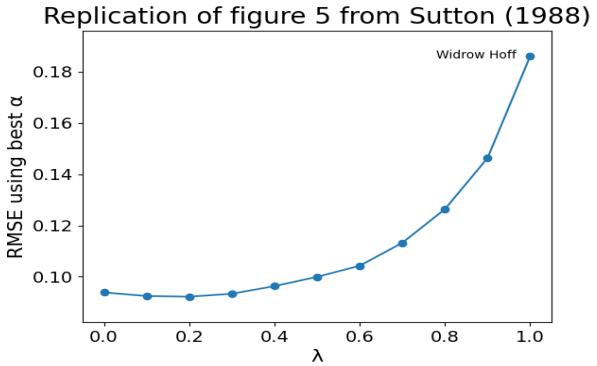


Figure 7: Replicated image of figure 5, Sutton (1988) showing average error at best $\alpha$ value on random walk problem with single presentation for $\lambda$.

The above figure 7, which is a close reproduction of figure 5 from Sutton's paper, demonstrates the best RMSEs achieved in each $\lambda$ value from figure 4 of Sutton's paper. From this graph as well, it is pretty evident that TD($\lambda$ = 1) gives the worst predictions. The best $\lambda$ in this experiment is 0.2.

## 6 CONCLUSIONS

This report is aimed at providing a detailed walk through of bounded random walk process as discussed in Sutton's paper titled "Learning to predict by the methods of temporal difference". Section 2 provides background on Supervised Learning & Widrow Hoff, while section 3 synthesizes the mathematical derivations from Sutton's paper. Section 4 discusses the algorithm implemented to reproduce figure 3, 4, & 5 from Sutton's paper. This implementation and analysis done in context of this report, helps in acquiring a good understanding of Temporal Difference (TD) learning and comprehend the impact of parameters $\alpha$ and $\lambda$ on the outcome of the predictions. It is also evident that a tuned value of $\lambda$ should be adopted on the problem settings and leverage the strengths of temporal difference which include better accuracy & incremental updates of the predictions.

## 7 REFERENCES

1. J. Tsitsiklis and B. V. Roy, An analysis of temporal-difference learning with function approximation, IEEE Transactions on Automatic Control, vol. 42, no. 5, pp. 674690, 1997.
2. R. S. Sutton, Learning to predict by the methods of temporal differences, Machine Learning
3. R. S. Sutton, "Generalization in reinforcement learning: Successful examples using sparse coarse coding," in Advances in Neural Information Processing Systems, vol. 8, D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, Eds. Cambridge, MA: MIT Press, 1996.
4. D. P. Bertsekas, "A counterexample to temporal-difference learning," Neural Comp., vol. 7, pp. 270–279, 1994.
5. Richard S Sutton and Andrew G Barto. Reinforcement learning: An introduction. 2nd Ed. MIT press, 2020. url: http://incompleteideas.net/book/the-book-2nd.html.