Critique of Temporal Difference Learning by Sutton

How can it be stated that the TD method is reliable? How can you prove that it is faster at converging?

This can be clearly explained using an example of game play.

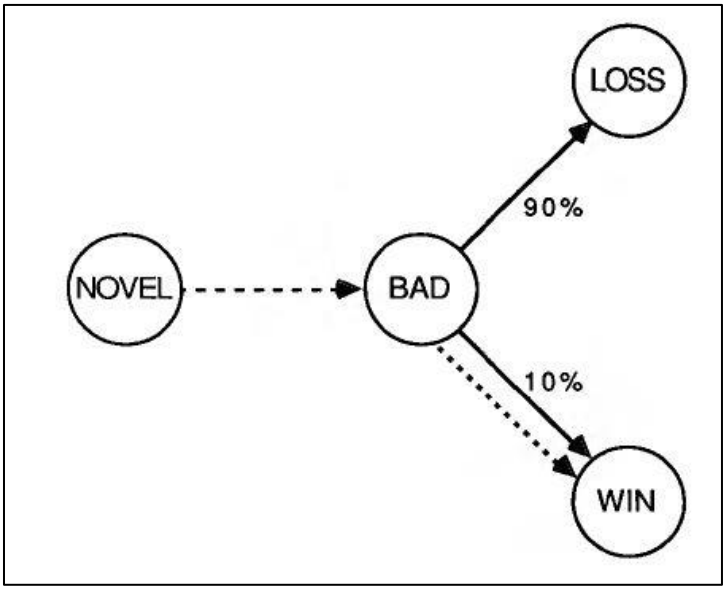


Figure 1: A game play with the Novel state leading to a win

From the above fig., a novel state implies and unknown/unvisited state. The novel state passes through a BAD state which usually (90% of times) leads to a loss, but during this trial apparently wins! In Supervised Learning the novel state would have been considered a good state which is not true as the Novel state leads to a bad state which eventually leads to a loss 90% of the times. As against to this, in Temporal Difference method, you take the difference of predictions from the consecutive states i.e., since the novel state leads me to a bad state, it can be inferred that the novel state is also a bad state but with lesser probability.

It must be observed that both TD method and Supervised Learning converge after long period of training. But as observed above, the TD method is able to estimate better due to its update from consequent states. In specific to this example, as the TD method takes the difference from the Prediction of the bad state, the novel method is also considered to be a bad state.

Although this methodology works in general, it is at a disadvantage when the novel state actually leads to a win which then will have a to be considered as a good state but is considered bad with TD!

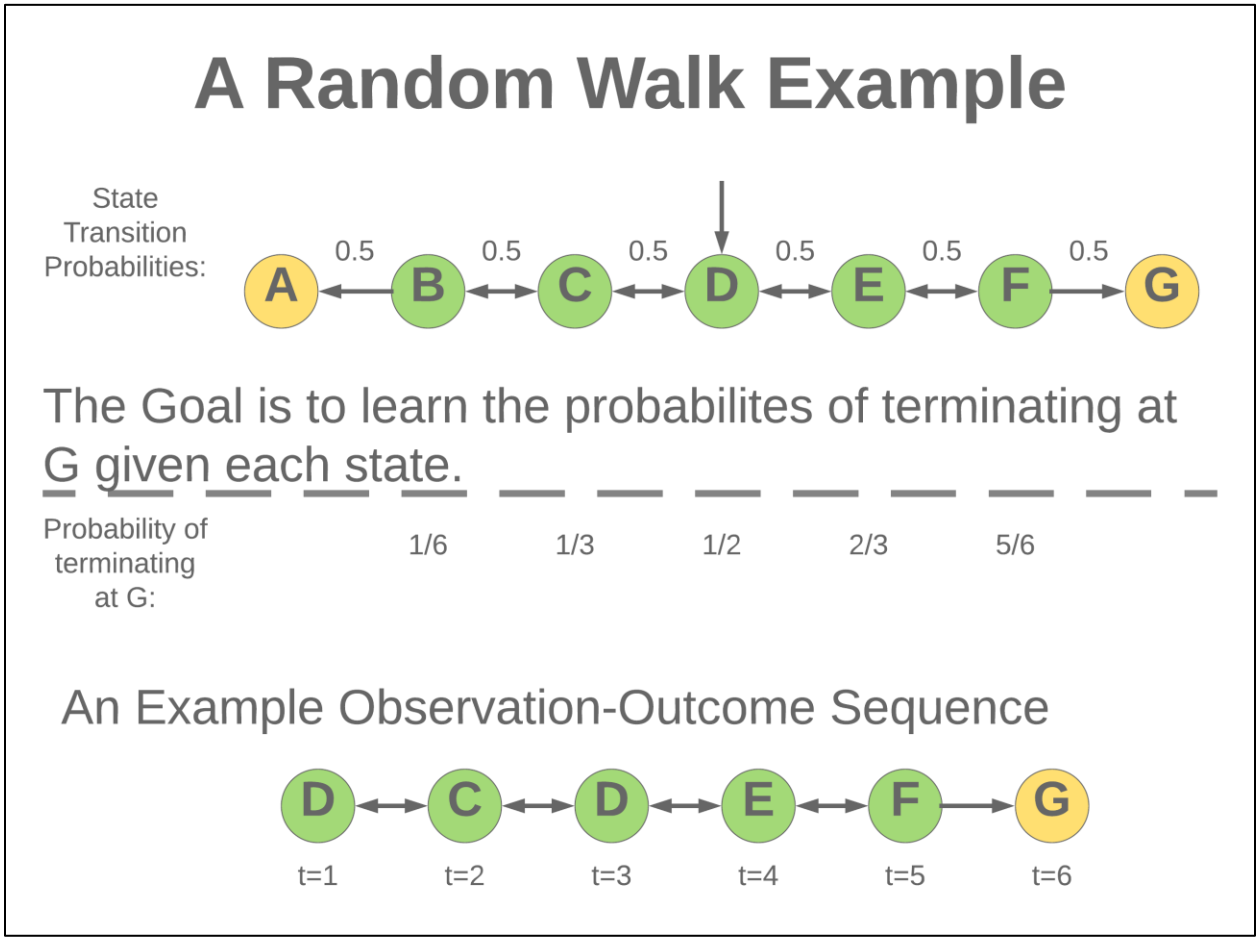
Explaining the TD algorithm through simple Random Walk example

The game setup is as below:

- The game consists of initial state as D.
- The probability of transition to either of the adjacent states is equal.
- The transition can be made bidirectionally.
- A and G are terminating states while
- B, C, D, E and F are non-terminal states.
- All the non-terminating states are represented in vector form as $x_t = \mathrm{x}_D = [0,0,1,0,0]$
- Here, the goal is to reach the terminal state G.
- Implies that **finding the chances of winning when at each state must be found**.

The experiment for arriving at the required result was conducted as below:

- Generate 10 observation-outcome sequence per training set.
- Generate 100 such training sets.
- Test the experiment with different values of $\lambda = 0,0.1,0.3,0.5,0.7,0.9,1.$



The algorithm for TD:

1. $P_t = w^T.x_t\ (linear\ relation)$

2. Hence for $t = 1$, compute P_1 using the above equation.
3. Then compute P_2.
4. Now to update the weight vector w, compute $\Delta w_t = \alpha(P_{t+1} - P_t).\sum_{k=1}^{t} \lambda^{t-k} \frac{\partial P_k}{\partial w}$.
5. That is, $\Delta w_1 = \alpha(P_2 - P_1).\sum_{k=1}^{1} \lambda^0 \frac{\partial P_k}{\partial w} = \alpha(P_2 - P_1).\mathrm{x}_1$
6. The Δw_t is accumulated for all timesteps from t=1 to 6
7. w is updated as $w = w + \sum_{t=1}^{m} \Delta w_t$.
8. Here, the $(m+1)^{\text{th}}$ state is defined as $P_{m+1} = z = 1$.
9. Hence, when the weights are equally initialized, the updates Δw_t will all be zero except for the last state m.
10. This algorithm is repeated for all sequences in the training set until convergence.
11. The results are compared with all the 100 training sets to test convergence.
12. This experiment is conducted with different values of λ as stated above and the below results are obtained.

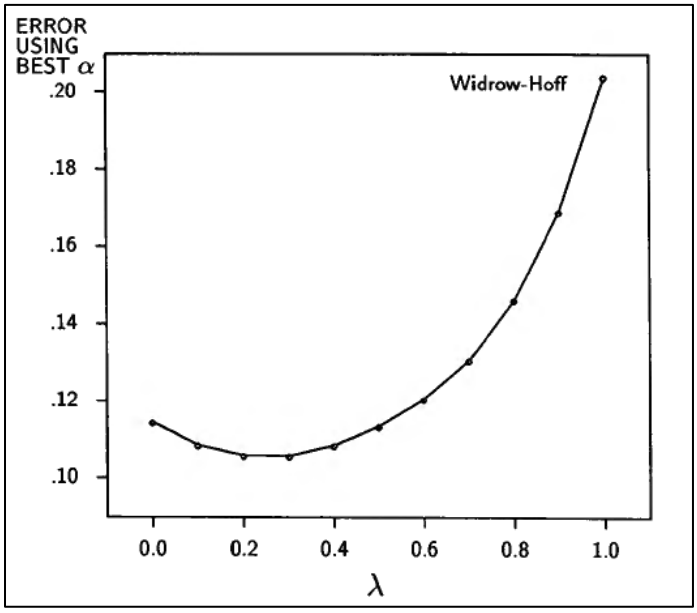


Figure 2: Least error achieved for different values of λ

Observations made:

- When $\lambda = 1$, the error is propagated to all the states in the sequence and is popularly called the Widrow-Hoff equation and results in a very high rate of error.
- When $\lambda = 0$, the error is not propagated to the previous states at all! While this is good, the previous and consequent states are dependent on each other and hence require the error to be propagated as well.
- As a result, when $\lambda = 0.3$, the lowest error is achieved.
- Fig. 3 is a result of the experiment conducted with different learning rates α and λ.

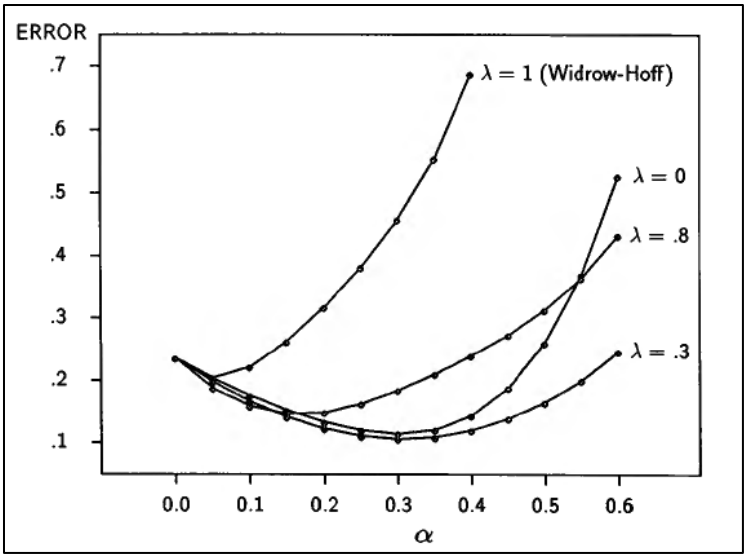


Figure 3: Least error achieved for different learning rates α and discounting factor λ

The author also presents a detailed proof for convergence of the algorithm. He proves the two following propositions:

1. Linear TD(0) algorithm converges asymptotically when presented with new data sequences. The below fig. 4 shows a depiction of asymptotic convergence.
2. Linear TD(0) converges on repeated presentations of a training set.
3. Consequently, he also shows how the TD method is similar to the gradient Descent methods.

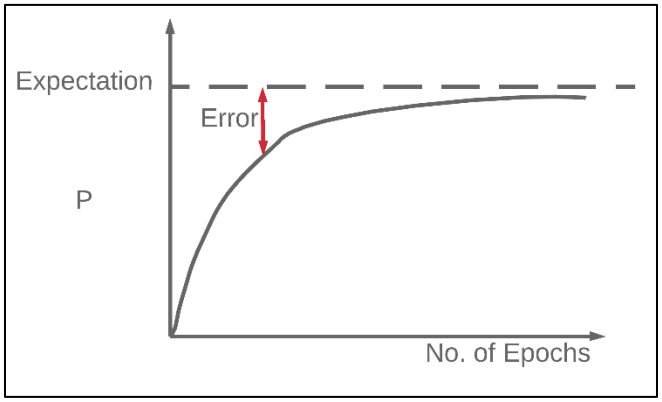


Figure 4: Asymptotic convergence of TD(0) to achieve the expected result

The TD algorithm can be extended for other algorithms which require accumulation of a value like a cost over the sequence. In such situations, a parameter c_{t+1} can be added to the error term $(P_{t+1} - P_t)$ to get $(c_{t+1} + P_{t+1} - P_t)$ and the expected output can be set to obtain $z_t = \sum_{k=t}^{m} c_{k+1}$. Here you accumulate the cost at each time step and predict the cost for the remaining timesteps which is given by z_t.

Another proposition by Sutton was intra sequence updates, where the weights can be updated at each time step rather than accumulating them over all time steps and then updating.

The resulting equation is simply as below:

$$w_{t+1} = w_t + \alpha(P_{t+1} - P_t).\sum_{k=1}^{t} \lambda^{t-k} \frac{\partial P_k}{\partial w}, where\ P_t \stackrel{\text{def}}{=} P(x_t, w_{t-1})$$