Stanford CME 241 (Winter 2023) - Assignment 7

Due: 3/6 @ 11:59pm (after the exam). Please solve all 3.

1. In the following question, we explore the connection between TD and MC algorithms.

   a. Implement the TD(λ) Prediction algorithm from scratch in Python code. First do it for the Tabular case, then do it for the case of Function Approximation.

   b. Prove that the MC Error can be written as the sum of discounted TD errors, i.e.,

   $$G_t - V(S_t) = \sum_{u=t}^{T-1} \gamma^{u-t} \cdot (R_{u+1} + \gamma \cdot V(S_{u+1}) - V(S_u))$$

   The goal here is for you to practice formal proof-writing of these types of simple yet important identities. So aim to work this out from scratch rather than treating this as a special case of a more general result proved in class or in the textbook.

   c. Test your above implementation of TD(λ) Prediction algorithm by comparing the Value Function of an MRP you have previously developed (or worked with) as obtained by Policy Evaluation (DP) algorithm, as obtained by MC, as obtained by TD, and as obtained by your TD(λ) implementation. Plot graphs of convergence for different values of λ.

   d. Extend RandomWalkMRP (in rl/chapter10/random_walk_mrp.py) to RandomWalkMRP2D which is a random walk in 2-D with states $\{(i,j)|0 \le i \le B_1, 0 \le j \le B_2\}$ with terminal states as $(0,j)$ and (B_1,j) for all j, $(i,0)$ and (i,B_2) for all i, and with reward of 0 for all $(0,j)$ and for all $(i,0)$, reward of 1 for all (B_1,j) and for all (i,B_2), and with discrete probabilities of 4 movements - UP, DOWN, LEFT, RIGHT from any non-terminal state. Analyze the convergence of MC and TD on this RandomWalkMRP2D much like how we analyzed it for RandomWalkMRP, along with plots of similar graphs.

2. In this question, we will explore three different algorithms for control based on MC or TD. Please complete 2 of the following 3 implementations. For each algorithm, we expect you to test your implementation against the Optimal Value Function/Optimal Policy obtained by DP on SimpleInventoryMDPCap in rl/chapter3/simple_inventory_mdp_cap.py. Then, generalize to MC Control with Function approximation and test your implementation against the Optimal Value Function/Optimal Policy obtained by ADP on AssetAllocDiscrete in rl/chapter7/asset_alloc_discrete.py.

   1. Implement Tabular Monte-Carlo Control algorithm in Python with GLIE implemented as $\epsilon = \frac{1}{k}$ for episode number k and initial state of each episode sampled uniformly from the state space.
   2. Implement Tabular SARSA algorithm in Python with GLIE and a parameterized trajectory of decreasing step sizes.
   3. Implement Tabular Q-Learning algorithm in Python with infinite exploration of all (state, action) pairs and with a parameterized trajectory of decreasing step sizes.

3. Finally, we will explore reinforcment learning algorithms and apply them to the problem of American options pricing. Implement the following two algorithms and apply them to the problem of American Options Pricing, as covered in class. Test by comparing the pricing of American Calls and Puts against the Binomial Tree implmeentation in rl/chapter8/optimal_exercise_bin_tree.py.

1. LSPI
2. Deep Q-Learning