

Stanford CME 241 (Winter 2022) - Assignment 15

Assignments:

1. Assume you have fixed data available as N complete episodes where each episode is in the form:

$$S_0, R_1, S_1, R_2, S_2, \dots, S_{n-1}, R_n, T$$

where S_0, S_1, \dots, S_{n-1} is the sequence of non-terminal states visited in the episode and R_1, R_2, \dots, R_n are the associated rewards following the visited states. T is the terminal state. Note that n (the length of an episode) can vary across the N episodes. Note that there are no actions here, so the setting for this data is an underlying MRP and not MDP. Assume discount factor $\gamma = 1$.

Given only this data of fixed N episodes, your task is to implement working Python code that will estimate the Value Function for discount factor $\gamma = 1$ based on a variety of methods. An outline of the code is made available for you [here](#). A couple of functions (`get_state_return_samples`, required for tabular Monte-Carlo, and `get_state_reward_next_state_samples`, required for the other methods) have been implemented for you and you may use them as helper functions. Your task is to implement the following methods, each compatible with their respective function interfaces provided in this outline code.

- **Tabular Monte-Carlo:** Implement `get_mc_value_function`.
- **MRP:** Implement `get_probability_and_reward_functions` and using its output, implement `get_mrp_value_function` (based on MRP Bellman Equation).
- **Tabular TD(0):** Implement `get_td_value_function`.
- **LSTD:** Implement `get_lstd_value_function`.

Make sure to read the comments/hints provided in the outline code (within each of the above functions you need to implement).

If you run the `__main__` code, it will evaluate each of your 4 methods on the data that is set up in the `__main__` code. Do they all give the same Value Function? If not, do some of them give the same Value Function? Explain the Value Functions you obtain with these 4 different methods based on the theory you have learnt.

2. **Optional** Implement TD with Gradient Correction (abbreviated as TDC, a form of Gradient TD) for Prediction in Python. As covered in class, you have to do cascade learning, by learning the \mathbf{w} parameters as well as the $\boldsymbol{\theta}$ parameters. Test your TDC Prediction algorithm against MC or TD prediction that you had previously written on an example simple MRP that you have previously modeled.