Planning Taxi Domain

# Part A: Computing Policies

## Taxi Domains

* **State space:** We have defined it as a list of 5-element tuple (x, y, status, a, b), in which (x,y) represent coordinate of taxi, (a,b) represent coordinate of passenger and status represent whether taxi is occupied by passenger or not. **Goal state** is reached when taxi has dropped the passenger at destination which means (goal[0],goal[1],False,goal[0],goal[1]).
* **Action space:** There are 6 actions possible [‘N’,’S’,’E’,’W’,’Pick’,’Drop’]. At every possible states all six actions can be performed, whether it will result in something or not doesn’t matter.
* **Transition model:** T(s,a,s’) <- transition[state][action][result], it stores probability of getting into state s’(result) from s(state) over taking action a(action).
* **Reward model:** R(s,a,s’) <- reward[state][action][result], it stores reward for a(action) when jumping from state s(state) to state s’(result). Value of reward is as per stated in question.

Simulator: Implementation of different things like next state based on current state and current action, stochastic effect of actions, rewards for each action.

Instance: Contains all information about starting passenger location, destination location, starting taxi location.

## Value Iteration

Implemented value iteration by using the formula (V\_k+1(s) <- max\_a ∑\_s T(s,a,s’)[R(s,a,s’) + Y.V\_k(s’)]), V\_k+1=cvfn, V\_k=pvfn. Termination of iteration when max-norm error becomes less than epsilon.

1. For discount factor = 0.9 and epsilon = 0.01, number of iteration required was 22.
2. Discount factor = 0.01 => 3 iteration

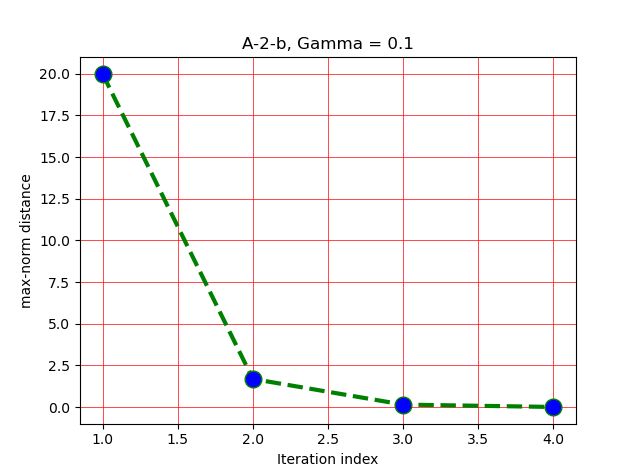
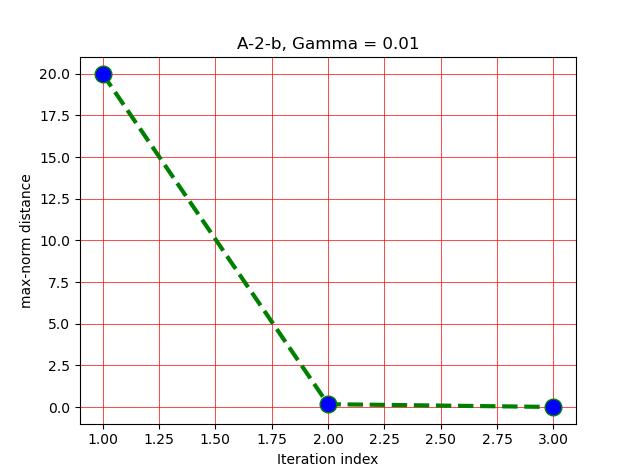
Discount factor = 0.10 => 4 iteration

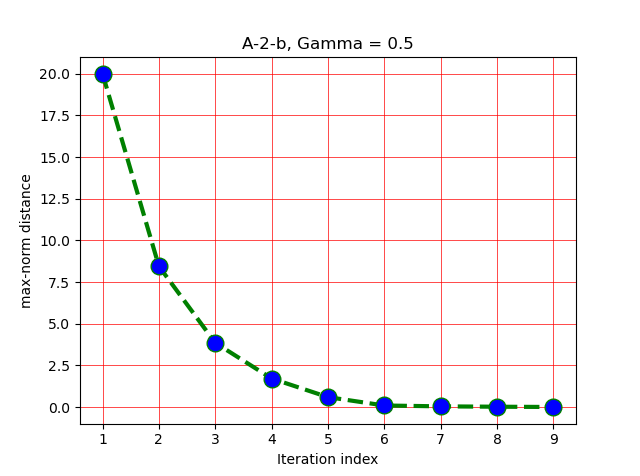
Discount factor = 0.50 => 9 iteration

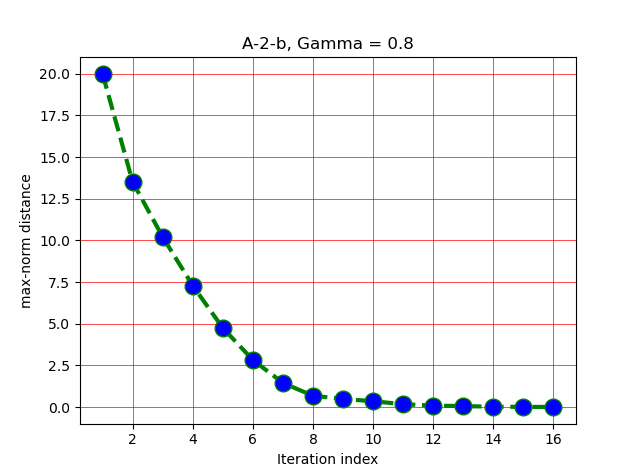
Discount factor = 0.80 => 16 iteration

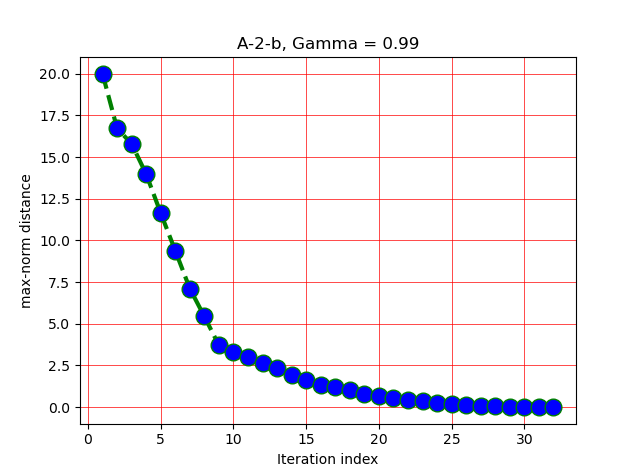
Discount factor = 0.99 => 32 iteration

\*\* PLOT the graph b/w iteration index(x-axis) and max-norm(y-axis) for each and describe observation. \*\*





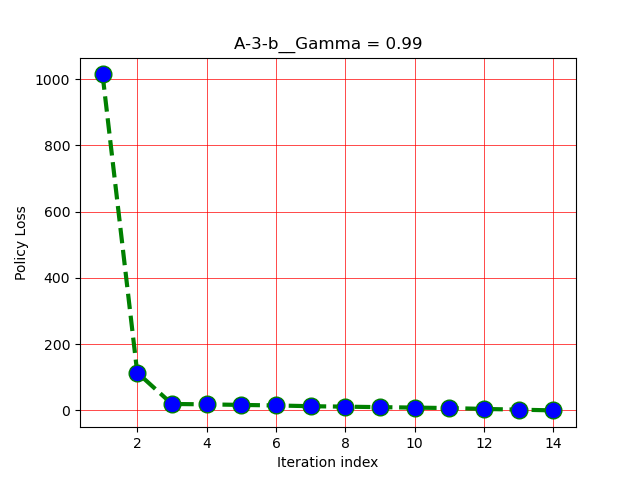
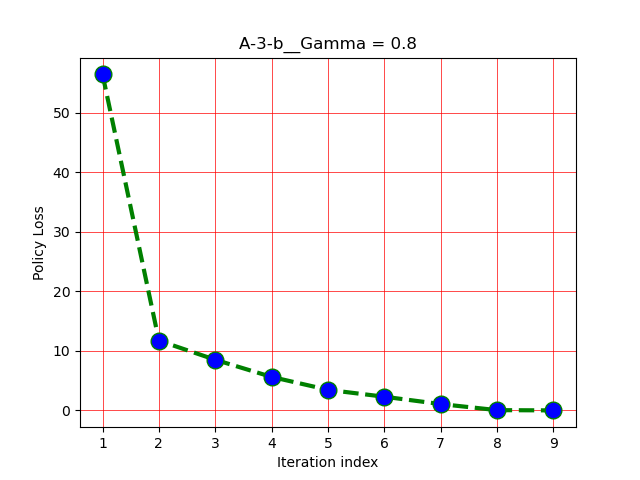
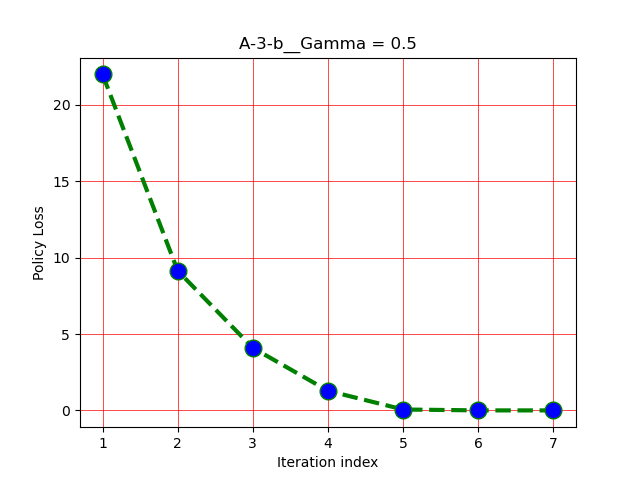
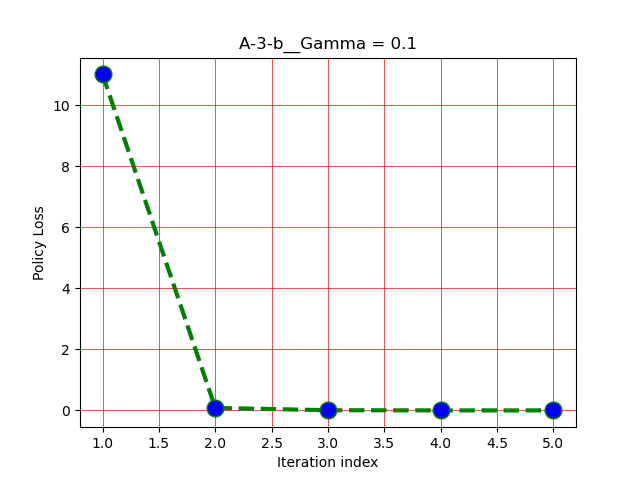
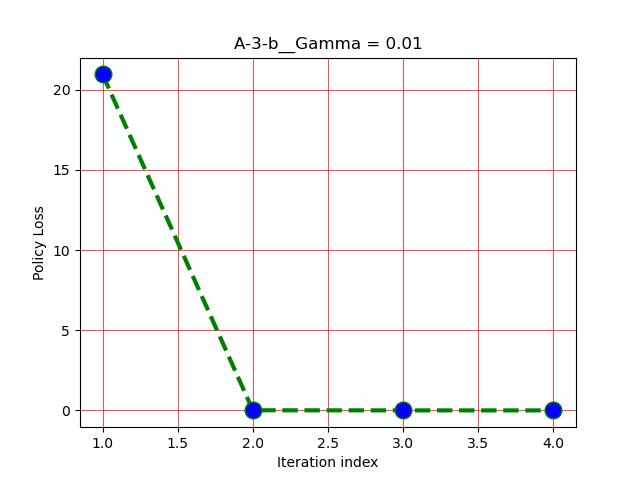




From above graphs we can see that as gamma increases max-norm decrease slowly.

1. For the first time both are giving the same policy but if we repeat algorithm for many times we will find higher discount factor gives better policy.

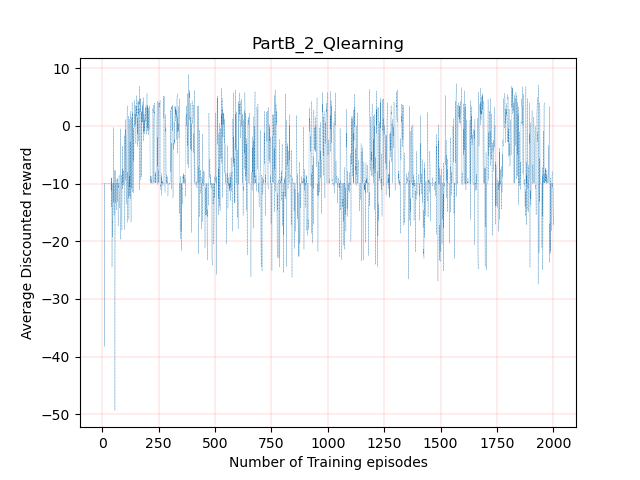
## Policy Iteration

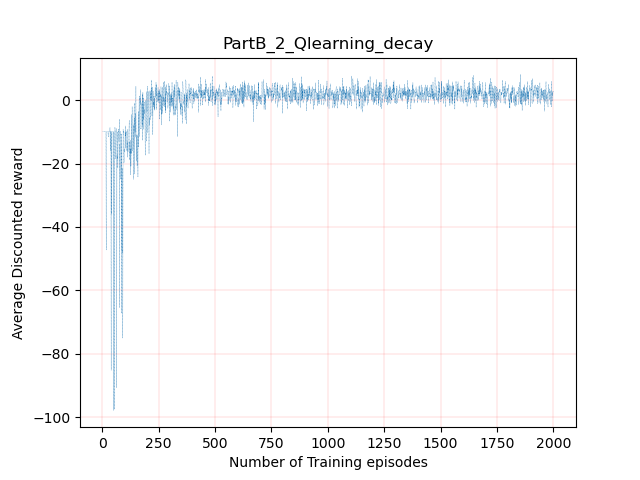


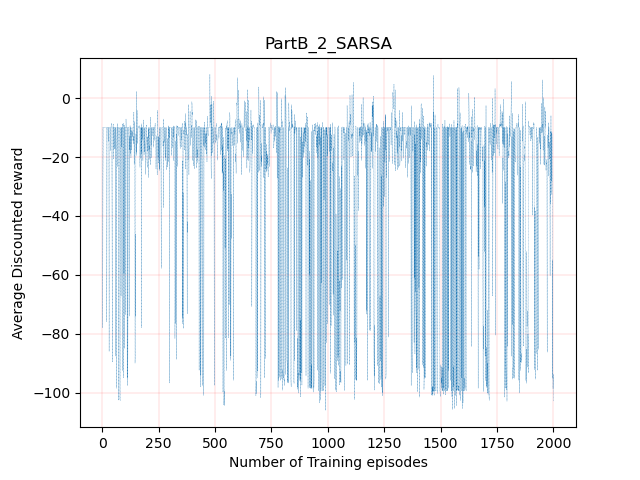
Lower the discount factor faster it converge to zero policy loss.

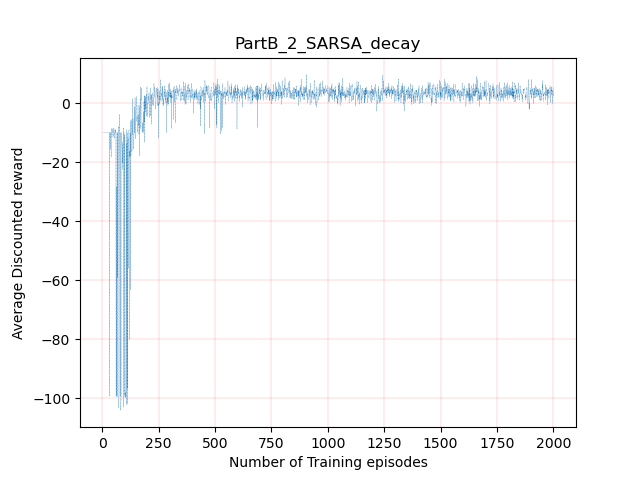
# Part B: Incorporating Learning

## 2)









## 4)

