Assignment 2

CPS 590.1: Decision Making for Robots and Autonomous Systems — Fall 2015

Instructions

This assignment consists of 3 questions. Each of them asks you to implement one or more algorithms, and then answer some questions and/or produce some graphs. Your report should contain both *evidence that your implementation is correct* **and** *answers to the questions and/or the graphs*.

You will be graded on the report itself. Please make it readable and easy to understand. Graphs are good, as are tables with a small set of relevant numbers and the best performers for each category in bold. Large tables of numbers (i.e., *data dumps*) are bad. Nice visualizations of value functions and policies are good. Etc.

Submission: Email your PDF and a .zip file containing all your code to `gdk@cs.duke.edu`.
Due date: December 7th 2015

Question 1: Hierarchical Reinforcement Learning (30 points)

Implement Sarsa (you can modify your previous code—use $\lambda = 0$) for the Taxi domain, using the following sets of options:

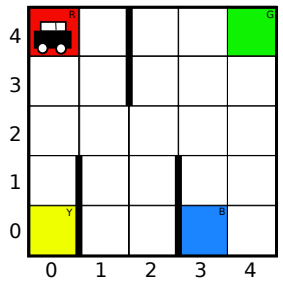
1. Options to move the Taxi to each of the four depots, executable when the taxi is not already there.
2. Options to move the *passenger* to each of the four depots, executable when the passenger is not already there (picking them up if necessary, and dropping them off when they arrive).

Compare the performance of flat (i.e., no options) learning, learning with the first set of options, and learning with the second set of options, by graphing the return obtained per episode for all three methods. Note that you should use $\gamma = 1$, so you do not need to worry about discounting according to the number of steps each options takes to execute.

Note that for this question the options are given: you should write code to express their initiation sets and termination conditions, and to simulate their policies.

Question 2: POMDPs (30 points)

Consider a simpler version of the Taxi problem (shown below for reference). The taxi is only concerned with navigating from one depot to another; it is not concerned with the location of the passenger, or whether or not they are in the taxi. Its only actions are to move in each of the four usual directions, which are stochastic, as before, and its state consists of its x and y position.



However, the Taxi cannot observe its x and y position directly—it is in a POMDP, not an MDP. Instead it observes four binary variables, each representing whether or not there is currently a wall to each side. (For example, at

position $(x = 1, y = 0)$ it would observe $o = (up = 0, down = 1, left = 1, right = 0)$, because it is next to an internal wall, and there are walls surrounding the grid. However, this observation function is noisy, in that it flips the value of a randomly selected variable in the observation vector with probability 0.2.

Write code to estimate the taxi's belief over its state (x and y location) given a sequence of observations and actions. To test it, you should hand-code a policy that gets the agent from any state to any depot. Run a few episodes by choosing a start and goal depot, and at each step pick the most likely state (breaking ties randomly) from the belief distribution, and moving as if the taxi were in that state.

Question 3: Kalman Filters (30 points)

Consider an airplane flying in a two dimensional plane. Its state is described by its x and y positions in the plane as well as its velocity in each direction, $\dot{x}$ and $\dot{y}$. It has a control input, $u = [\delta\dot{x}, \delta\dot{y}]$, which is added to the velocity variables at each timestep.

a Implement the airplane dynamics using a linear system, suitable for use in a Kalman filter. Implement a simple policy (e.g., a sine wave over the elements of u) and graph the airplane's position over time.

b Add Gaussian noise to the control, as in the Kalman filter formulation, and graph the plane's position over time for the same control policy as in part (a).

c Create a noisy observation function by adding Gaussian noise to the airplane's position and velocity. Plot the noisy samples along with the true trajectory of the airplane.

d Implement a Kalman filter, and show the estimated trajectory of the plane overlaid on its true trajectory. Also show the noisy observations.