RL exam

These are the questions that I remember from 2024 Spring Semester Exam:

1. What is the deadly triad in RL? Explain each
2. Write down Bellman Expectation and Bellman Optimality Equation for Q-value function.
3. Rank RL algorithms in terms of sample efficiency. Choices: Policy gradients, MBRL, DQN
4. Write down Vanilla Policy Gradient formula
5. Write batch constrained policy action formula.
6. Explain two reasons why we need intrinsic exploration in RL.
7. What are the reasons we can't directly apply off-policy algorithms to offline RL problems?
8. Can imitation learning be considered a special case of offline RL?
9. How does PPO and TRPO improve over vanilla policy gradient?
10. What is "Noisy-TV" problem in exploration?
11. What networks are used in deployment of AlphaGo?
12. Which one is better: every visit MC or first visit MC?

Programming questions:

1. Find 3 mistakes in the implementation of Double DQN. It was mostly checking if you know which $Q-$value function to use in which update.
2. Thompson Sampling implementation from scratch.

Overall, it is important to study the course slides and do every single programming assignment on your own. However, I don't think even doing that would help you get perfect 1.0 grade. I assume you have to have some background in RL to answer some of the tricky questions. Oh, by the way, be ready to write down easy-to-remember formulas (Bellman, Policy Gradient) during the exam.

Sample exams

Unfortunately, they do not provide past papers. However, you can search online to get RL exams from other universities. Of course, the content would be different, but still I find solving such exams quite helpful. Here are some samplesL

- https://www.cs.hmc.edu/~rhodes/courses/RL/sp20/handouts/midterm2-solution.pdf
- https://www.cs.hmc.edu/~rhodes/courses/RL/sp20/handouts/final.pdf

Good courses online

There are couple of RL courses online that you should definitely check out:

1. Stanford CS234: https://web.stanford.edu/class/cs234/index.html. This course is more related to the course at FAU. There is another great course (CS285) by the RL godfather Sergey Levine, but it is at a more advanced level: https://rail.eecs.berkeley.edu/deeprlcourse/
2. Foundations of Deep RL by another RL godfather Pieter Abbeel: https://youtube.com/playlist?list=PLwRJQ4m4UJjNymuBM9RdmB3Z9N5-0IlY0&si=4pw9EIzD6VHixKlL. The first four lectures are especially relevant.
3. Introduction to RL by Benjamin Eysenbach: https://ben-eysenbach.github.io/intro-rl/. The written notes are super interesting and well-written. Definitely, worth doing the exercises as well.
4. DeepMind xUCL RL lecture series: https://www.youtube.com/watch?v=TCCjZe0y4Qc&list=PLqYmG7hTraZDVH599EItlEWsUOsJbAodm. They offer great walk-through explanations over some of the algorithms (value iteration, UBC). I find these more intuitive and easier to follow.

Q&A

Question 1 The state space is two-dimensional, and you have prior knowledge that one dimension has a more significant effect on the value function than the other. How would you design the tilings to take advantage of this prior knowledge?
Answer 1: To emphasize generalization across x_1, we can use tilings that are **finer** along x_1 and **coarser** along x_2. This means that each tile will cover a smaller range in x_1 and a larger range in x_2.

Question 2: What does Advantage function measure?
Answer 2: Advantage Function measures how much better or worse taking a particular action a in state s is compared to the average action in that state. Remember $V(s)$ gives the total expected value of that state while $Q(s,a)$ gives a value for taking that action while being in that state.

$$A(s,a) = Q(s,a) - V(s)$$

Question 3: Which one helps our agent faster:
a)information about *invariant actions* on *good states* or
b) information about *good actions* in *challenging states*?
Answer 3: a) is better because it is more informative for guiding exploration and enforces distinction between good and bad actions

Question 4: What are the sources of distribution mismatch in offline RL?

- The learned policy would create a different state-visitation distribution compared to the behavior distribution.

- During policy improvement, the learned policy requires an evaluation of the Q-function on unseen actions(out of distribution). An over-estimation of the Q-value on these out-of-distribution samples lead to learning of a suboptimal policy.(**Extrapolation error**)

Question 5: What is the main idea behind DeepHashing?
Instead of counting high dimensional states(images), we count the latent compressed states. Map a state to a hash code, then count up states visited with that hash code. Encourage visiting states with low count hash codes
Question 6: What is the limitation of prediction error as bonus(ICM)(Curiosity-driven Exploration by self-supervised prediction)?
Even with ICM module(including inverse dynamics instead of autoencoders), the agent will forever be rewarded even though the model cannot improve because it is attracted to noise states, with unpredictable outcomes. ("Noisy TV")
Question 7: What is the main idea behind Go-explore?

There are two problems: detachment and derailment
Detachment is the idea that an agent driven by IM could become detached from the regions that are high with IM and doesn't know how to get back to those regions.(catastrophic forgetting). Go-explore solves this by explicitly storing an archive of promising states visited so that they can be visited and explored later again.
Derailment: idea when an agent has discovered a promising state and it would be beneficial to return to that state and explore from it. However IM causes agents to not want to to return to those states

Question 8: How to reduce variance in Policy Gradients?

- causality(reward to go). Only punish the actions after time step t.
- generalized advantage function: Use advantage function in policy gradient equation

Question 9: What is bootstrapping in RL?
Bootstrapping refers to the use of Q-values as targets themselves when updating Q-values.
Question 10: Why Q learning is off-policy?
Because the target value that it computes to perform updates does not depend on the current policy. This is because the target value uses the maximization operator to choose the action at s'. This is in contrast to policy gradient methods, which require $\log \pi_\theta$ at the current θ.
Question 11: What are the problems in classic Q-learning?
Q-learning suffers from maximization bias. A way to address this issue is by Double Q-learning. We use one of the Q-values to perform the maximization step, and another to perform the evaluation:

$$r + \gamma Q_1(s', argmax_a Q_2(s', a))$$

This breaks the maximization bias.
Question 12: What is the deadly triad in deep RL?

- Function approximation:
- Bootstrapping: refers to the use of a model output as a target to regress towards
- Off-policy learning: refers to performing updates using transitions that are not generated by the policy used to collect samples.
  Tabular Q-learning incorporates already two parts of the deadly triad(bootstrapping, off-policy learning). DQN adds the function approximation !

Question 13 How does DQN tries to counter deadly triad?

1. the use of a replay buffer and experience replay. Minimizes the impact of temporal correlations and allows multiple passes through the same points
2. the use of a target network(that is fixed for some time, only updated infrequently). The target is constantly changing over the course of training. The use of a target network helps mitigate this problem by maintaining a separate network that is used to compute the value-to-go portion of the target. Update it occasionally