CS 285 Homework 5

Jeffrey Cheng

November 21, 2022

Commands used to run each question can be found in the README.md of the submission folder.

1 Part 1

1.1 RND and Random Exploration

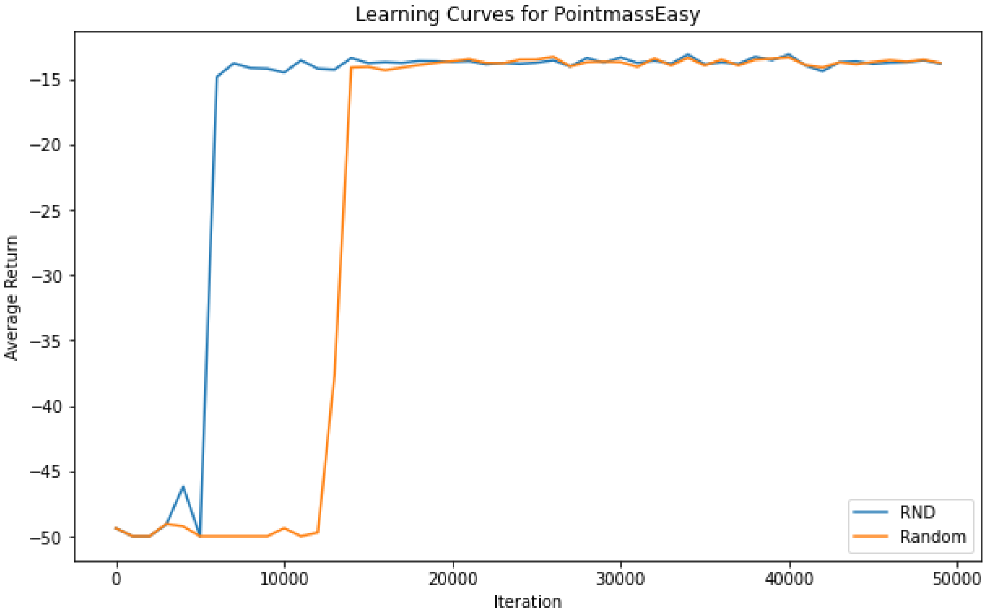
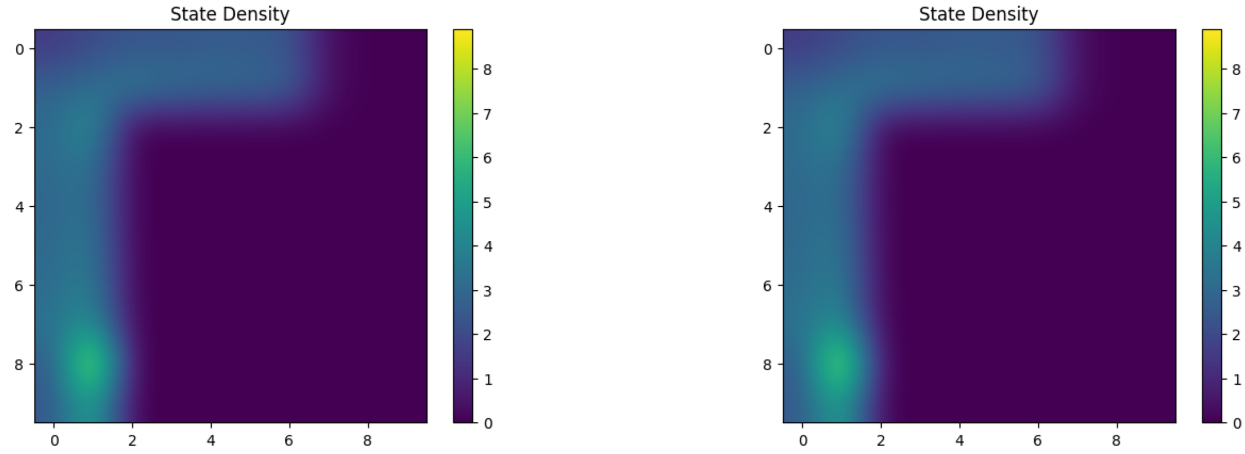


Figure 1: RND and random exploration learning curves in PointmassEasy

RND reaches its optimal performance twice as fast as the random exploration in the easy environment.



(a) Random Exploration State Density

(b) RND Exploration State Density

Figure 2: Exploration State Densities, PointmassEasy

As expected, both random and RND exploration state densities are relatively evenly distributed at the end of 50,000 iterations, as the state space is quite small.

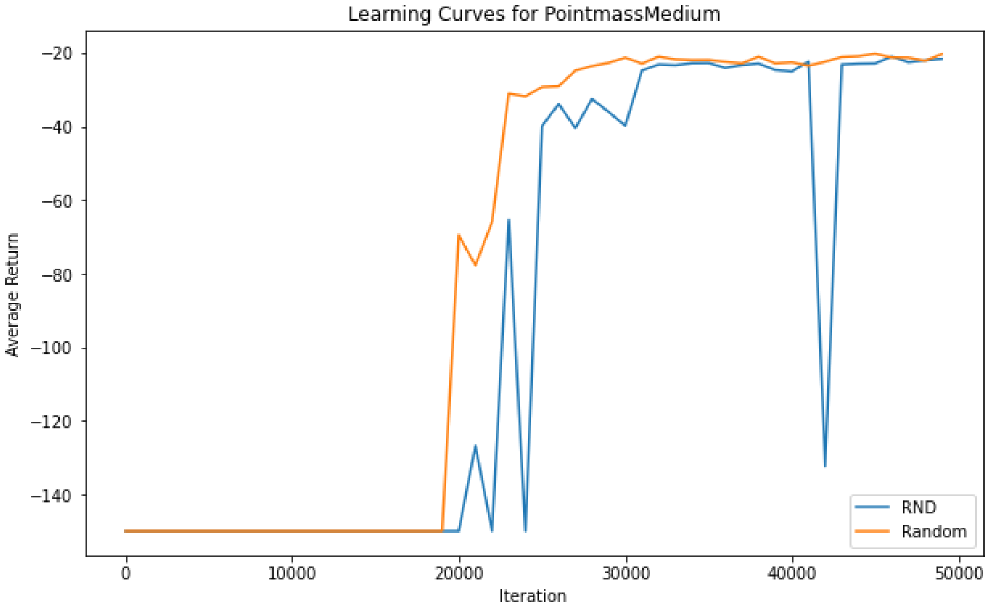
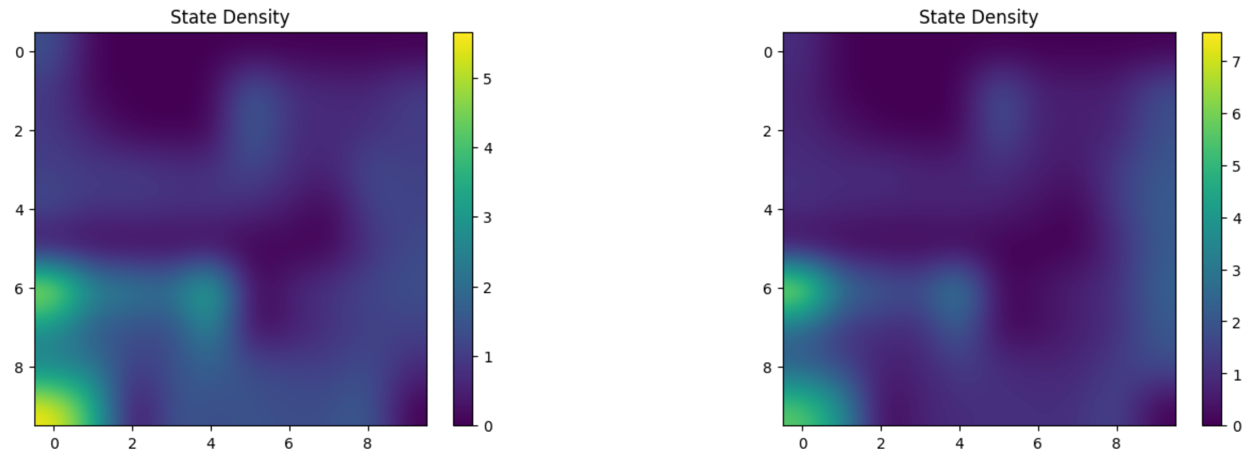


Figure 3: RND and random exploration learning curves in PointmassMedium

Random exploration seems slightly faster and significantly more stable than RND exploration in the medium environment.



(a) Random Exploration State Density

(b) RND Exploration State Density

Figure 4: RND and Random Exploration State Densities, PointmassMedium

The random algorithm appears more evenly distributed than the RND algorithm, which seems to get stuck exploring the long vertical corridor.

1.2 RND and Boltzmann Exploration

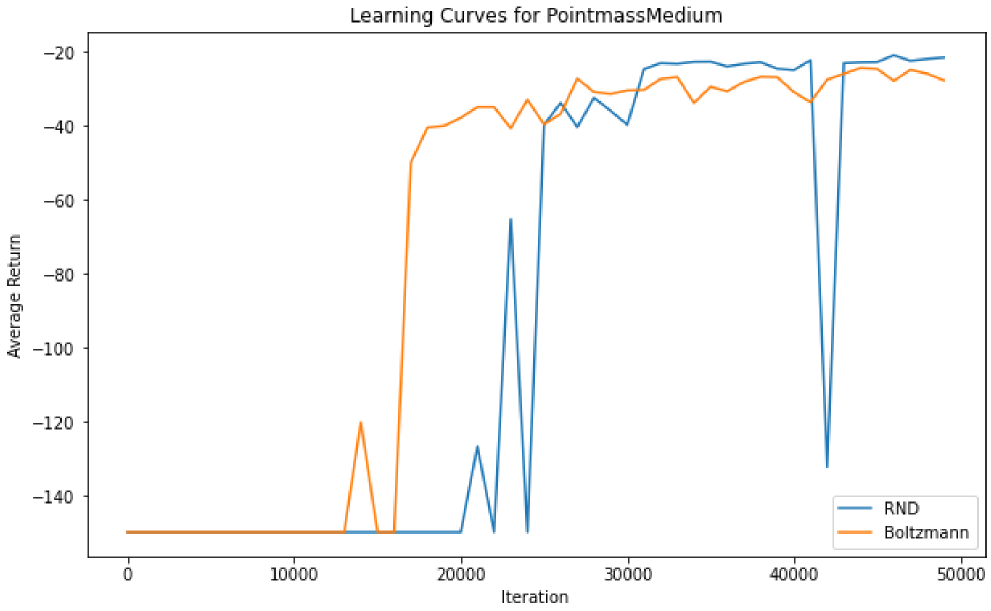
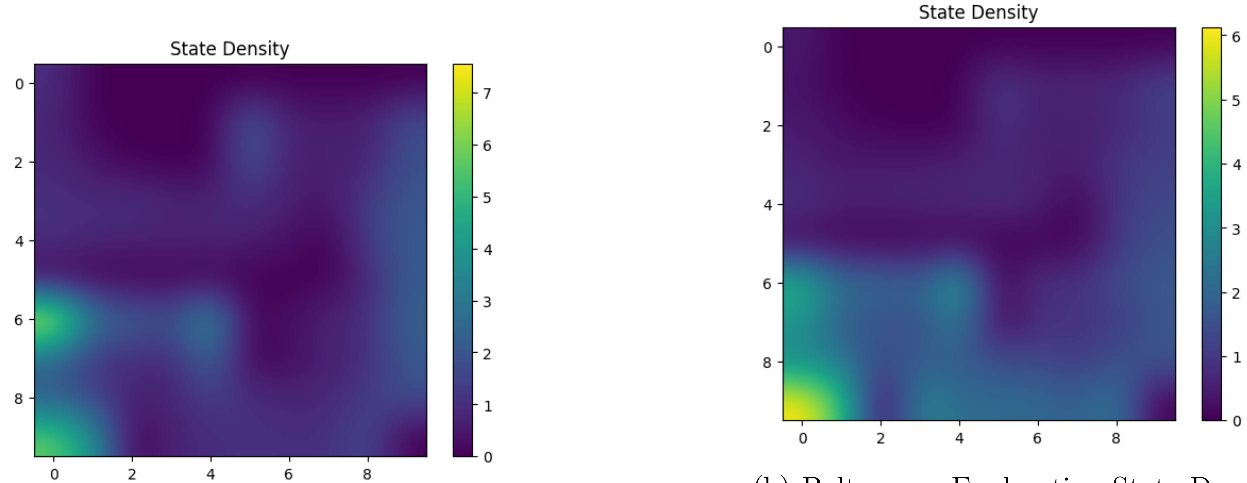


Figure 5: RND and Boltzmann exploration learning curves in PointmassMedium

Boltzmann exploration performs faster than RND exploration and exhibits slightly more stable plateau performance.



(a) RND Exploration State Density

(b) Boltzmann Exploration State Density

Figure 6: RND and Boltzmann Exploration State Densities, PointmassMedium

Here we see that Boltzmann exploration more thoroughly and evenly explores the initial area.

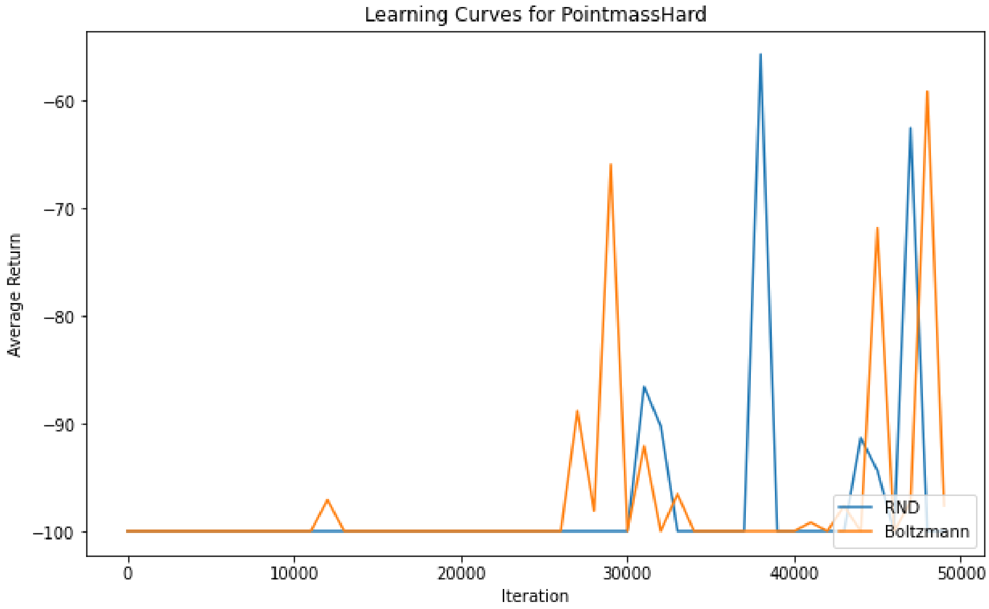
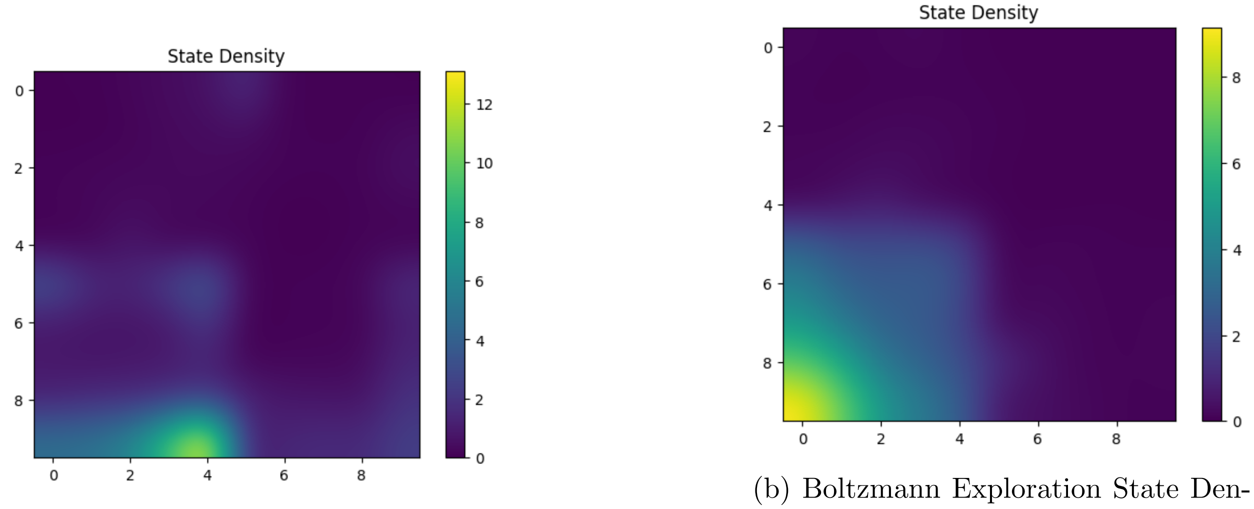


Figure 7: RND and Boltzmann exploration learning curves in PointmassHard

In the hard environment, both RND and Boltzmann struggle to learn.



(a) RND Exploration State Density

(b) Boltzmann Exploration State Density

Figure 8: RND and Boltzmann Exploration State Densities, PointmassHard

Both algorithms get stuck primarily in the initial quadrant of the space. Boltzmann exploration fans out from the starting point where as RND branches off in a direction.

2 Part 2

2.1 DQN and CQL

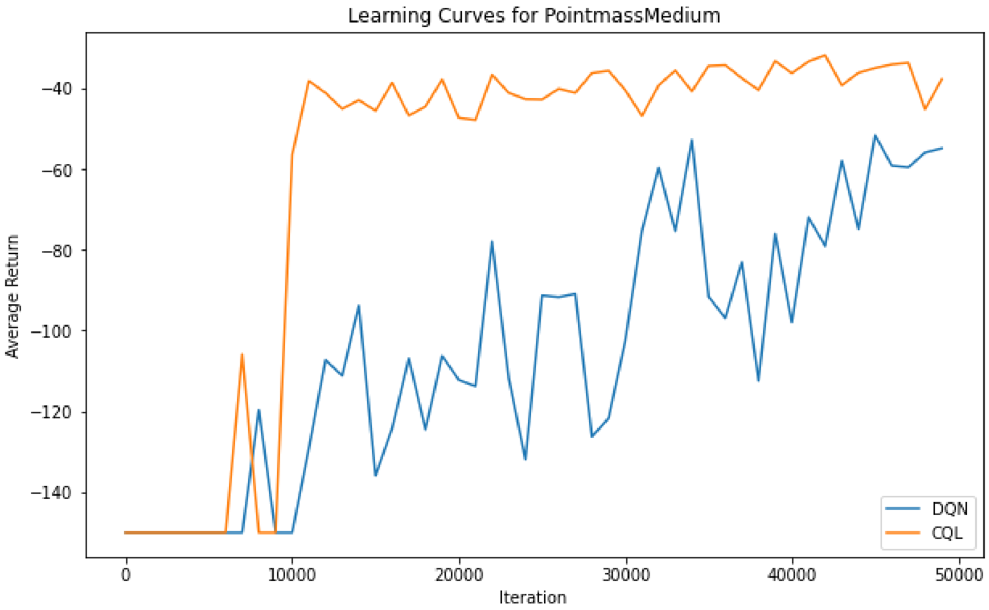
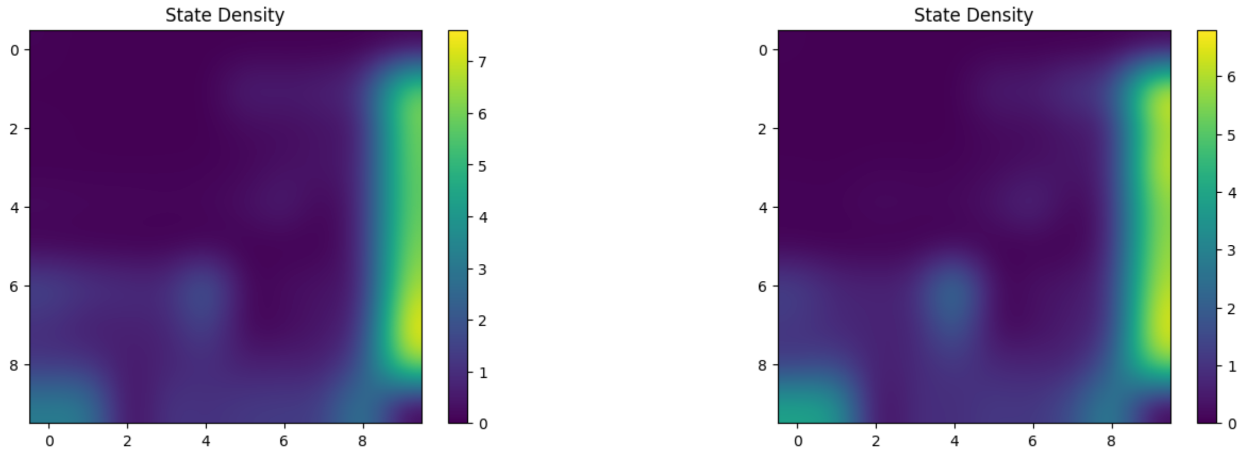


Figure 9: DQN and CQL learning curves in PointmassMedium

CQL seems to learn much quickly than DQN.

However, from the state densities, it seems that both algorithms get stuck in the long vertical corridor.



(a) DQN State Density

(b) CQL State Density

Figure 10: DQN and CQL State Densities, PointmassMedium

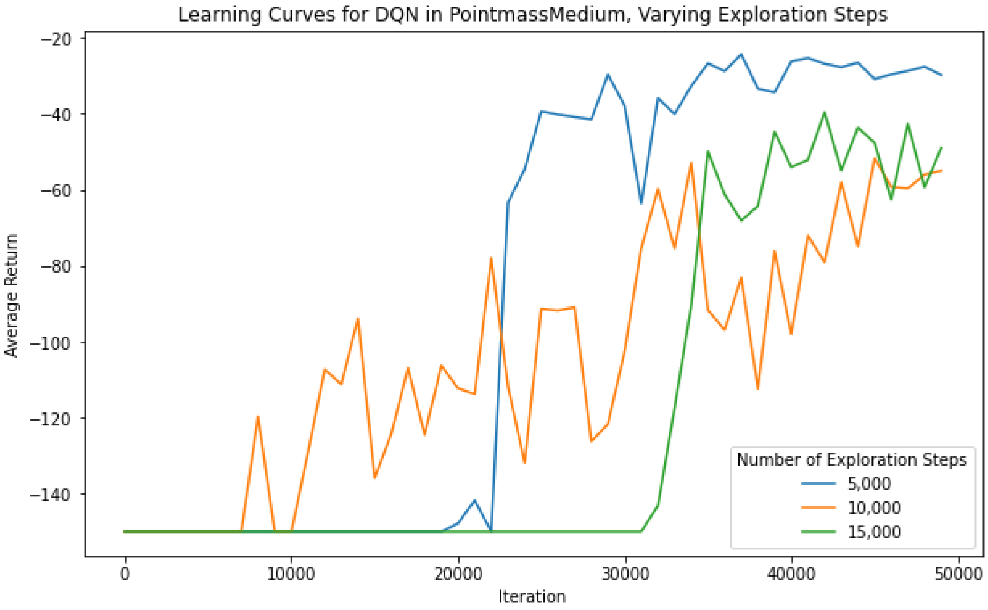


Figure 11: Learning curves for DQN in PointmassMedium, varying exploration steps

For DQN, it actually seems that reducing the number of exploration steps improves end of training performance.

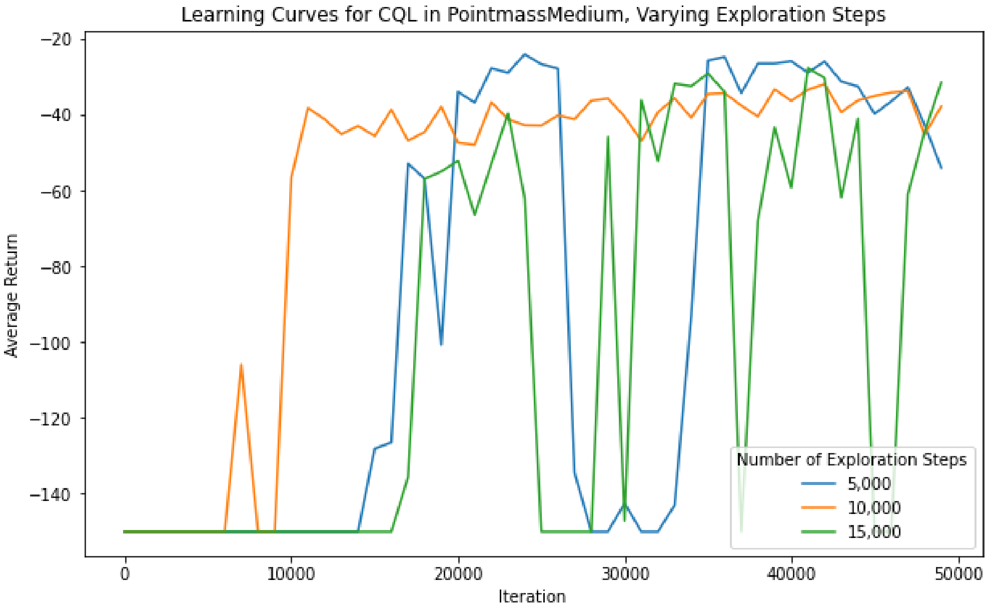


Figure 12: Learning curves for CQL in PointmassMedium, varying exploration steps

For CQL, ignoring the instability, it does not seem that exploration steps affects the end of training performance.

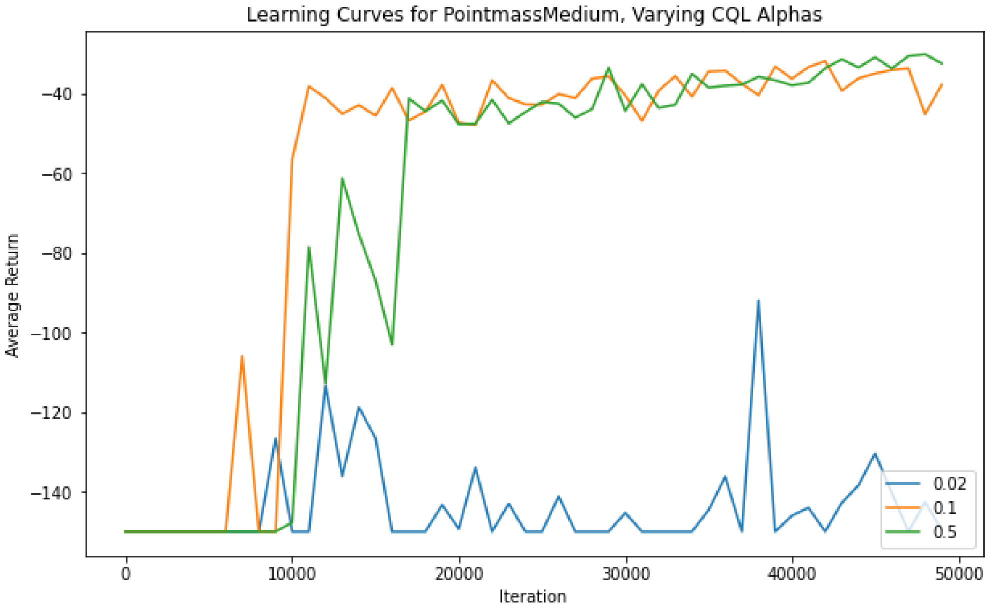


Figure 13: Learning curves for CQL in PointmassMedium, varying α

As for mixtures of DQN and CQL, it seems that having some balanced mixture of DQN and CQL is optimal. Oddly pure DQN performed fine in PointmassMedium, but an α of 0.02 performed very poorly.

3 Part 3

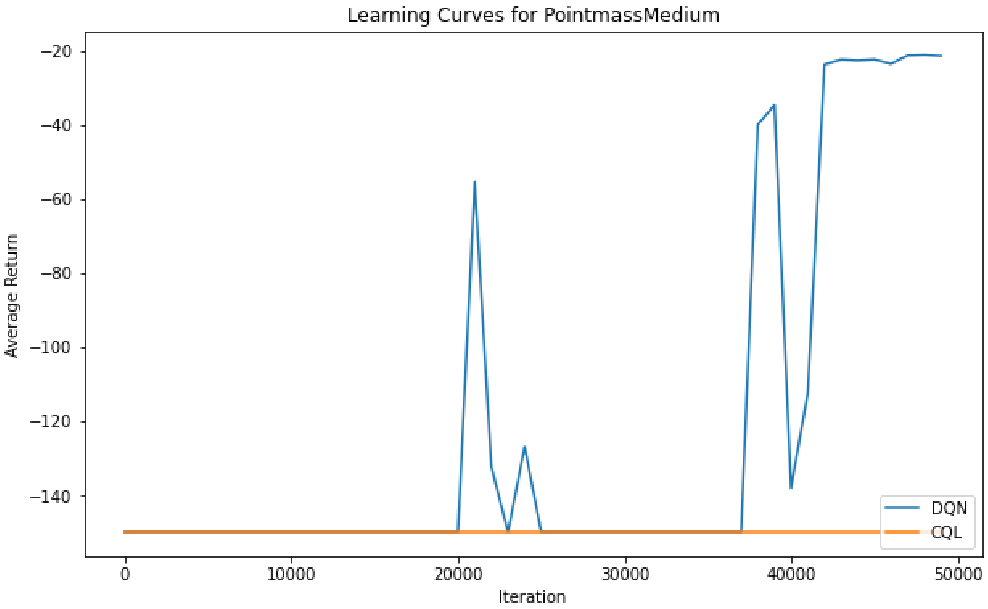
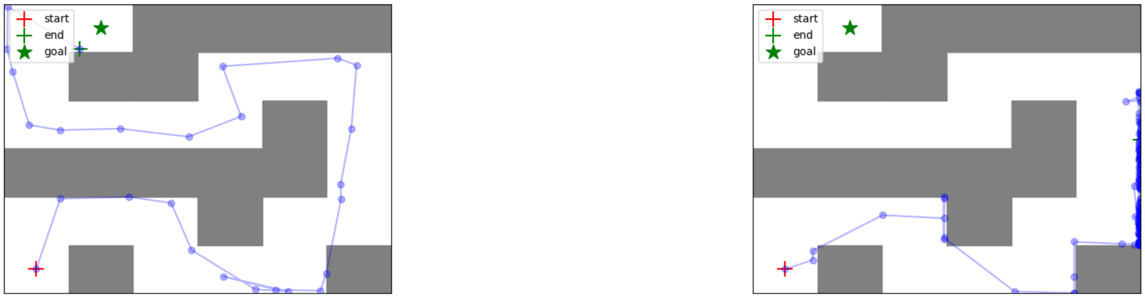


Figure 14: Learning curves for pure DQN and CQL in PointmassMedium, mixed rewards

DQN achieved very unstable results, whereas CQL saw no increase in returns at all.



(a) DQN Last Trajectory

(b) CQL Last Trajectory

Figure 15: DQN and CQL Last Trajectories, PointmassMedium

We can see from the trajectories that CQL got stuck in the vertical corridor. Perhaps this was an unlucky seed?

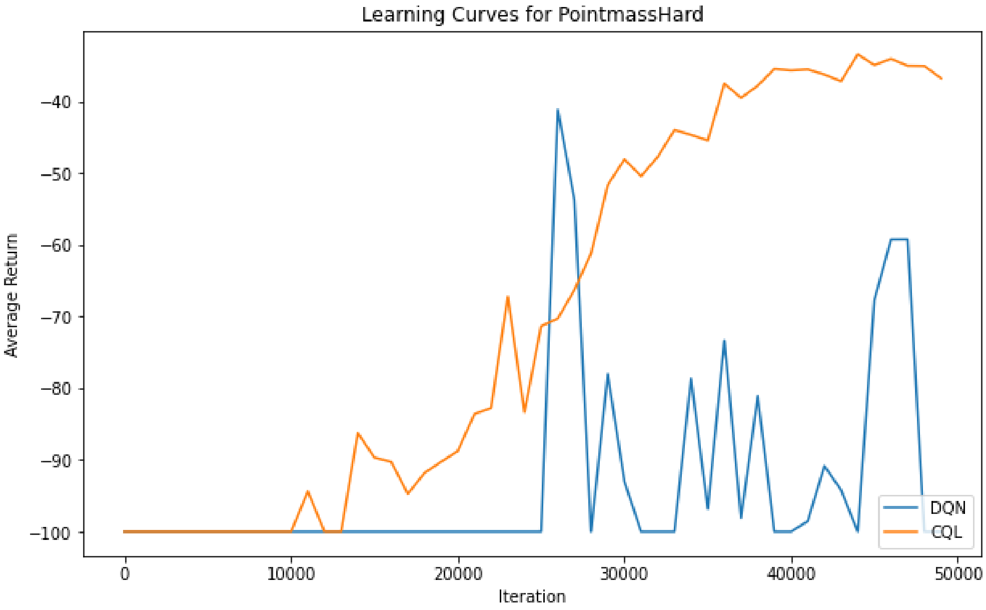


Figure 16: Learning curves for pure DQN and CQL in PointmassHard, mixed rewards

CQL exhibited much better performance over DQN in the hard environment, with relatively stable learning.

4 Part 4

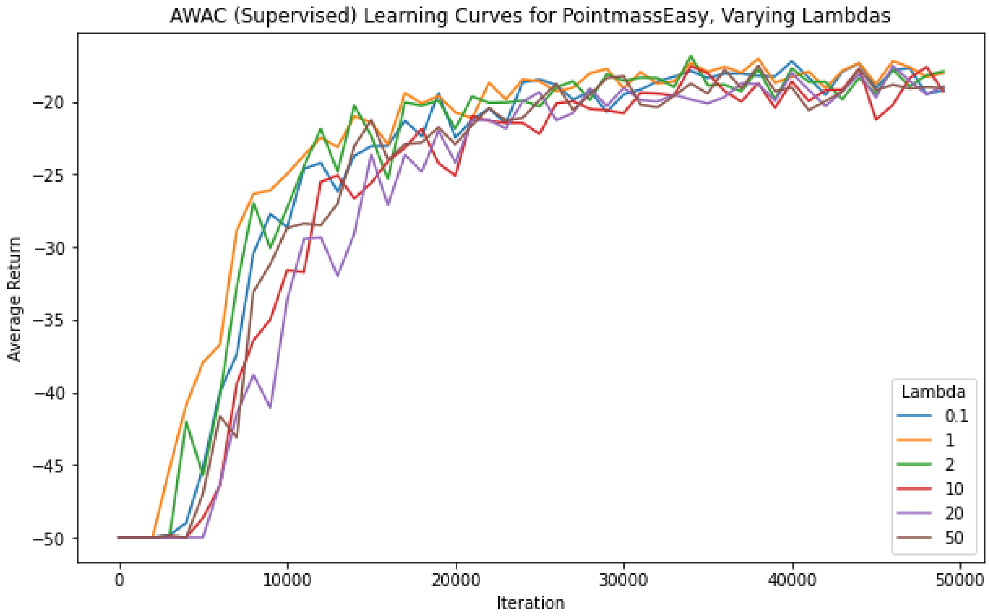


Figure 17: Learning curves for supervised AWAC in PointmassMedium, varying λs

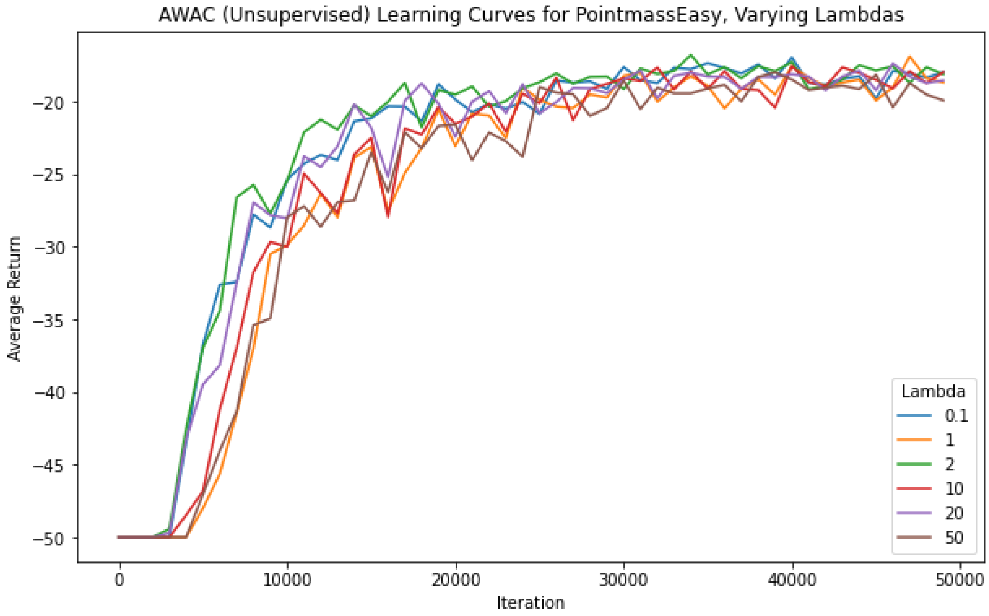


Figure 18: Learning curves for unsupervised AWAC in PointmassMedium, varying λs

Not much difference between runs with different λ values, nor between supervised and unsupervised runs in the easy environment.

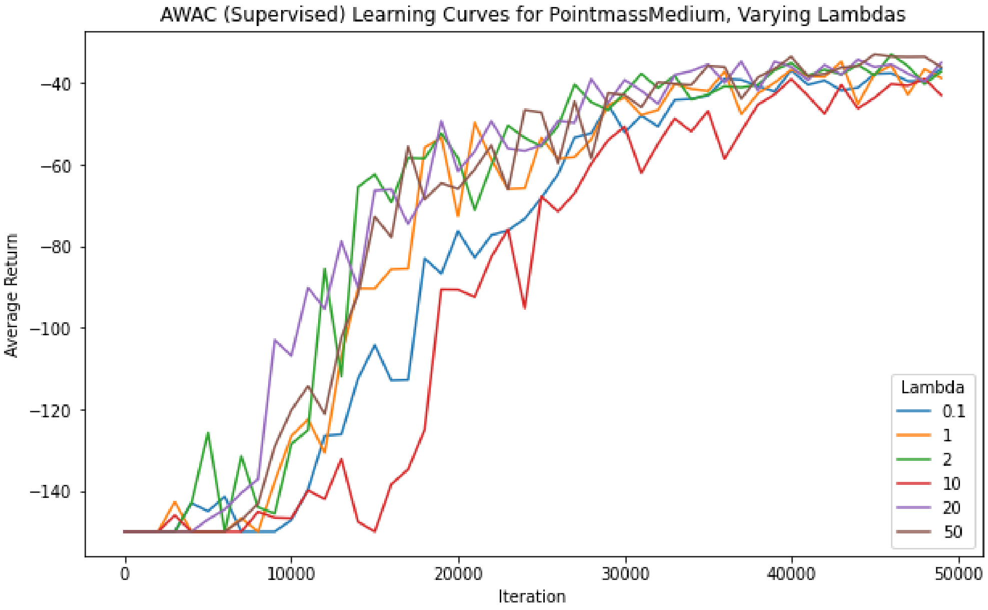


Figure 19: Learning curves for supervised AWAC in PointmassHard, varying λs

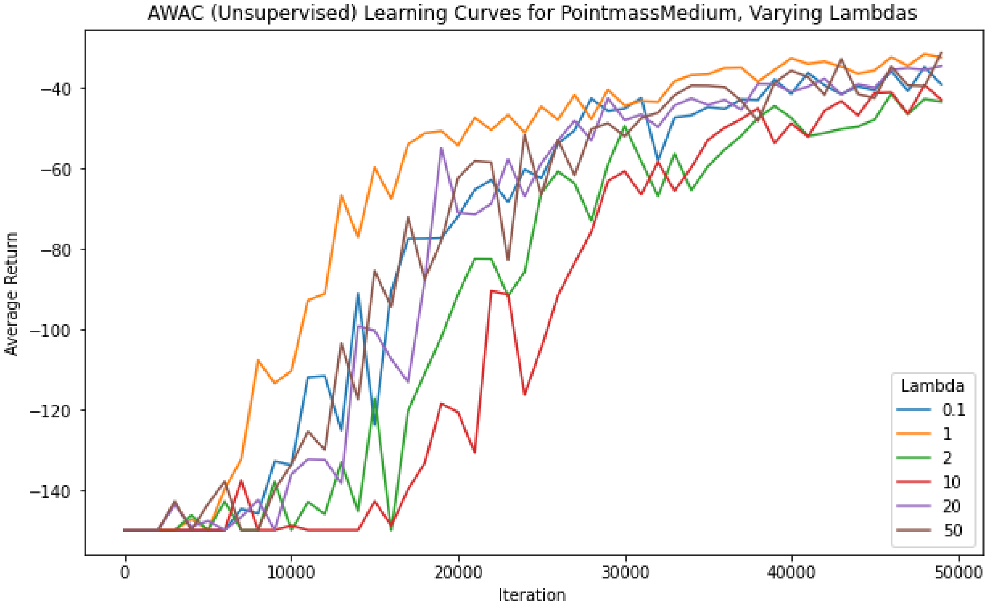


Figure 20: Learning curves for unsupervised AWAC in PointmassHard, varying λs

Slightly more spread learning rates across supervised and unsupervised runs with varying λ values, but the end performance looks roughly the same. There is no obvious rule regarding the parameter that we can observe.

5 Part 5

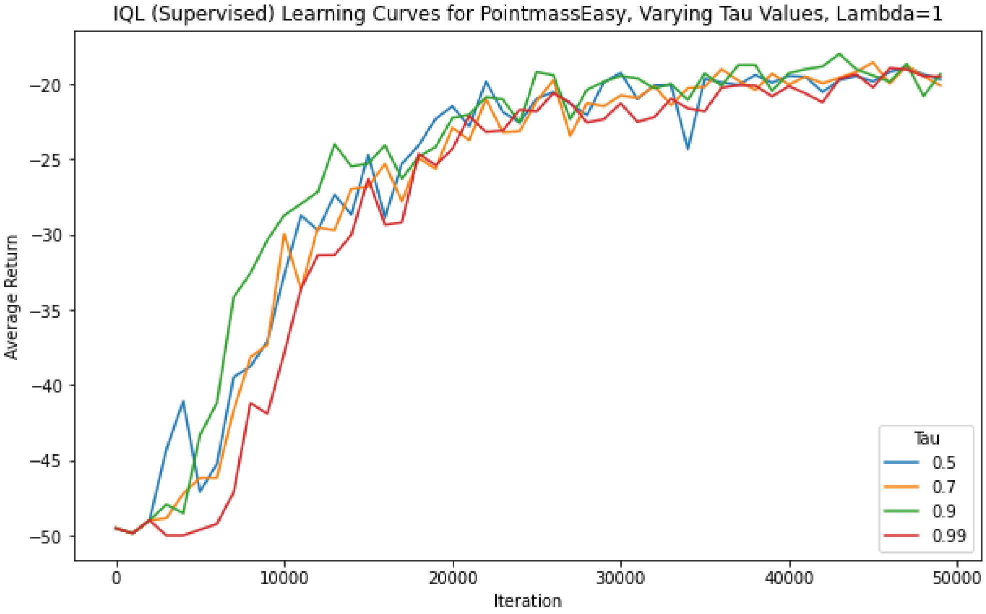


Figure 21: Learning curves for supervised IQL in PointmassMedium, varying τs, $\lambda = 1$

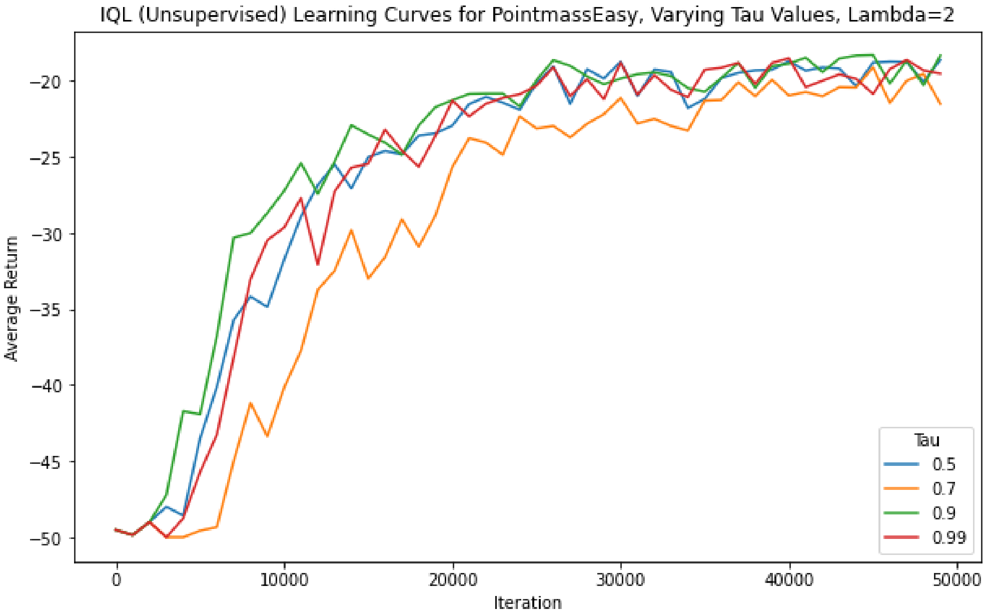


Figure 22: Learning curves for unsupervised IQL in PointmassMedium, varying τs, $\lambda = 2$

Again, not much difference between runs with different expectiles, nor between supervised and unsupervised runs in the easy environment.

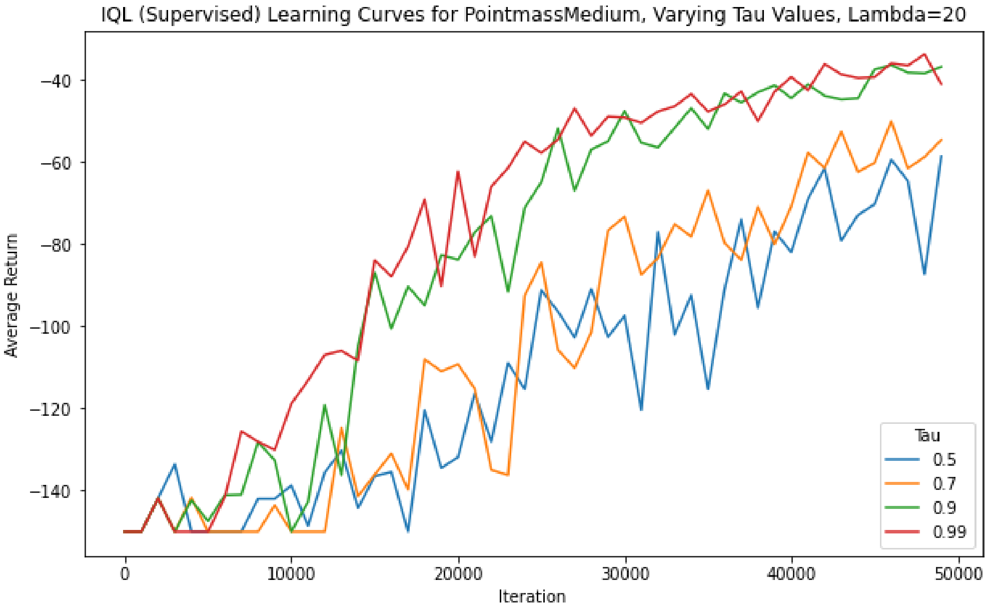


Figure 23: Learning curves for supervised IQL in PointmassHard, varying τs, $\lambda = 20$

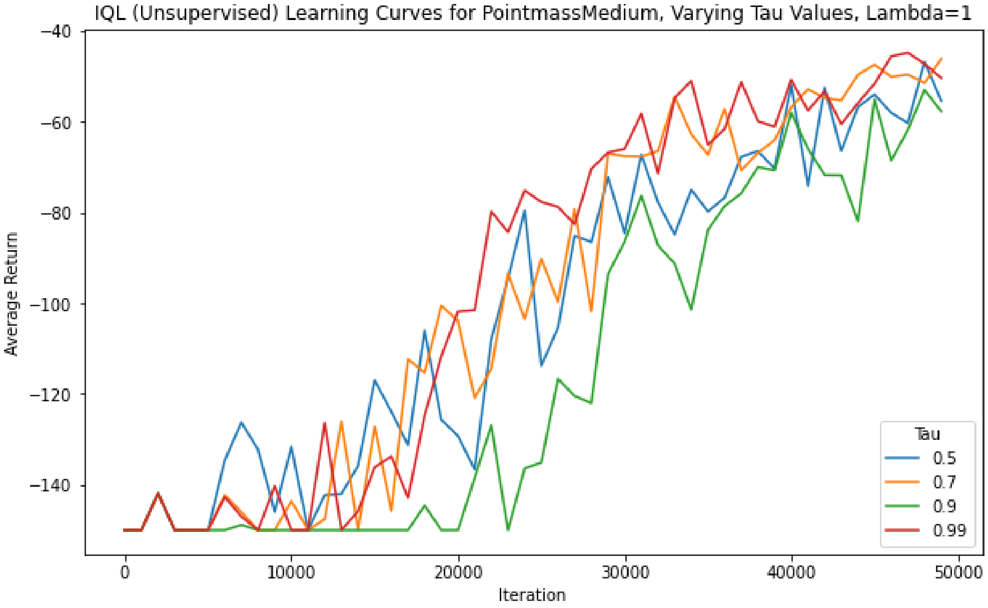


Figure 24: Learning curves for unsupervised IQL in PointmassHard, varying τs, $\lambda = 1$

In the supervised IQL runs in the medium environment, higher valued expectiles tended to perform better overall. The same does not apply to unsupervised IQL runs in the medium environment.

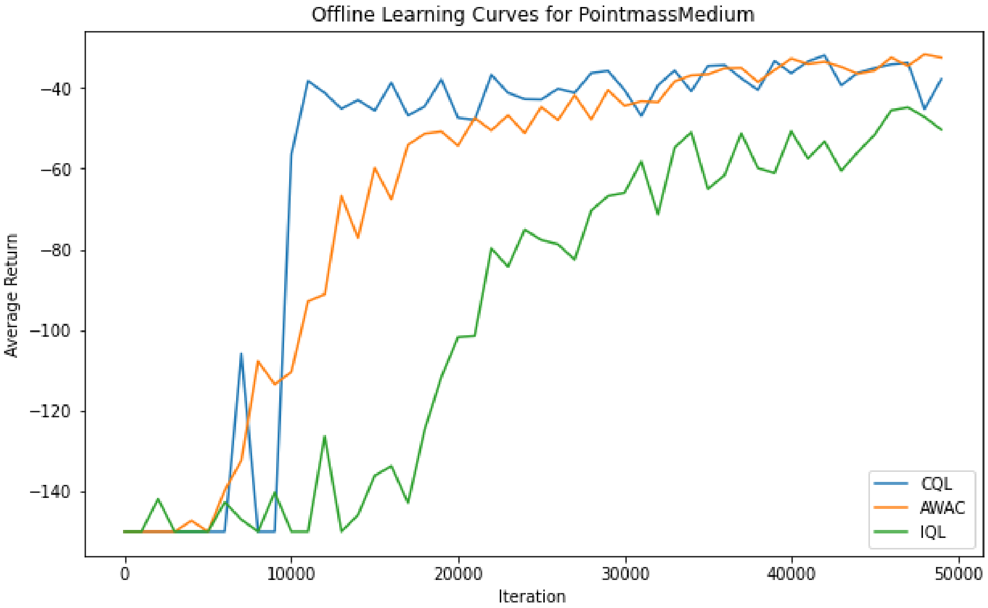


Figure 25: Offline learning curves for CQL, AWAC, and IQL in PointmassMedium

Comparing the best AWAC and IQL runs to the CQL run in Pointmass Medium, it appears that CQL learns the fastest, but AWAC and IQL seem to actually have room to learn. Perhaps if 50,000 more iterations were run, both AWAC and IQL would surpass CQL performance.