Assignment 2: Policy Gradient

Andrew ID: achulawa
Collaborators:
NOTE: Please do **NOT** change the sizes of the answer blocks or plots.

5 Small-Scale Experiments

5.1 Experiment 1 (Cartpole) – [25 points total]

5.1.1 Configurations

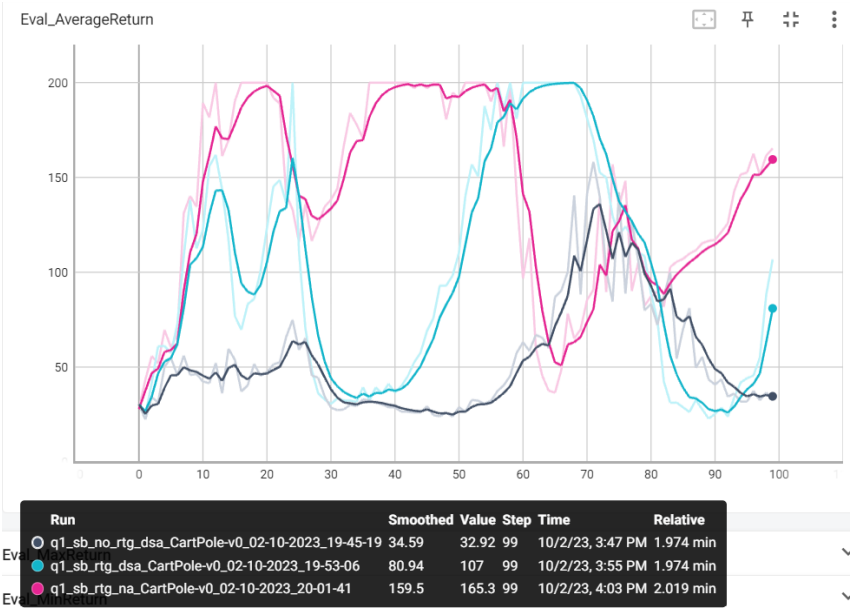
Q5.1.1

```
python rob831/scripts/run_hw2.py --env_name CartPole-v0 -n 100 -b 1000 \
    -dsa --exp_name q1_sb_no_rtg_dsa

python rob831/scripts/run_hw2.py --env_name CartPole-v0 -n 100 -b 1000 \
    -rtg -dsa --exp_name q1_sb_rtg_dsa

python rob831/scripts/run_hw2.py --env_name CartPole-v0 -n 100 -b 1000 \
    -rtg --exp_name q1_sb_rtg_na

python rob831/scripts/run_hw2.py --env_name CartPole-v0 -n 100 -b 5000 \
    -dsa --exp_name q1_lb_no_rtg_dsa

python rob831/scripts/run_hw2.py --env_name CartPole-v0 -n 100 -b 5000 \
    -rtg -dsa --exp_name q1_lb_rtg_dsa

python rob831/scripts/run_hw2.py --env_name CartPole-v0 -n 100 -b 5000 \
    -rtg --exp_name q1_lb_rtg_na
```

5.1.2 Plots

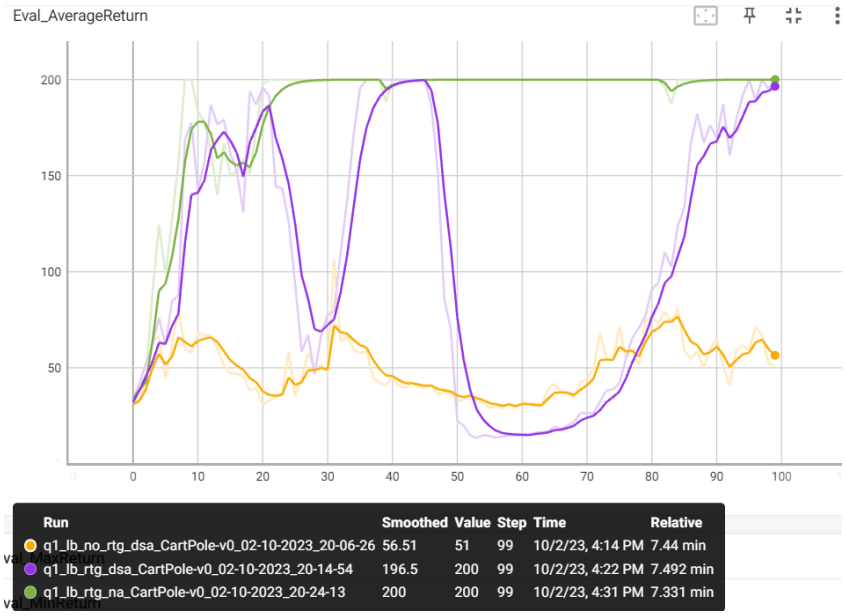
5.1.2.1 Small batch – [5 points]

Q5.1.2.1



5.1.2.2 Large batch – [5 points]

Q5.1.2.2



5.1.3 Analysis

5.1.3.1 Value estimator – [5 points]

Q5.1.3.1

The value estimator using reward-to-go has better performance without advantage standardisation compared to a trajectory centric estimator. This might be because of its lower bias.

5.1.3.2 Advantage standardization – [5 points]

Q5.1.3.2

Advantage standardization can help stabilize and improve the performance of policy gradient methods. Experiments in both small and large batch size perform much better with advantage standardisation taking place.

5.1.3.3 Batch size – [5 points]

Q5.1.3.1

The batch size clearly made a difference. While they did not visibly affect the average reward, larger batch size did decrease the variance in the data being observed. This makes the result more trustworthy, but comes at the cost of more computational time (nearly 4 times).

5.2 Experiment 2 (InvertedPendulum) – [15 points total]

5.2.1 Configurations – [5 points]

Q5.2.1

```
Configuration for some initial runs provided here along with the configuration for the best result.

python rob831/scripts/run_hw2.py --env_name InvertedPendulum-v4 \
    --ep_len 1000 --discount 0.9 -n 100 -l 2 -s 64 -b 1000 -lr 0.001 -rtg --exp_name q2_b1000_r1e-3

python rob831/scripts/run_hw2.py --env_name InvertedPendulum-v4 \
    --ep_len 1000 --discount 0.9 -n 100 -l 2 -s 64 -b 2000 -lr 0.001 -rtg --exp_name q2_b2000_r1e-3

python rob831/scripts/run_hw2.py --env_name InvertedPendulum-v4 \
    --ep_len 1000 --discount 0.9 -n 100 -l 2 -s 64 -b 2000 -lr 0.005 -rtg --exp_name q2_b2000_r5e-3

python rob831/scripts/run_hw2.py --env_name InvertedPendulum-v4 \
    --ep_len 1000 --discount 0.9 -n 100 -l 2 -s 64 -b 2000 -lr 0.008 -rtg --exp_name q2_b2000_r8e-3

python rob831/scripts/run_hw2.py --env_name InvertedPendulum-v4 \
    --ep_len 1000 --discount 0.9 -n 100 -l 2 -s 64 -b 2500 -lr 0.007 -rtg --exp_name q2_b2500_r7e-3

python rob831/scripts/run_hw2.py --env_name InvertedPendulum-v4 \
    --ep_len 1000 --discount 0.9 -n 100 -l 2 -s 64 -b 2500 -lr 0.01 -rtg --exp_name q2_b2500_r10e-3

python rob831/scripts/run_hw2.py --env_name InvertedPendulum-v4 \
    --ep_len 1000 --discount 0.9 -n 100 -l 2 -s 64 -b 5000 -lr 0.008 -rtg --exp_name q2_b5000_r8e-3

python rob831/scripts/run_hw2.py --env_name InvertedPendulum-v4 \
    --ep_len 1000 --discount 0.9 -n 100 -l 2 -s 64 -b 5000 -lr 0.005 -rtg --exp_name q2_b5000_r5e-3

-- Best Configuration --
python rob831/scripts/run_hw2.py --env_name InvertedPendulum-v4 \
    --ep_len 1000 --discount 0.9 -n 100 -l 2 -s 64 -b 2500 -lr 0.008 -rtg --exp_name q2_b2500_r8e-3
```

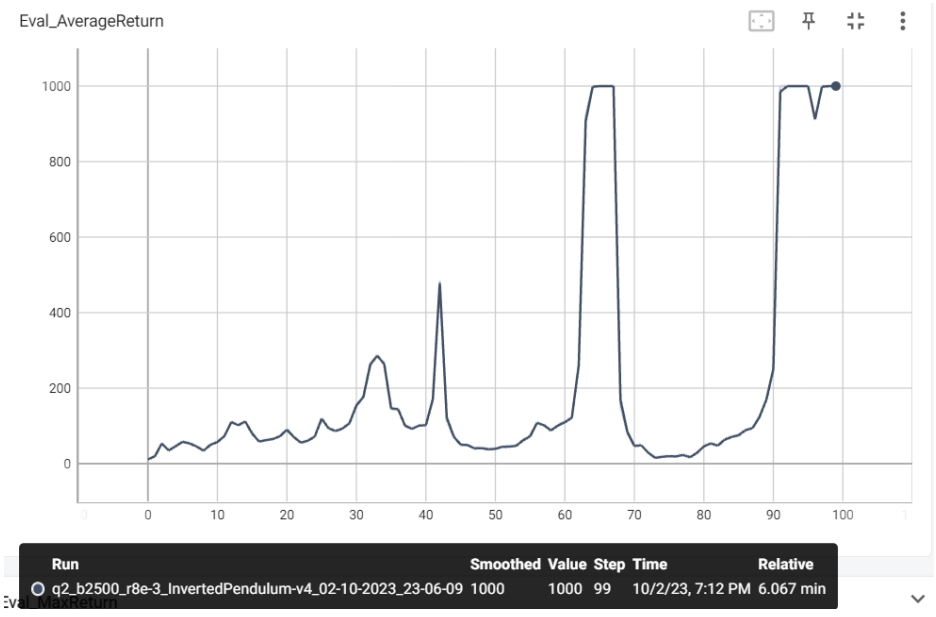
5.2.2 smallest b* and largest r* (same run) – [5 points]

Q5.2.2

Smallest batch size with decent results = 2500
Largest learning rate with decent results = 0.008
While these results show manageable variances, a larger batch size would be preferred to reduce the variance.

5.2.3 Plot – [5 points]

Q5.2.3



7 More Complex Experiments

7.1 Experiment 3 (LunarLander) – [10 points total]

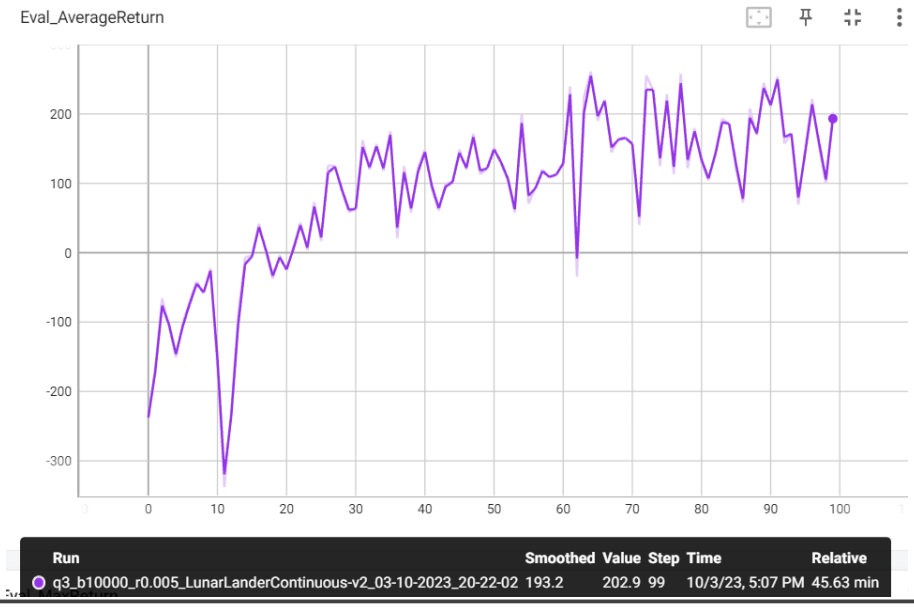
7.1.1 Configurations

Q7.1.1

```
Seed set to 1000

python rob831/scripts/run_hw2.py \
    --env_name LunarLanderContinuous-v4 --ep_len 1000
    --discount 0.99 -n 100 -l 2 -s 64 -b 40000 -lr 0.005 \
    --reward_to_go --nn_baseline --exp_name q3_b40000_r0.005
```

7.1.2 Plot – [10 points]

Q7.1.2



7.2 Experiment 4 (HalfCheetah) – [30 points]

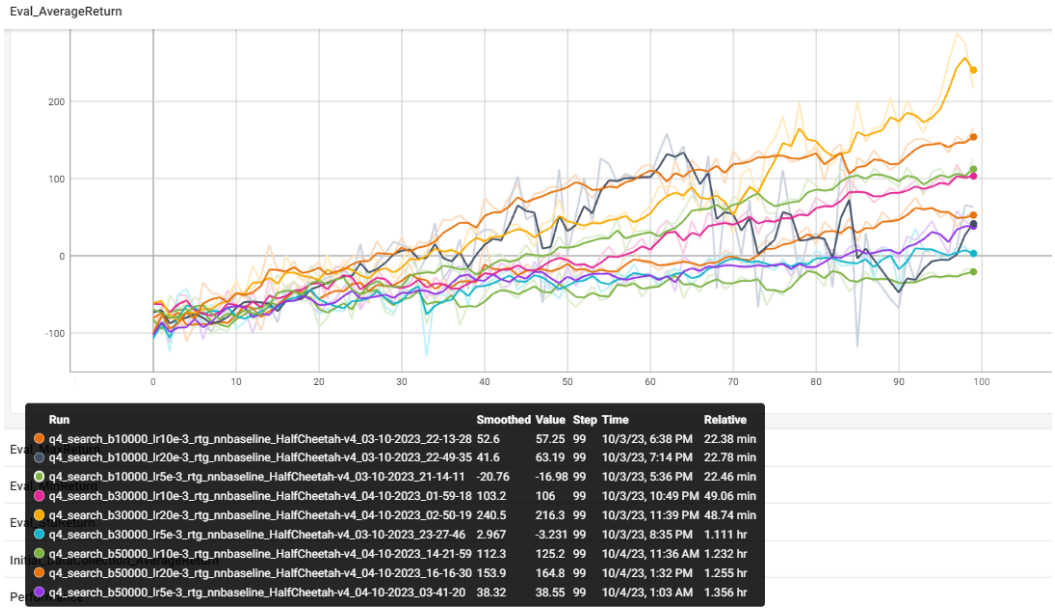
7.2.1 Configurations

Q7.2.1

```
Seed set to 1000

python rob831/scripts/run_hw2.py --env_name HalfCheetah-v4 --ep_len 150 \
    --discount 0.95 -n 100 -l 2 -s 32 -b 10000 -lr 0.005 -rtg --nn_baseline \
    --exp_name q4_search_b10000_lr5e-3_rtg_nnbaseline
python rob831/scripts/run_hw2.py --env_name HalfCheetah-v4 --ep_len 150 \
    --discount 0.95 -n 100 -l 2 -s 32 -b 10000 -lr 0.01 -rtg --nn_baseline \
    --exp_name q4_search_b10000_lr10e-3_rtg_nnbaseline
python rob831/scripts/run_hw2.py --env_name HalfCheetah-v4 --ep_len 150 \
    --discount 0.95 -n 100 -l 2 -s 32 -b 10000 -lr 0.02 -rtg --nn_baseline \
    --exp_name q4_search_b10000_lr20e-3_rtg_nnbaseline
python rob831/scripts/run_hw2.py --env_name HalfCheetah-v4 --ep_len 150 \
    --discount 0.95 -n 100 -l 2 -s 32 -b 30000 -lr 0.005 -rtg --nn_baseline \
    --exp_name q4_search_b30000_lr5e-3_rtg_nnbaseline
python rob831/scripts/run_hw2.py --env_name HalfCheetah-v4 --ep_len 150 \
    --discount 0.95 -n 100 -l 2 -s 32 -b 30000 -lr 0.01 -rtg --nn_baseline \
    --exp_name q4_search_b30000_lr10e-3_rtg_nnbaseline
python rob831/scripts/run_hw2.py --env_name HalfCheetah-v4 --ep_len 150 \
    --discount 0.95 -n 100 -l 2 -s 32 -b 30000 -lr 0.02 -rtg --nn_baseline \
    --exp_name q4_search_b30000_lr20e-3_rtg_nnbaseline
python rob831/scripts/run_hw2.py --env_name HalfCheetah-v4 --ep_len 150 \
    --discount 0.95 -n 100 -l 2 -s 32 -b 50000 -lr 0.005 -rtg --nn_baseline \
    --exp_name q4_search_b50000_lr5e-3_rtg_nnbaseline
python rob831/scripts/run_hw2.py --env_name HalfCheetah-v4 --ep_len 150 \
    --discount 0.95 -n 100 -l 2 -s 32 -b 50000 -lr 0.01 -rtg --nn_baseline \
    --exp_name q4_search_b50000_lr10e-3_rtg_nnbaseline
python rob831/scripts/run_hw2.py --env_name HalfCheetah-v4 --ep_len 150 \
    --discount 0.95 -n 100 -l 2 -s 32 -b 50000 -lr 0.02 -rtg --nn_baseline \
    --exp_name q4_search_b50000_lr20e-3_rtg_nnbaseline
```

7.2.2 Plot – [10 points]

Q7.2.2



Run	Smoothed	Value	Step	Time	Relative
q4_search_b10000_lr10e-3_rtg_nnbaseline_HalfCheetah-v4_03-10-2023_22-13-28	52.6	57.25	99	10/3/23, 6:38 PM	22.38 min
q4_search_b10000_lr20e-3_rtg_nnbaseline_HalfCheetah-v4_03-10-2023_22-49-35	41.6	63.19	99	10/3/23, 7:14 PM	22.78 min
q4_search_b10000_lr5e-3_rtg_nnbaseline_HalfCheetah-v4_03-10-2023_21-14-11	-20.76	-16.98	99	10/3/23, 5:36 PM	22.46 min
q4_search_b30000_lr10e-3_rtg_nnbaseline_HalfCheetah-v4_04-10-2023_01-59-18	103.2	106	99	10/3/23, 10:49 PM	49.06 min
q4_search_b30000_lr20e-3_rtg_nnbaseline_HalfCheetah-v4_04-10-2023_02-50-19	240.5	216.3	99	10/3/23, 11:39 PM	48.74 min
q4_search_b30000_lr5e-3_rtg_nnbaseline_HalfCheetah-v4_03-10-2023_23-27-46	2.967	-3.231	99	10/3/23, 8:35 PM	1.111 hr
q4_search_b50000_lr10e-3_rtg_nnbaseline_HalfCheetah-v4_04-10-2023_14-21-59	112.3	125.2	99	10/4/23, 11:36 AM	1.232 hr
q4_search_b50000_lr20e-3_rtg_nnbaseline_HalfCheetah-v4_04-10-2023_16-16-30	153.9	164.8	99	10/4/23, 1:32 PM	1.255 hr
q4_search_b50000_lr5e-3_rtg_nnbaseline_HalfCheetah-v4_04-10-2023_03-41-20	38.32	38.55	99	10/4/23, 1:03 AM	1.356 hr

7.2.3 Optimal b* and r* – [3 points]

Q7.2.3

From the nine experiments conducted:
Optimal batch size = 30000
Optimal learning rate = 0.02

7.2.4 Describe how b* and r* affect task performance – [7 points]

Q7.2.4

The batch size determines how many experiences are used in each update step. Larger batches lead to more stable training and better generalization, but can be computationally intensive. On the other hand, smaller batches introduce more randomness, aiding in exploration. The learning rate, on the other hand, controls the step size during parameter updates. It's a delicate balance; too high a learning rate can cause the optimization process to diverge, while too low a learning rate can result in slow convergence.

7.2.5 Configurations with optimal b* and r* – [3 points]

Q7.2.5

```
Seed set to 1000

python rob831/scripts/run_hw2.py --env_name HalfCheetah-v4 --ep_len 150 \
    --discount 0.95 -n 100 -l 2 -s 32 -b 30000 -lr 0.02 \
    --exp_name q4_b30000_r20e-3

python rob831/scripts/run_hw2.py --env_name HalfCheetah-v4 --ep_len 150 \
    --discount 0.95 -n 100 -l 2 -s 32 -b 30000 -lr 0.02 -rtg \
    --exp_name q4_b30000_r20e-3_rtg

python rob831/scripts/run_hw2.py --env_name HalfCheetah-v4 --ep_len 150 \
    --discount 0.95 -n 100 -l 2 -s 32 -b 30000 -lr 0.02 --nn_baseline \
    --exp_name q4_b30000_r20e-3_nnbaseline

python rob831/scripts/run_hw2.py --env_name HalfCheetah-v4 --ep_len 150 \
    --discount 0.95 -n 100 -l 2 -s 32 -b 30000 -lr 0.02 -rtg --nn_baseline \
    --exp_name q4_b30000_r20e-3_rtg_nnbaseline
```

7.2.6 Plot for four runs with optimal b* and r* – [7 points]

Q7.2.6

8 Implementing Generalized Advantage Estimation

8.1 Experiment 5 (Hopper) – [20 points]

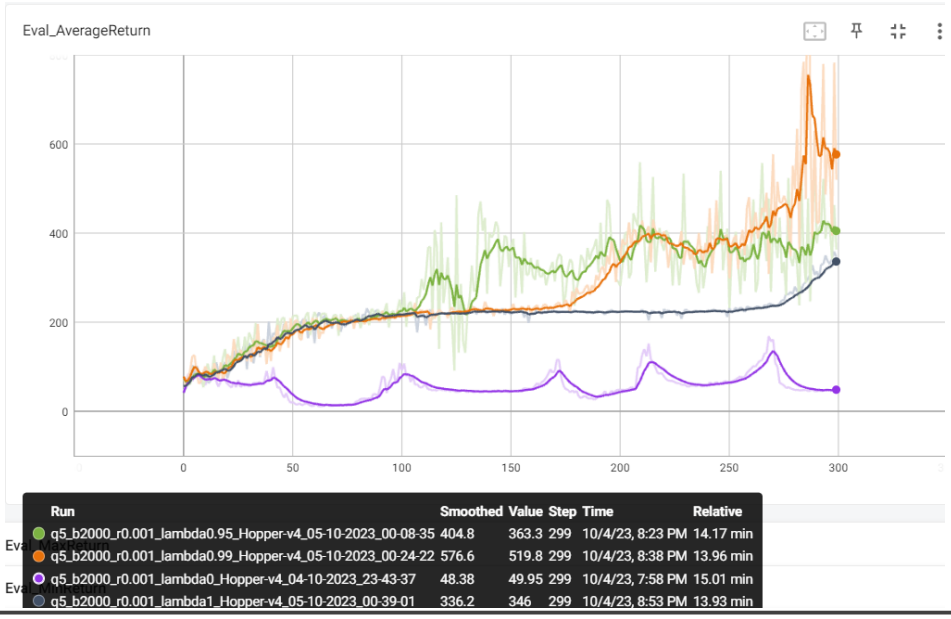
8.1.1 Configurations

Q8.1.1

```
Seed set to 1000
python rob831/scripts/run_hw2.py \
    --env_name Hopper-v4 --ep_len 1000 --discount 0.99 -n 300 -l 2 -s 32 -b 2000 -lr 0.001 \
    --reward_to_go --nn_baseline --action_noise_std 0.5 --gae_lambda 0 --exp_name q5_b2000_r0.001_lambda0

python rob831/scripts/run_hw2.py \
    --env_name Hopper-v4 --ep_len 1000 --discount 0.99 -n 300 -l 2 -s 32 -b 2000 -lr 0.001 \
    --reward_to_go --nn_baseline --action_noise_std 0.5 --gae_lambda 0.95 --exp_name q5_b2000_r0.001_lambda0.95

python rob831/scripts/run_hw2.py \
    --env_name Hopper-v4 --ep_len 1000 --discount 0.99 -n 300 -l 2 -s 32 -b 2000 -lr 0.001 \
    --reward_to_go --nn_baseline --action_noise_std 0.5 --gae_lambda 0.99 --exp_name q5_b2000_r0.001_lambda0.99

python rob831/scripts/run_hw2.py \
    --env_name Hopper-v4 --ep_len 1000 --discount 0.99 -n 300 -l 2 -s 32 -b 2000 -lr 0.001 \
    --reward_to_go --nn_baseline --action_noise_std 0.5 --gae_lambda 1 --exp_name q5_b2000_r0.001_lambda1
```

8.1.2 Plot – [13 points]

Q8.1.2



8.1.3 Describe how λ affects task performance – [7 points]

Q8.1.3

A higher lambda emphasizes future rewards, reducing bias but potentially increasing variance. A lower lambda does the opposite, affecting the agent's preference for short-term versus long-term rewards and impacting task performance accordingly. Lambda value of 0.99 seems to work best in this case of the Hopper environment, resulting in a reward of around 500.

9 Bonus! (optional)

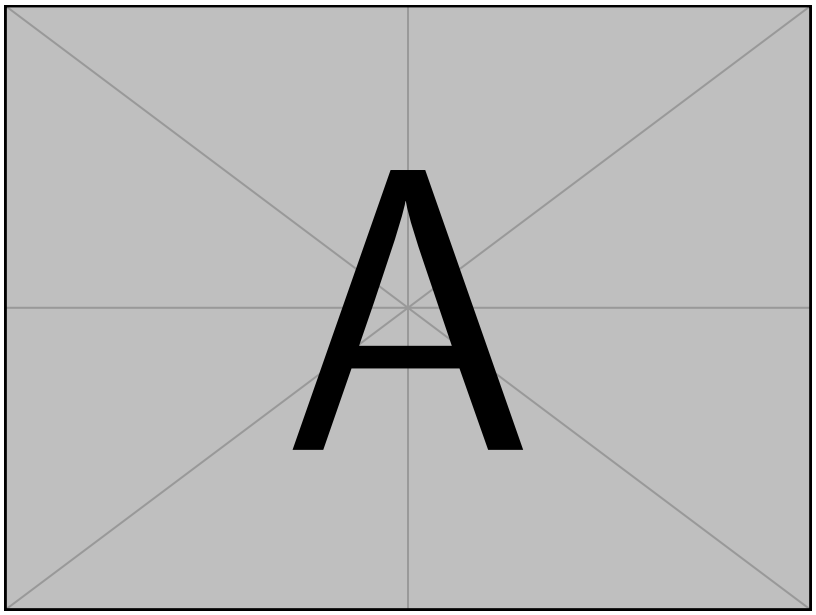
9.1 Parallelization – [15 points]

Q9.1

Reduction in training time by 7.5The multiprocessing library in Python was utilised for this problem, and the modified result was tested on the solution of experiment (Inverted Pendulum environment). The result saw a faster iteration speed when using two threads. I was unable to get successful runs when using more threads, but theoretically that should decrease the speed. The modified file is stored as rl_trainer_speedupdate.py. Change the name to rl_trainer.py and replace the original final name with rl_trainer_original.py to run the script.

9.2 Multiple gradient steps – [5 points]

Q9.1




```
python rob831/scripts/run_hw2.py \
```