Assignment 2: Policy Gradient

Andrew ID: `shriishs`
Collaborators: `None`
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

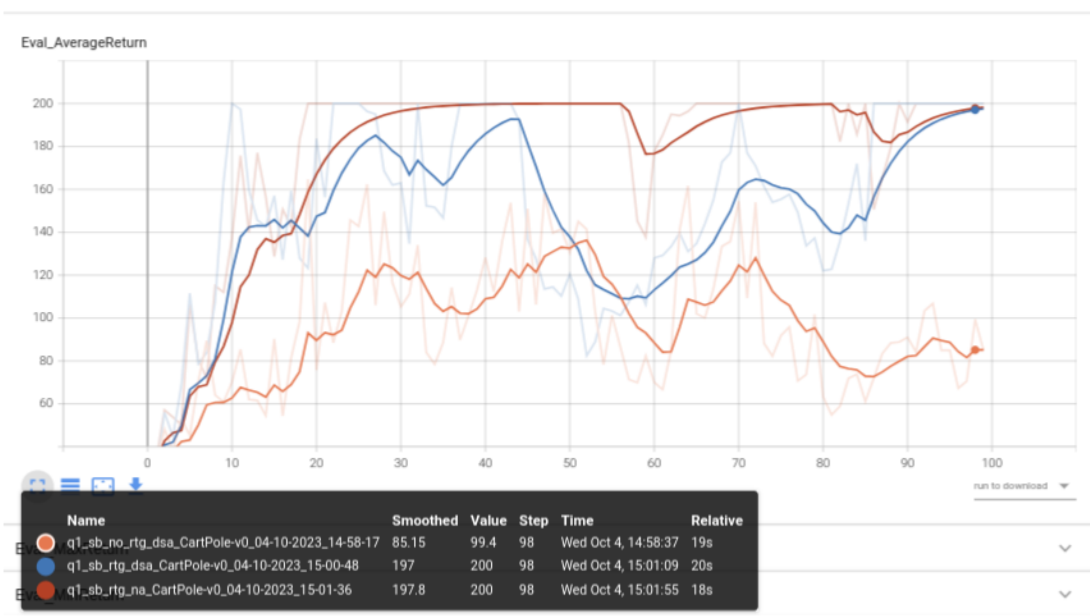
Q5.1.1

```
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

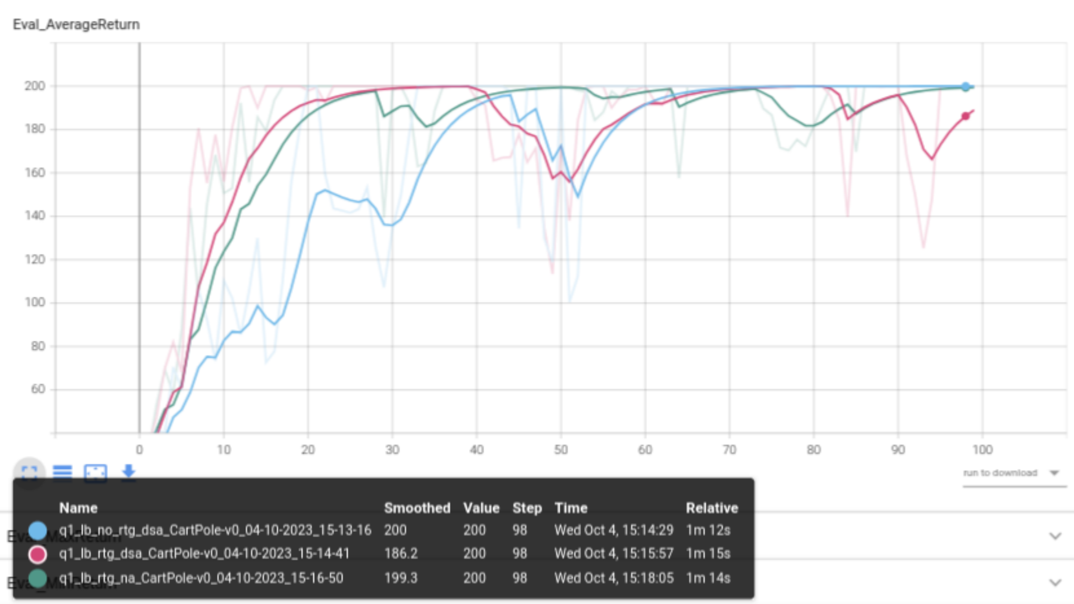
5.1.2.1 Small batch – [5 points]

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

It can be seen from both the small and large batch experiments that the reward-to-go estimator has better performance without advantage standardization. The difference is seen more when the batch size is smaller.

5.1.3.2 Advantage standardization – [5 points]

Q5.1.3.2

Yes, advantage standardization helped the policy reach a high return much faster and remain more stable than the case without it.

5.1.3.3 Batch size – [5 points]

Q5.1.3.1

Using a larger batch size helps the policy reach a high return much faster. The policy then fluctuates about the converged value. The performance is much better.

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

5.2.1 Configurations – [5 points]

Q5.2.1

```
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 1e-2 -rtg \
    --exp_name q2_b1000_r1e-2

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 2e-2 -rtg \
    --exp_name q2_b1000_r2e-2

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 1e-2 -rtg \
    --exp_name q2_b2000_r1e-2

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 2e-2 -rtg \
    --exp_name q2_b2000_r2e-2

python rob831/scripts/run_hw2.py --env_name InvertedPendulum-v4 \
    --ep_len 1000 --discount 0.9 -n 100 -l 2 -s 64 -b 3000 -lr 1e-2 -rtg \
    --exp_name q2_b3000_r1e-2

python rob831/scripts/run_hw2.py --env_name InvertedPendulum-v4 \
    --ep_len 1000 --discount 0.9 -n 100 -l 2 -s 64 -b 3000 -lr 2e-2 -rtg \
    --exp_name q2_b3000_r2e-2

python rob831/scripts/run_hw2.py --env_name InvertedPendulum-v4 \
    --ep_len 1000 --discount 0.9 -n 100 -l 2 -s 64 -b 4000 -lr 1e-2 -rtg \
    --exp_name q2_b4000_r1e-2

python rob831/scripts/run_hw2.py --env_name InvertedPendulum-v4 \
    --ep_len 1000 --discount 0.9 -n 100 -l 2 -s 64 -b 4000 -lr 2e-2 -rtg \
    --exp_name q2_b4000_r2e-2
```

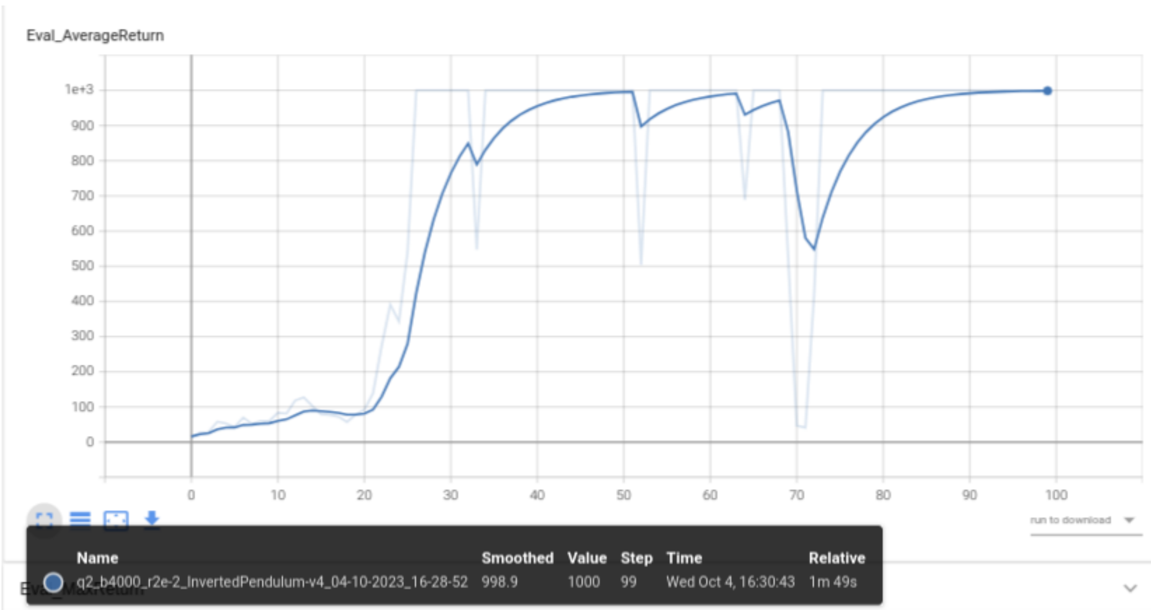
5.2.2 smallest b* and largest r* (same run) – [5 points]

Q5.2.2

Smallest **b*** = 4000
Largest **r*** = 2e-2

5.2.3 Plot – [5 points]

Q5.2.3



7 More Complex Experiments

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

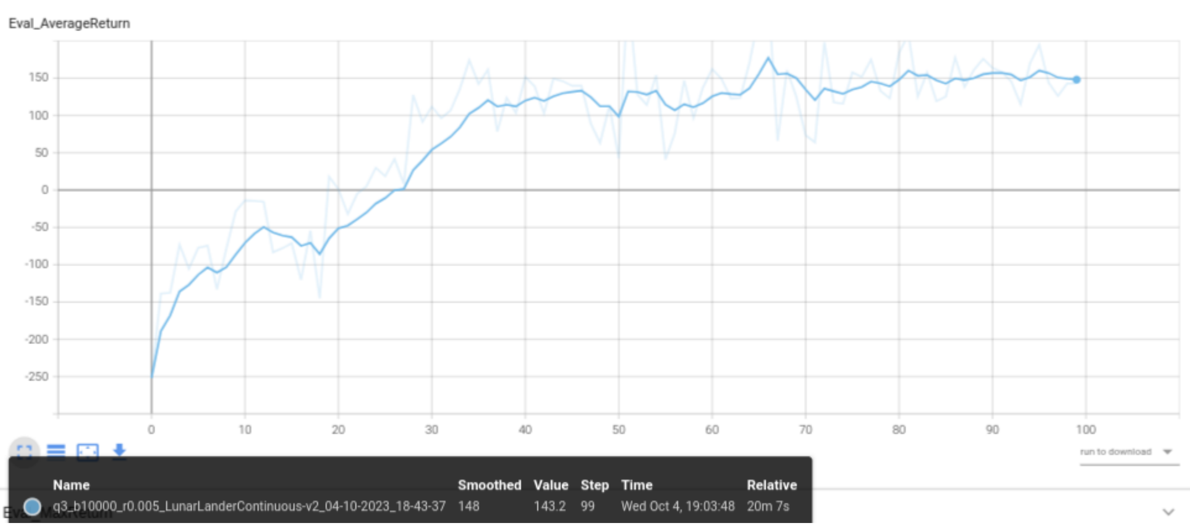
7.1.1 Configurations

Q7.1.1

```
python rob831/scripts/run_hw2.py \
    --env_name LunarLanderContinuous-v4 --ep_len 1000
    --discount 0.99 -n 100 -l 2 -s 64 -b 10000 -lr 0.005 \
    --reward_to_go --nn_baseline --exp_name q3_b10000_r0.005
```

7.1.2 Plot – [10 points]

Q7.1.2



7.2 Experiment 4 (HalfCheetah) – [30 points]

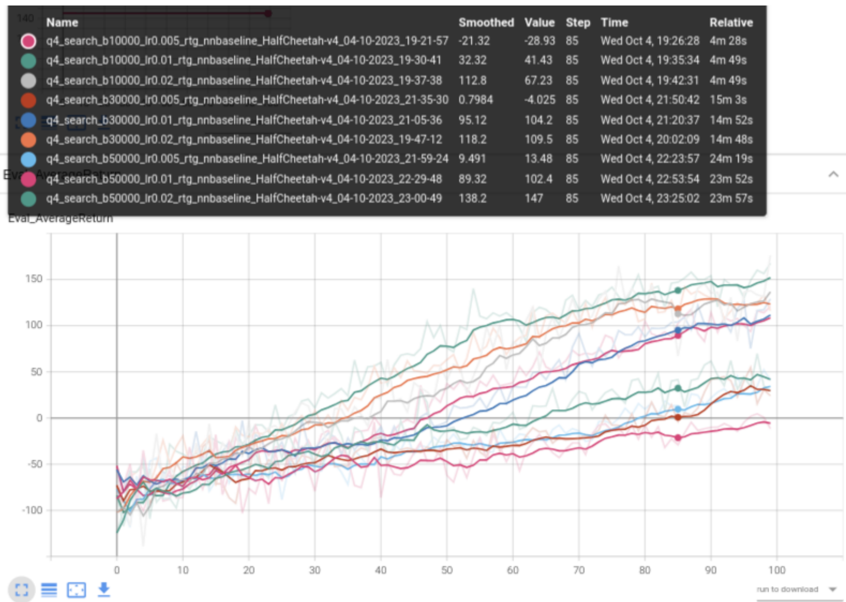
7.2.1 Configurations

Q7.2.1

```
# b ∈ [10000, 30000, 50000], r ∈ [0.005, 0.01, 0.02]
python rob831/scripts/run_hw2.py --env_name HalfCheetah-v4 --ep_len 150 \
    --discount 0.95 -n 100 -l 2 -s 32 -b <b> -lr <r> -rtg --nn_baseline \
    --exp_name q4_search_b<b>_lr<r>_rtg_nnbaseline
```

7.2.2 Plot – [10 points]

Q7.2.2



7.2.3 Optimal b* and r* – [3 points]

Q7.2.3

Optimal b* = 50000, r* = 0.02

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

Q7.2.4

It can be observed that fixing a batch size and increasing learning rate improves the performance. Similarly, increasing the batch size with a constant learning rate also improves the performance.

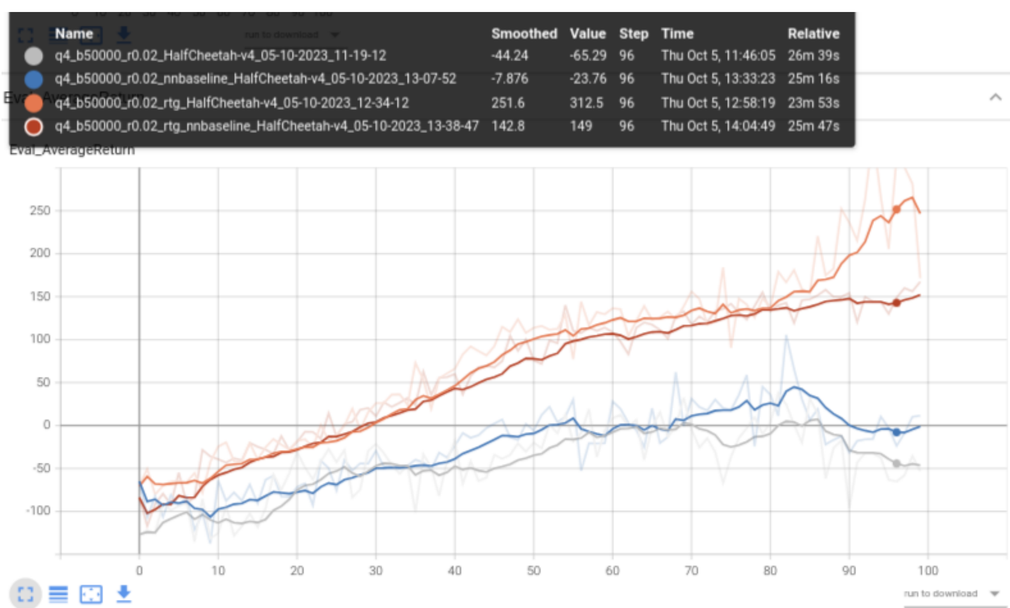
7.2.5 Configurations with optimal b* and r* – [3 points]

Q7.2.5

```
python rob831/scripts/run_hw2.py --env_name HalfCheetah-v4 --ep_len 150 \
    --discount 0.95 -n 100 -l 2 -s 32 -b <b*> -lr <r*> \
    --exp_name q4_b<b*>_r<r*>

python rob831/scripts/run_hw2.py --env_name HalfCheetah-v4 --ep_len 150 \
    --discount 0.95 -n 100 -l 2 -s 32 -b <b*> -lr <r*> -rtg \
    --exp_name q4_b<b*>_r<r*>_rtg

python rob831/scripts/run_hw2.py --env_name HalfCheetah-v4 --ep_len 150 \
    --discount 0.95 -n 100 -l 2 -s 32 -b <b*> -lr <r*> --nn_baseline \
    --exp_name q4_b<b*>_r<r*>_nnbaseline

python rob831/scripts/run_hw2.py --env_name HalfCheetah-v4 --ep_len 150 \
    --discount 0.95 -n 100 -l 2 -s 32 -b <b*> -lr <r*> -rtg --nn_baseline \
    --exp_name q4_b<b*>_r<r*>_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]

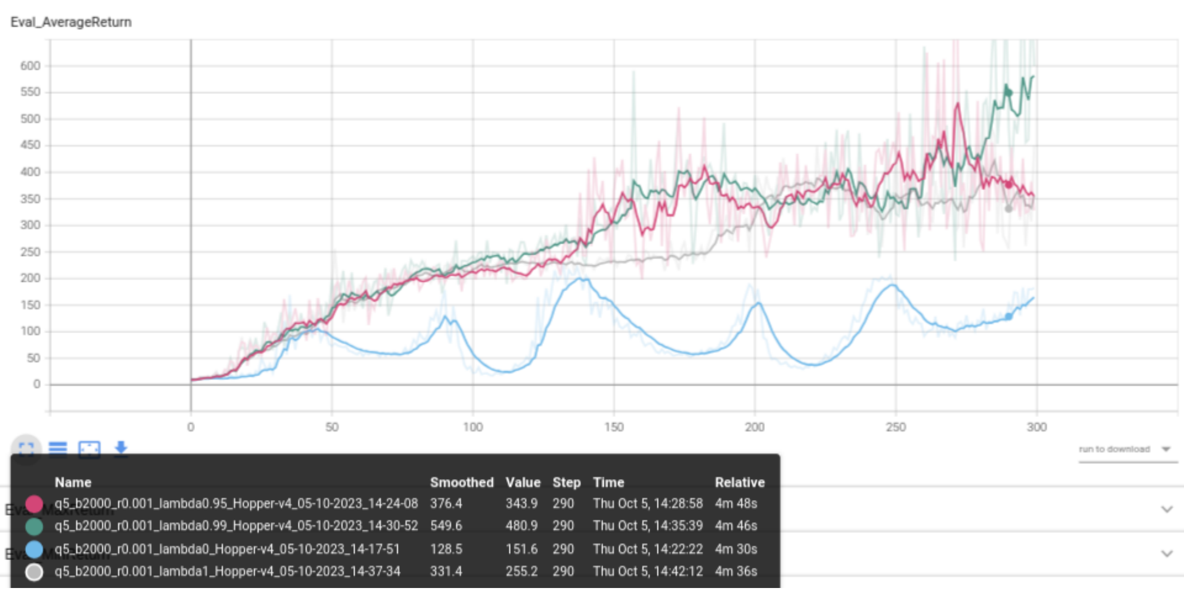
8.1.1 Configurations

Q8.1.1

```
# λ ∈ [0, 0.95, 0.99, 1]
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 <λ> \
    --exp_name q5_b2000_r0.001_lambda<λ>
```

8.1.2 Plot – [13 points]

Q8.1.2



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

Q8.1.3

As taught in class, λ serves as a control for the bias-variance tradeoff, where increasing λ decreases bias and increases variance. It can be seen that $\lambda = 0$ does not learn well. Setting λ to 0.95 and 0.99 gives good results with 0.99 being the best in practice.

9 Bonus! (optional)

9.1 Parallelization – [15 points]

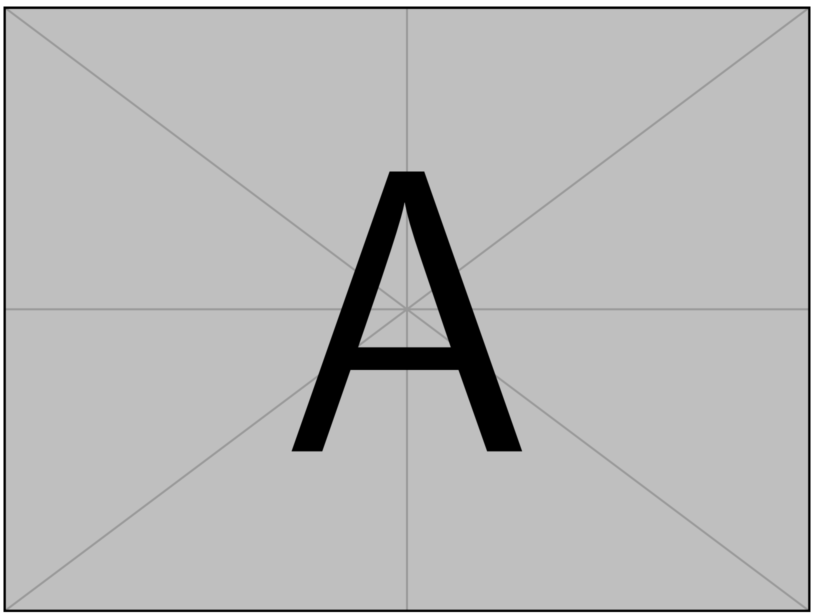
Q9.1

Difference in training time:

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

9.2 Multiple gradient steps – [5 points]

Q9.1




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