# Assignment 2: Policy Gradient

**Andrew ID:** dsreeniv
**Collaborators:** hravisan, vmohta
**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

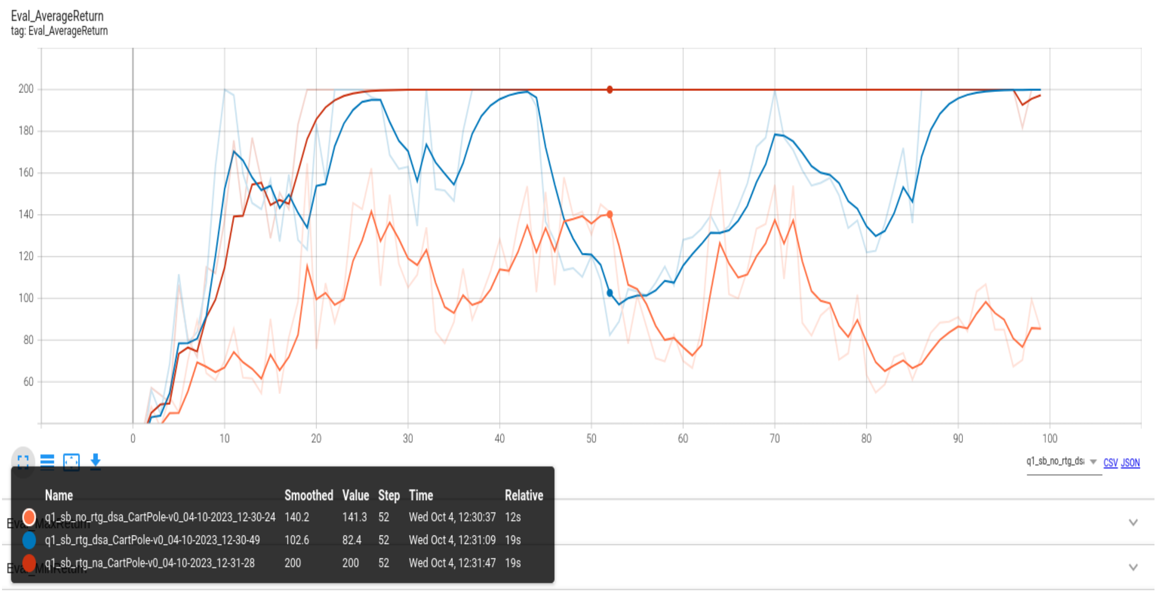
Q5.1.1

```
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]

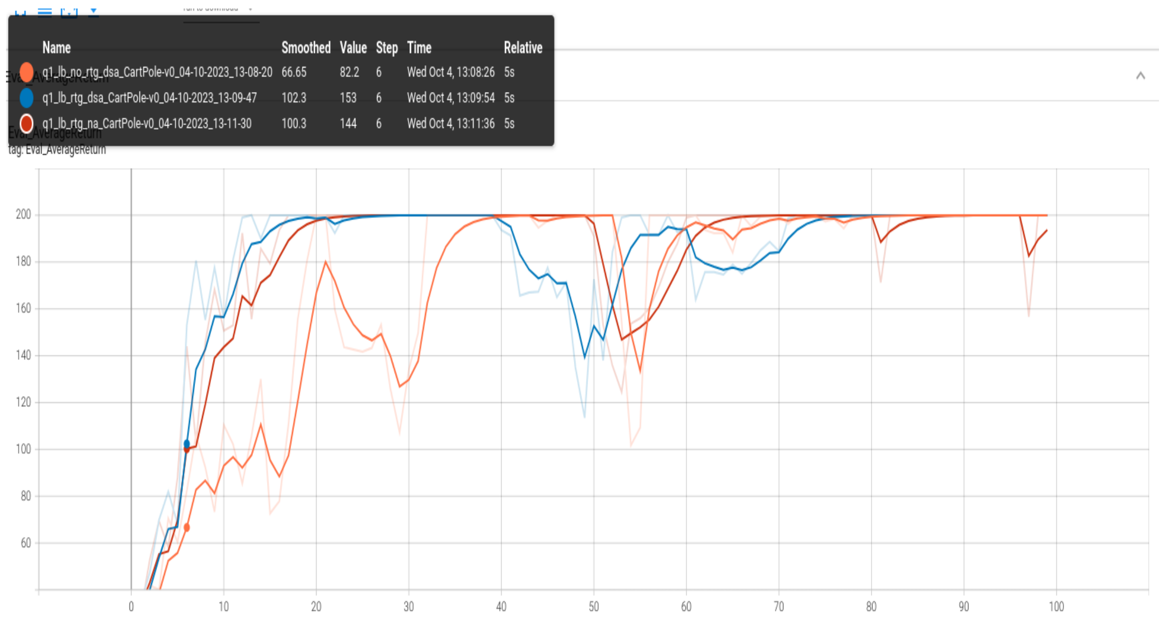
Q5.1.2.1



| | Name | Smoothed | Value | Step | Time | Relative |
|---|---|---|---|---|---|---|
| ● | q1_sb_no_rtg_dsa_CartPole-v0_04-10-2023_12-30-24 | 140.2 | 141.3 | 52 | Wed Oct 4, 12:30:37 | 12s |
| ● | q1_sb_rtg_dsa_CartPole-v0_04-10-2023_12-30-49 | 102.6 | 82.4 | 52 | Wed Oct 4, 12:31:09 | 19s |
| ● | q1_sb_rtg_na_CartPole-v0_04-10-2023_12-31-28 | 200 | 200 | 52 | Wed Oct 4, 12:31:47 | 19s |

#### 5.1.2.2 Large batch – [5 points]

Q5.1.2.2



| | Name | Smoothed | Value | Step | Time | Relative |
|---|---|---|---|---|---|---|
| ● | q1_lb_no_rtg_dsa_CartPole-v0_04-10-2023_13-08-20 | 66.65 | 82.2 | 6 | Wed Oct 4, 13:08:26 | 5s |
| ● | q1_lb_rtg_dsa_CartPole-v0_04-10-2023_13-09-47 | 102.3 | 153 | 6 | Wed Oct 4, 13:09:54 | 5s |
| ● | q1_lb_rtg_na_CartPole-v0_04-10-2023_13-11-30 | 100.3 | 144 | 6 | Wed Oct 4, 13:11:36 | 5s |

### 5.1.3 Analysis

#### 5.1.3.1 Value estimator – [5 points]

Q5.1.3.1

In both cases, the policy with reward-to-go significantly outperforms the trajctory-centric ones without advantage normalization. The variance is reduced and hence the policy converges faster and stays at maximum reward without deviating

#### 5.1.3.2 Advantage standardization – [5 points]

Q5.1.3.2

For small batch sizes, normalizing advantages greatly improves the stability of the policy signifcantly, whereas for large batch sizes the difference is not as evident. Since for higher batch sizes, the swing in advantages will be lower the apparent benefit of normalization is expected to be lower

#### 5.1.3.3 Batch size – [5 points]

Q5.1.3.1

Larger batch sizes improve the convergence speed and the stability of the trained networks across all cases. Even in the worst case a large batch size helps get the policy to reach a high reward unlike in the small batch size case

## 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 40000 -lr 0.005 -rtg \
--exp_name q2_b50000_r0.01

python rob831/scripts/run_hw2.py --env_name InvertedPendulum-v4 \
--ep_len 1000 --discount 0.9 -n 100 -l 2 -s 64 -b 30000 -lr 0.05 -rtg \
--exp_name q2_b30000_r0.05

python rob831/scripts/run_hw2.py --env_name InvertedPendulum-v4 \
--ep_len 1000 --discount 0.9 -n 100 -l 2 -s 64 -b 20000 -lr 0.01 -rtg \
--exp_name q2_b20000_r0.01

python rob831/scripts/run_hw2.py --env_name InvertedPendulum-v4 \
--ep_len 1000 --discount 0.9 -n 100 -l 2 -s 64 -b 20000 -lr 0.05 -rtg \
--exp_name q2_b20000_r0.05

python rob831/scripts/run_hw2.py --env_name InvertedPendulum-v4 \
--ep_len 1000 --discount 0.9 -n 100 -l 2 -s 64 -b 20000 -lr 0.001 -rtg \
--exp_name q2_b20000_r0.001

python rob831/scripts/run_hw2.py --env_name InvertedPendulum-v4 \
--ep_len 1000 --discount 0.9 -n 100 -l 2 -s 64 -b 10000 -lr 0.001 -rtg \
--exp_name q2_b20000_r0.001
```

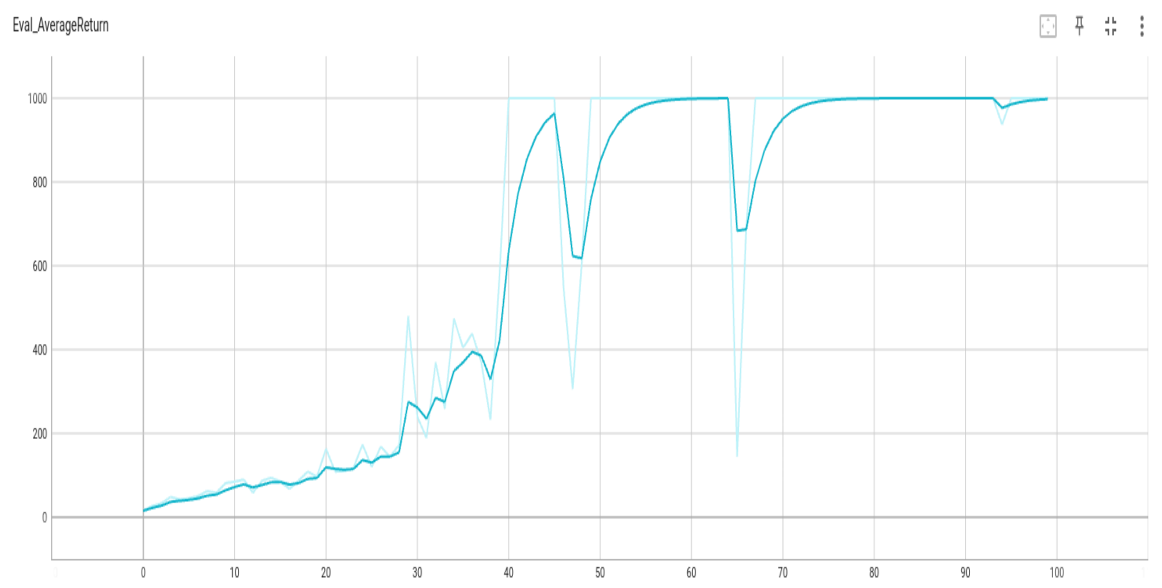
### 5.2.2 smallest b* and largest r* (same run) – [5 points]

Q5.2.2

The best results came from b* = 30000, r* = 0.01

#### 5.2.3 Plot – [5 points]

Q5.2.3



## 7 More Complex Experiments

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

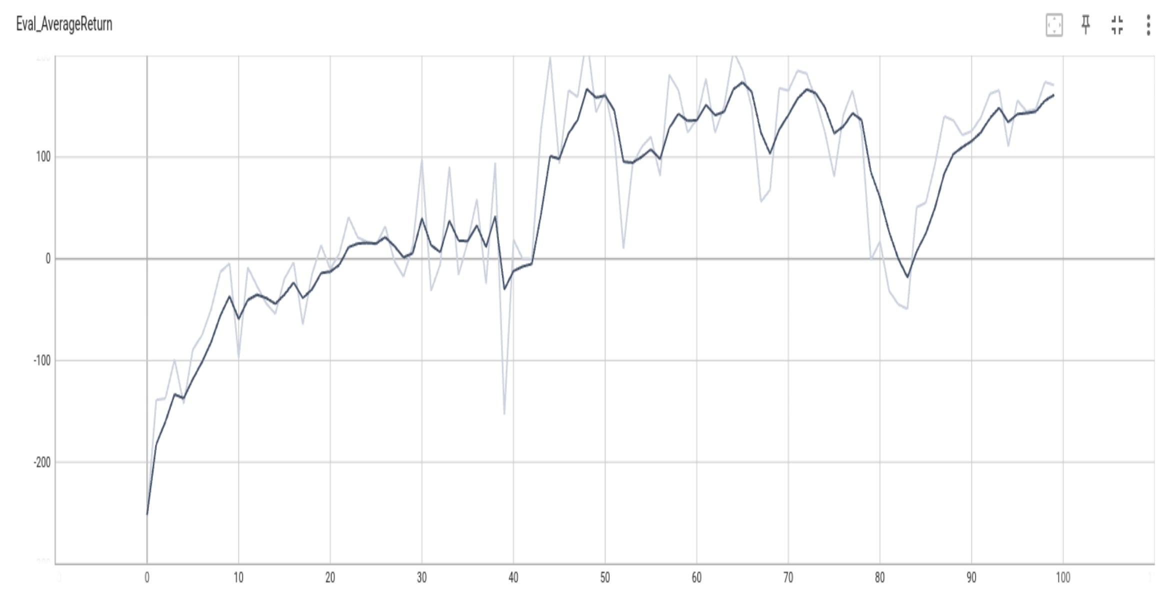
#### 7.1.1 Configurations

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]

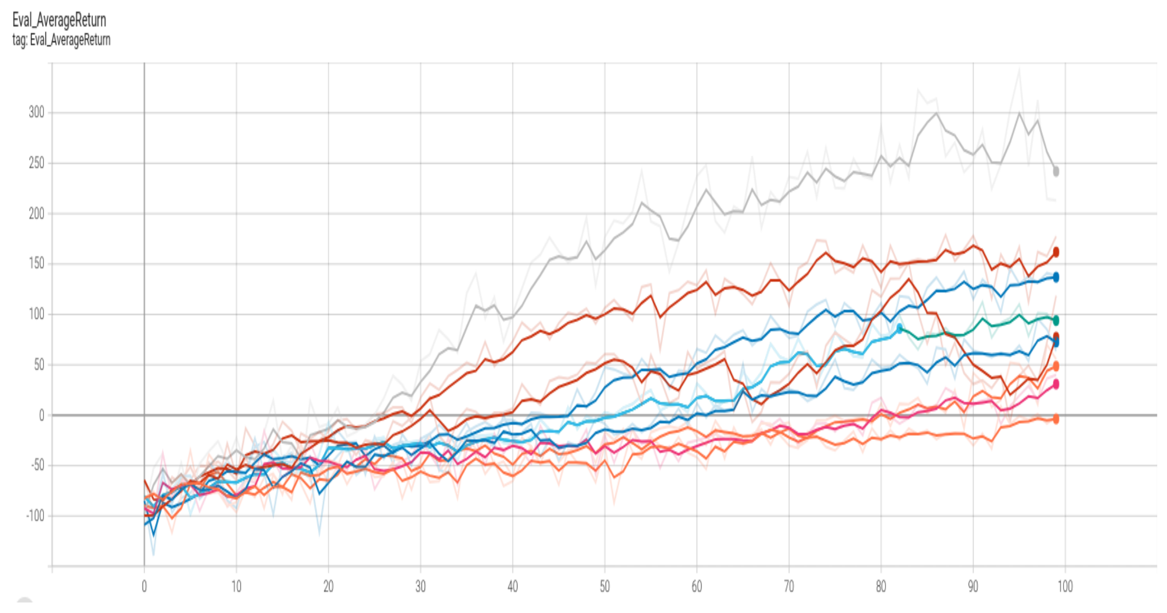
### 7.2.1 Configurations

Q7.2.1

```
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_lr0.005_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_lr0.01_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_lr0.02_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_lr0.005_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_lr0.01_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_lr0.02_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_lr0.005_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_lr0.01_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_lr0.02_rtg_nnbaseline
```

### 7.2.2 Plot – [10 points]

Q7.2.2



### 7.2.3 Optimal b* and r* – [3 points]

Q7.2.3

Optimal values are b* = 30000, r* = 0.02

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

Q7.2.4

On increasing the batch size, the variance of the return is lower and the policy is generally more stable over iterations, but after a point the high amount of data means there is no noise in the dataset which means that the overall speed of convergence is lower
On increaasing learning rate, the performance improves rapidly. After a point, increasing it impacts stability since the network will keep missing minimas and overstep good regions, leading to rapid drop-offs and instability over iterations.

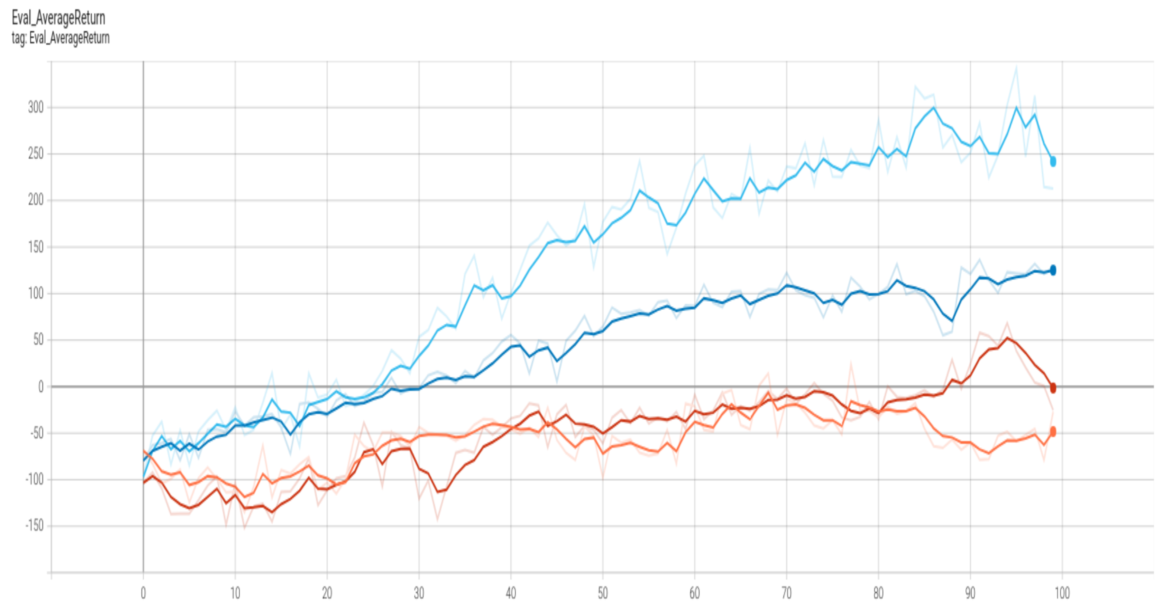
### 7.2.5 Configurations with optimal b* and r* – [3 points]

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]

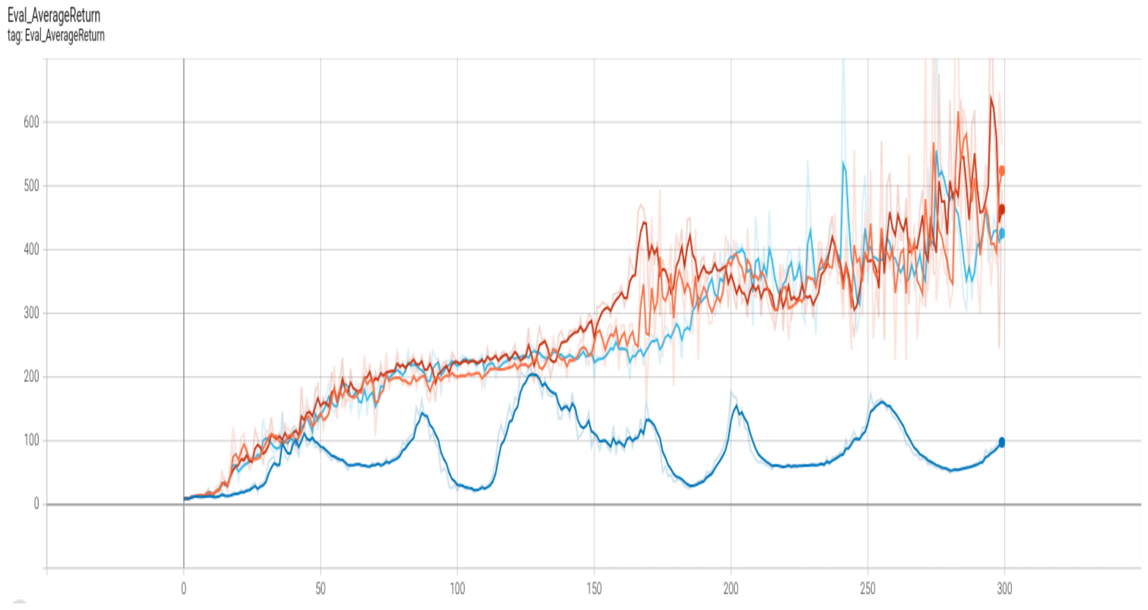
### 8.1.1 Configurations

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 $\lambda$ affects task performance – [7 points]

Q8.1.3

For $\lambda = 0$, we see that the network doesn't learn since we are not using the baseline network at all For $\lambda = 0.95, 0.99$, we observe competitive performance with 0.95 being more stable but 0.99 converging to a higher value faster For $\lambda = 1$ we see it being the most stable but converging slower than any both of the above

## 9 Bonus! (optional)

### 9.1 Parallelization – [15 points]

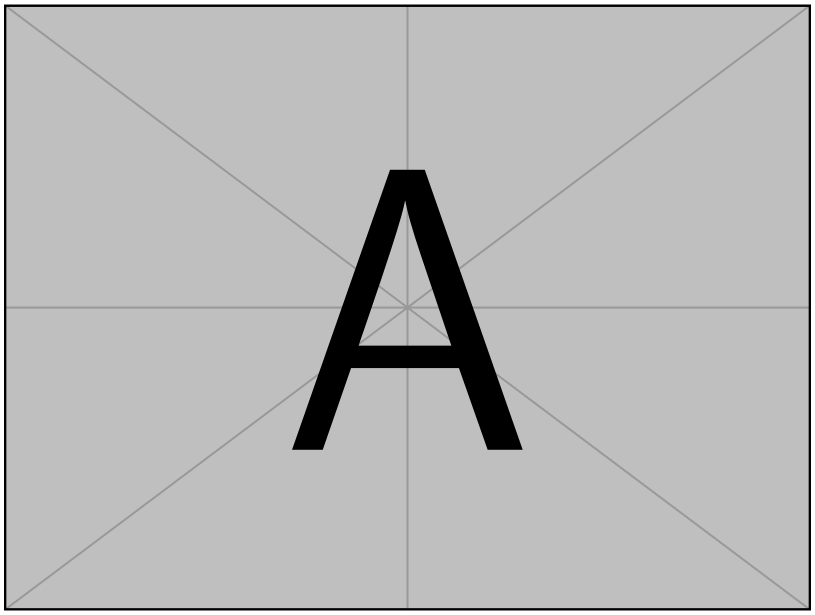
Q9.1

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 \
```