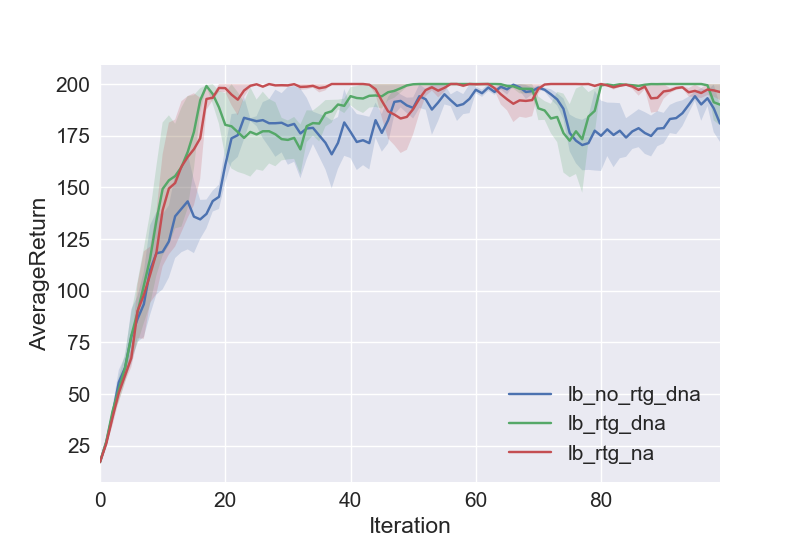
CS294-112 Homework 2: Policy Gradients

**Problem 4:**

**Figure 1:** Average returns vs. Number of iterations for SB



**Figure 2:** Average returns vs. Number of iterations for LB



**Short Answers:**

Which gradient estimator has better performance without advantage centering – the trajectory centric one, or the one using reward to go?

The one using reward to go has better performance without advantage centering.

Did advantage centering help?

Yes

Did the batch size make an impact?

Yes. A larger batch size allowed for a faster convergence rate.

**Command Line Expressions:**

python train\_pg\_f18.py CartPole-v0 -n 100 -b 1000 -e 3 -dna --exp\_name sb\_no\_rtg\_dna

python train\_pg\_f18.py CartPole-v0 -n 100 -b 1000 -e 3 -rtg -dna --exp\_name sb\_rtg\_dna

python train\_pg\_f18.py CartPole-v0 -n 100 -b 1000 -e 3 -rtg --exp\_name sb\_rtg\_na

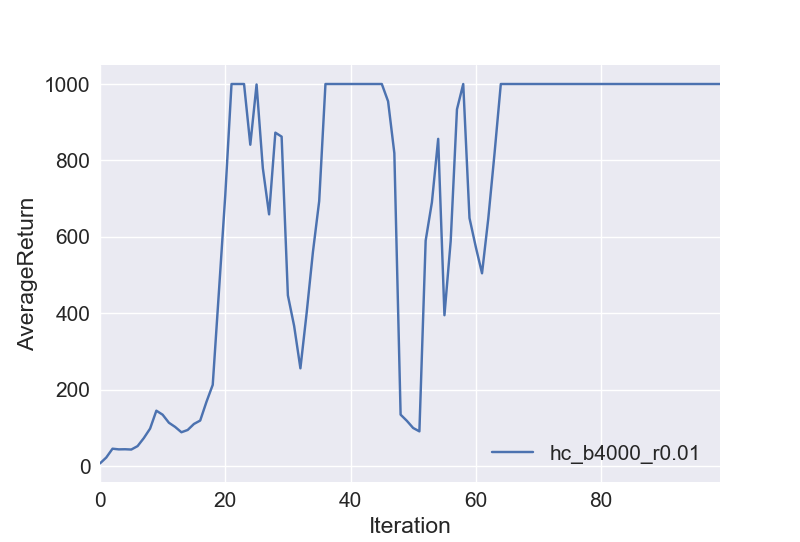
python train\_pg\_f18.py CartPole-v0 -n 100 -b 5000 -e 3 -dna --exp\_name lb\_no\_rtg\_dna

python train\_pg\_f18.py CartPole-v0 -n 100 -b 5000 -e 3 -rtg -dna --exp\_name lb\_rtg\_dna

python train\_pg\_f18.py CartPole-v0 -n 100 -b 5000 -e 3 -rtg --exp\_name lb\_rtg\_na

**Problem 5:**

**Figure 3:** Average returns vs. Number of iterations for InvertedPendulum. I used a batch size of 4000 and a learning rate of 0.01.

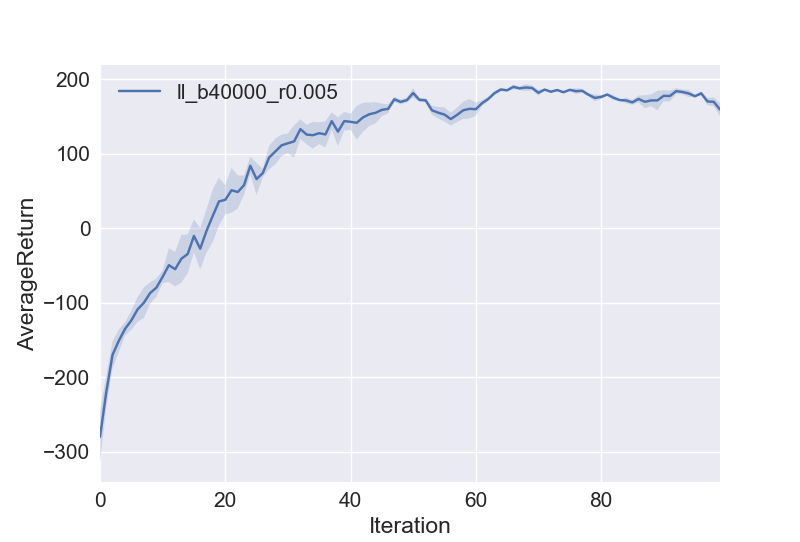


**Command Line Expressions:**

python train\_pg\_f18.py InvertedPendulum-v2 -ep 1000 --discount 0.9 -n 100 -e 3 -l 2 -s 64 -b 4000 -lr 0.01 -rtg --exp \_name hc\_b4000\_r0.01

**Problem 7:**

**Figure 4:** Average returns vs. Number of iterations for LunarLander

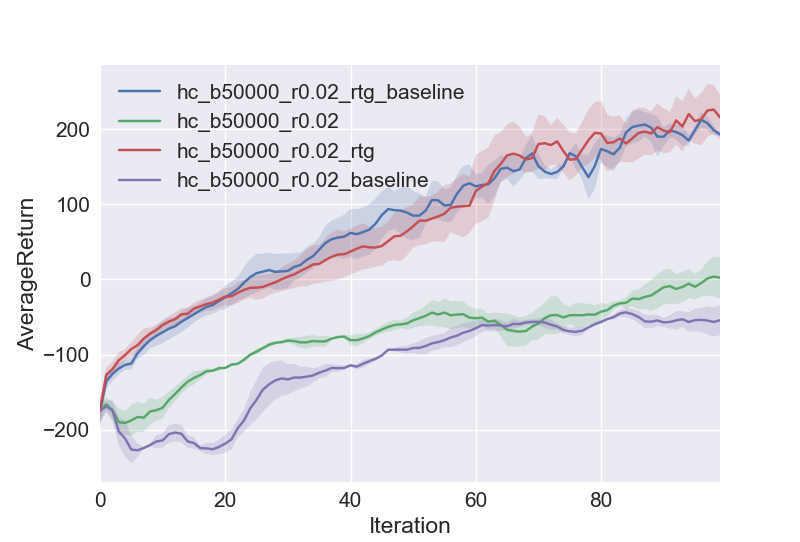


**Command Line Expressions:**

python train\_pg\_f18.py LunarLanderContinuous-v2 -ep 1000 --discount 0.99 -n 100 -e 3 -l 2 -s 64 -b 40000 -lr 0.005 -rtg --nn\_baseline --exp\_name ll\_b40000\_r0.005

**Problem 8:**

**Figure 5:** Average returns vs. Number of Iterations for HalfCheetah



**Short Answers:**

How did the batch size and learning rate affect the performance?

In general, an increased batch size and lower learning rate correlated with a better performance. A batch size of 50000 and learning rate of 0.02 seemed to give optimal performance.

**Command Line Expressions:**

python train\_pg\_f18.py HalfCheetah-v2 -ep 150 --discount 0.9 -n 100 -e 3 -l 2 -s 32 -b 50000 -lr 0.02 --exp\_name hc\_b50000\_r0.02

python train\_pg\_f18.py HalfCheetah-v2 -ep 150 --discount 0.9 -n 100 -e 3 -l 2 -s 32 -b 50000 -lr 0.02 -rtg --exp\_name hc\_b50000\_r0.02\_rtg

python train\_pg\_f18.py HalfCheetah-v2 -ep 150 --discount 0.9 -n 100 -e 3 -l 2 -s 32 -b 50000 -lr 0.02 --nn\_baseline --exp\_name hc\_b50000\_r0.02\_baseline

python train\_pg\_f18.py HalfCheetah-v2 -ep 150 --discount 0.9 -n 100 -e 3 -l 2 -s 32 -b 50000 -lr 0.02 -rtg --nn\_baseline --exp\_name hc\_b50000\_r0.02\_rtg\_baseline