[SOLUTION TEMPLATE] Assignment 2: Policy Gradients

Due September 25, 11:59 pm

4 Policy Gradients

- Create two graphs:
  - In the first graph, compare the learning curves (average return vs. number of environment steps) for the experiments prefixed with `cartpole`. (The small batch experiments.)
  - In the second graph, compare the learning curves for the experiments prefixed with `cartpole_lb`. (The large batch experiments.)

  For all plots in this assignment, the x-axis should be number of environment steps, logged as `Train_EnvstepsSoFar` (*not* number of policy gradient iterations).
- Answer the following questions briefly:
  - Which value estimator has better performance without advantage normalization: the trajectory-centric one, or the one using reward-to-go?
  - Did advantage normalization help?
  - Did the batch size make an impact?
- Provide the exact command line configurations (or `#@params` settings in Colab) you used to run your experiments, including any parameters changed from their defaults.

Solutions 运行如下代码：

```
export PYTHONPATH="/media/zks/T7 Shield/2025上学期/深度强化学习/LAB/homework_fall2023:$PYTHONPATH"
python hw2/cs285/scripts/run_hw2.py --env_name CartPole-v0 -n 100 -b 1000 \
--exp_name cartpole
python hw2/cs285/scripts/run_hw2.py --env_name CartPole-v0 -n 100 -b 1000 \
-rtg --exp_name cartpole_rtg
python hw2/cs285/scripts/run_hw2.py --env_name CartPole-v0 -n 100 -b 1000 \
-na --exp_name cartpole_na
python hw2/cs285/scripts/run_hw2.py --env_name CartPole-v0 -n 100 -b 1000 \
-rtg -na --exp_name cartpole_rtg_na
python hw2/cs285/scripts/run_hw2.py --env_name CartPole-v0 -n 100 -b 4000 \
--exp_name cartpole_lb
python hw2/cs285/scripts/run_hw2.py --env_name CartPole-v0 -n 100 -b 4000 \
-rtg --exp_name cartpole_lb_rtg
python hw2/cs285/scripts/run_hw2.py --env_name CartPole-v0 -n 100 -b 4000 \
-na --exp_name cartpole_lb_na
python hw2/cs285/scripts/run_hw2.py --env_name CartPole-v0 -n 100 -b 4000 \
-rtg -na --exp_name cartpole_lb_rtg_na
```

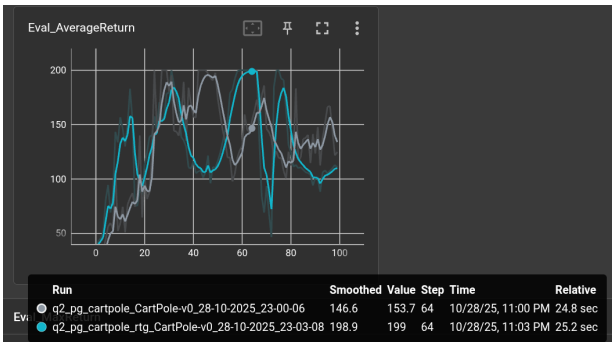
对比 reward-to-go 和 trajectory-centric，reward-to-go 在小批量样本上训练早期明显收敛更快，在大批量样本上差异不明显，因为大批量样本下方差已经足够小。

对比 advantage normalization，advantage normalization 在小批量样本上训练效果明显更好，在样本量少（例如 1000）情况下，优势的均值和方差波动大；如果不做归一化，policy gradient 的步长大小会忽上忽下；因此 -na 能显著提高稳定性、收敛更平滑在大批量样本上差异不明显，因为大批量样本优势的统计量更接近真实分布。

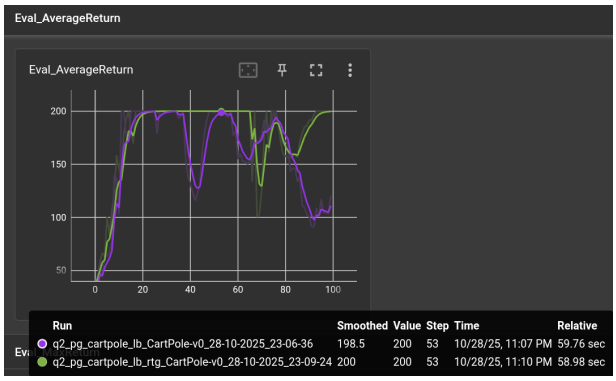
RTG vs Trajectory-based：小批量下 RTG 往往更快更稳；大批量时差距变小（大 batch 本身就降方差）。

Advantage 归一化：小批量帮助更明显；大批量时提升不明显或几乎重合。

批量大小：更大批量带来更平滑/稳定的学习曲线，通常更容易到 200，但每次更新更“慢”。

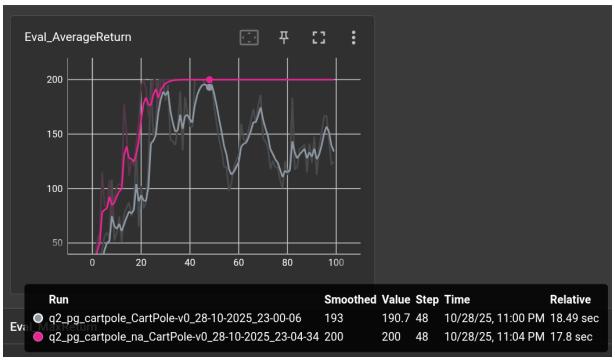


(a) Small Batch ($b = 1000$)



(b) Large Batch ($b = 4000$)

Figure 1: Comparison of Reward-to-Go (RTG) vs Trajectory-based PG under different batch sizes. RTG improves performance more noticeably in small-batch settings.



(a) Small Batch ($b = 1000$)

(b) Large Batch ($b = 4000$)

Figure 2: Comparison of Advantage Normalization PG under different batch sizes. Advantage Normalization improves performance more noticeably in small-batch settings.

5 Neural Network Baseline

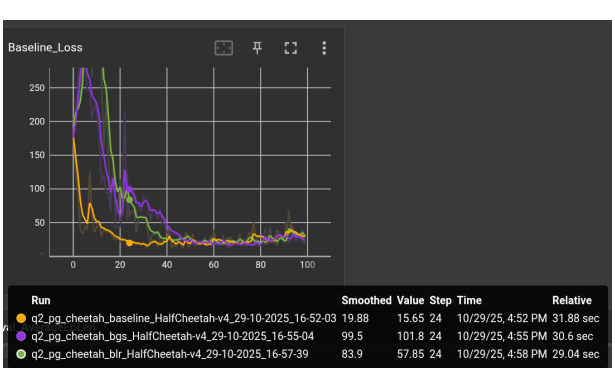
- Plot a learning curve for the baseline loss.
- Plot a learning curve for the eval return. You should expect to achieve an average return over 300 for the baselined version.
- Run another experiment with a decreased number of baseline gradient steps (`-bgs`) and/or baseline learning rate (`-blr`). How does this affect (a) the baseline learning curve and (b) the performance of the policy?
- **Optional:** Add `-na` back to see how much it improves things. Also, set `video_log_freq 10`, then open TensorBoard and go to the "Images" tab to see some videos of your HalfCheetah walking along!

Solution

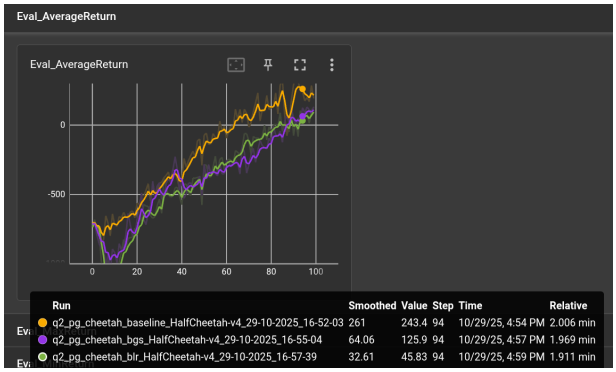
```
export PYTHONPATH="/media/zks/T7 Shield/2025上学期/深度强化学习/LAB/homework_fall2023:$PYTHONPATH"
# No baseline
python hw2/cs285/scripts/run_hw2.py --env_name HalfCheetah-v4 \
-n 100 -b 5000 -rtg --discount 0.95 -lr 0.01 \
--exp_name cheetah
# Baseline
```

```
python hw2/cs285/scripts/run_hw2.py --env_name HalfCheetah-v4 \
-n 100 -b 5000 -rtg --discount 0.95 -lr 0.01 \
--use_baseline -blr 0.01 -bgs 5 --exp_name cheetah_baseline
# Bgs
python hw2/cs285/scripts/run_hw2.py --env_name HalfCheetah-v4 \
-n 100 -b 5000 -rtg --discount 0.95 -lr 0.01 \
--use_baseline -blr 0.01 -bgs 1 --exp_name cheetah_bgs
# Blr
python hw2/cs285/scripts/run_hw2.py --env_name HalfCheetah-v4 \
-n 100 -b 5000 -rtg --discount 0.95 -lr 0.01 \
--use_baseline -blr 0.001 -bgs 5 --exp_name cheetah_blr
# NA
python hw2/cs285/scripts/run_hw2.py --env_name HalfCheetah-v4 \
-n 100 -b 5000 -rtg --discount 0.95 -lr 0.01 \
--use_baseline -blr 0.01 -bgs 5 -na --exp_name cheetah_na
# NA Video
python hw2/cs285/scripts/run_hw2.py --env_name HalfCheetah-v4 \
-n 100 -b 5000 -rtg --discount 0.95 -lr 0.01 --video_log_freq 10 \
--use_baseline -blr 0.01 -bgs 5 -na --exp_name cheetah_na
```

下图展示了 baseline loss 曲线和 eval return 曲线。可以看到，降低 baseline 的梯度更新步数或学习率都会导致 baseline 收敛变慢，进而影响 policy 的性能。



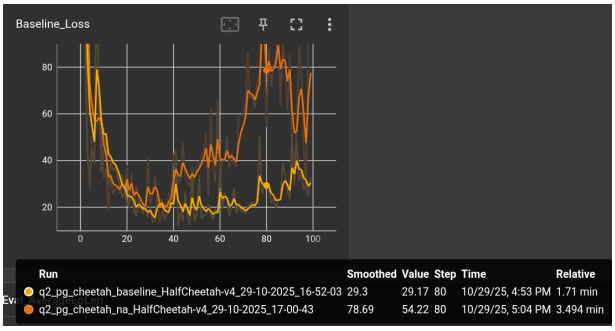
(a) Baseline Loss Curve



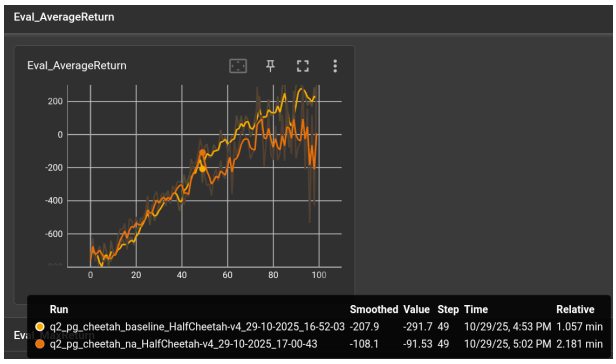
(b) Eval Return Curve

Figure 3: Learning Curves for baseline_gradient_steps and baseline_learning_rate on HalfCheetah-v4. Reducing baseline gradient steps or learning rate slows baseline convergence and degrades policy performance.

下图展示了 baseline loss 曲线和 eval return 曲线。可以看到，使用 baseline 方法和使用 advantage normalization 方法都能显著提升 policy 的性能，并且两者相差不大。



(a) Baseline Loss Curve



(b) Eval Return Curve

Figure 4: Learning Curves for baseline method and advantage normalization method on HalfCheetah-v4.

6 Generalized Advantage Estimation

- Provide a single plot with the learning curves for the `LunarLander-v2` experiments that you tried. Describe in words how λ affected task performance. The run with the best performance should achieve an average score close to 200 (180+).
- Consider the parameter λ. What does $\lambda = 0$ correspond to? What about $\lambda = 1$? Relate this to the task performance in `LunarLander-v2` in one or two sentences.

```
python hw2/cs285/scripts/run_hw2.py \
  --env_name LunarLander-v2 --ep_len 1000 \
  --discount 0.99 -n 300 -l 3 -s 128 -b 2000 -lr 0.001 \
  --use_reward_to_go --use_baseline --gae_lambda 0.00 \
  --exp_name lunar_lander_lambda0

python hw2/cs285/scripts/run_hw2.py \
  --env_name LunarLander-v2 --ep_len 1000 \
  --discount 0.99 -n 300 -l 3 -s 128 -b 2000 -lr 0.001 \
  --use_reward_to_go --use_baseline --gae_lambda 0.95 \
  --exp_name lunar_lander_lambda0.95

python hw2/cs285/scripts/run_hw2.py \
  --env_name LunarLander-v2 --ep_len 1000 \
  --discount 0.99 -n 300 -l 3 -s 128 -b 2000 -lr 0.001 \
  --use_reward_to_go --use_baseline --gae_lambda 0.98 \
  --exp_name lunar_lander_lambda0.98

python hw2/cs285/scripts/run_hw2.py \
  --env_name LunarLander-v2 --ep_len 1000 \
  --discount 0.99 -n 300 -l 3 -s 128 -b 2000 -lr 0.001 \
  --use_reward_to_go --use_baseline --gae_lambda 0.99 \
  --exp_name lunar_lander_lambda0.99

python hw2/cs285/scripts/run_hw2.py \
  --env_name LunarLander-v2 --ep_len 1000 \
  --discount 0.99 -n 300 -l 3 -s 128 -b 2000 -lr 0.001 \
  --use_reward_to_go --use_baseline --gae_lambda 1 \
  --exp_name lunar_lander_lambda1
```

下图展示了 everage return 曲线和 std 曲线，可以看到不同的 λ 值对任务性能有显著影响。λ =0 (TD(0))：

学习曲线：很快但波动大，训练过程中可能会有较强的震荡。因为 $\lambda = 0$ 使得估计的优势高度依赖于当前的 TD 目标（即，单步回报），导致方差较低但偏差较高。

平均回报：初期有较快的收敛，但最终的稳定性较差，可能会停留在 较低的分数，接近 100 150。

优势：快速收敛，但有较大的偏差。

$\lambda = 1$ (MC)：

学习曲线：学习更慢，波动较大，因为 $\lambda = 1$ 使用完整的蒙特卡洛估计（整条轨迹的回报），这种方法减少了偏差，但增加了方差。

平均回报：在训练的初期和中期，可能会看到较大的波动，最终可能接近 150 200，但总体比较不稳定，适应较慢。

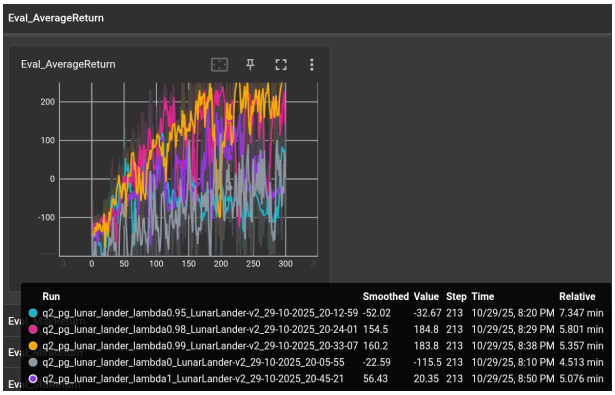
优势：降低偏差，提高长期表现，但需要更长时间才能稳定收敛。

$\lambda = 0.95, 0.98, 0.99$：

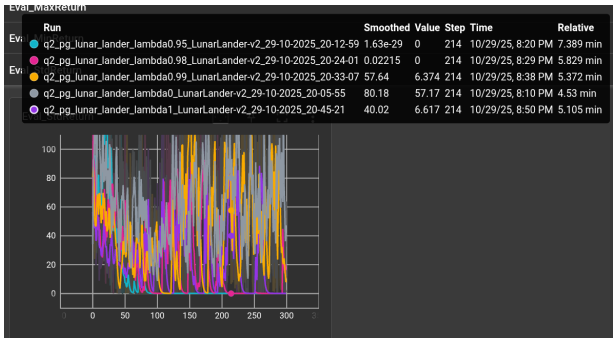
学习曲线：在 0.95-0.99 范围内，期望看到更平滑且稳定的曲线。这几个 λ 值会产生一个折衷：偏差与方差的平衡。

平均回报：曲线会更平稳，最终表现较好，接近 180 200。

优势：适中、稳定，不太保守也不太冒险，往往在偏差和方差之间找到一个较好的平衡。



(a) Eval Return Curve



(b) Std Return Curve

Figure 5: Learning Curves for different lamda.

7 Hyperparameter Tuning

1. Provide a set of hyperparameters that achieve high return on `InvertedPendulum-v4` in as few environment steps as possible.
2. Show learning curves for the average returns with your hyperparameters and with the default settings, with environment steps on the x-axis. Returns should be averaged over 5 seeds.

Solutions

```
for seed in $(seq 1 5); do
    python hw2/cs285/scripts/run_hw2.py --env_name InvertedPendulum-v4 -n 100 \
    --exp_name pendulum_default_s$seed \
    -rtg --use_baseline -na \
    --batch_size 5000 \
    --seed $seed
done

python hw2/cs285/scripts/run_hw2.py --env_name InvertedPendulum-v4 -n 100 \
    --exp_name pendulum_try1 \
    -rtg --use_baseline -na --discount 0.99 \
    --batch_size 5000 \
    --gae_lambda 0.99

python hw2/cs285/scripts/run_hw2.py --env_name InvertedPendulum-v4 -n 100 \
    --exp_name pendulum_try2 \
    -rtg --use_baseline -na --discount 0.99 -lr 0.005\
    --batch_size 5000 \
    --gae_lambda 0.99

python hw2/cs285/scripts/run_hw2.py --env_name InvertedPendulum-v4 -n 100 \
    --exp_name pendulum_try3 \
    -rtg --use_baseline -na --discount 0.99 -lr 0.01\
    --batch_size 5000 \
    --gae_lambda 0.99
```

随机的 5 个种子下，默认超参数的平均学习曲线如下：

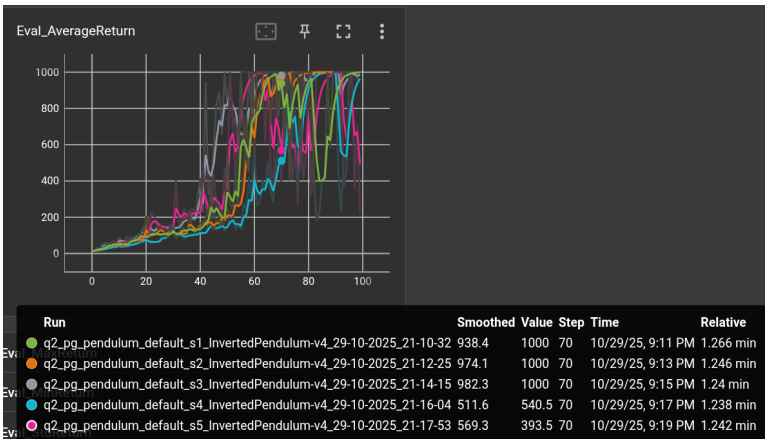


Figure 6: Learning Curves for default hyperparameters on InvertedPendulum-v4. Default hyperparameters achieve moderate returns but converge slowly.

try3 的超参数表现最好，学习曲线如下：

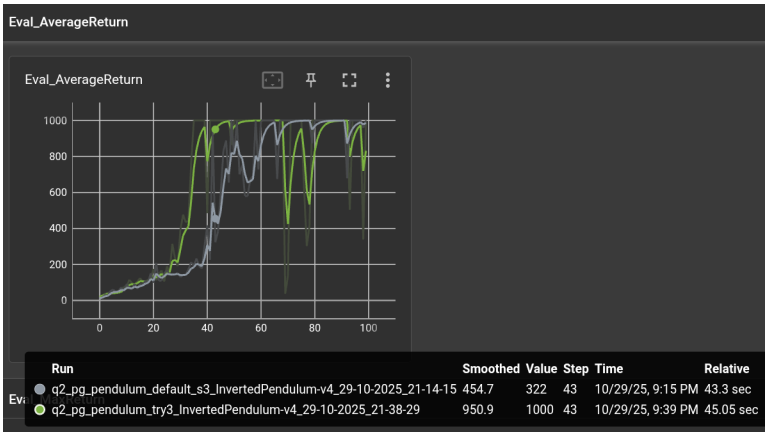


Figure 7: Learning Curves for default hyperparameters and tuned hyperparameters on InvertedPendulum-v4. Tuned hyperparameters achieve higher returns more quickly.

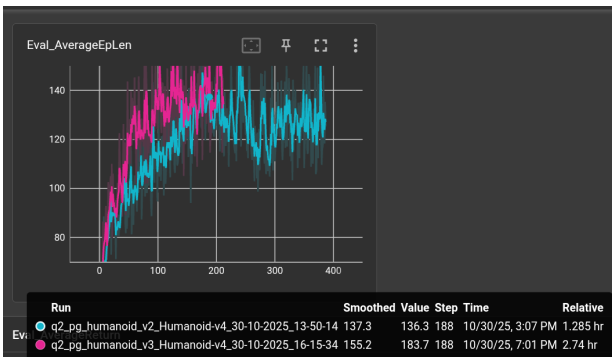
8 (Extra Credit) Humanoid

1. Plot a learning curve for the Humanoid-v4 environment. You should expect to achieve an average return of at least 600 by the end of training. Discuss what changes, if any, you made to complete this problem (for example: optimizations to the original code, hyperparameter changes, algorithmic changes).

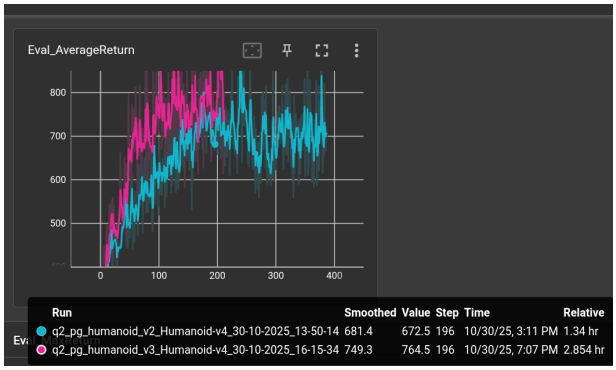
Solution

```
export PYTHONPATH="/media/zks/T7 Shield/2025FirstSemester/DeepRL/LAB/homework_fall2023:$PYTHONPATH"
python hw2/cs285/scripts/run_hw2_parallel.py \
    --env_name Humanoid-v4 --ep_len 1000 \
    --discount 0.99 -n 1000 -l 3 -s 256 -b 50000 -lr 0.001 \
    --baseline_gradient_steps 50 \
    -na --use_reward_to_go --use_baseline --gae_lambda 0.97 \
    --exp_name humanoid_v1

python hw2/cs285/scripts/run_hw2.py \
    --env_name Humanoid-v4 --ep_len 1000 \
    --discount 0.99 -n 1000 -l 3 -s 256 -b 50000 -lr 0.001 \
    --baseline_gradient_steps 50 \
    -na --use_reward_to_go --use_baseline --gae_lambda 0.97 \
    --exp_name humanoid_v2

python hw2/cs285/scripts/run_hw2_parallel.py \
    --env_name Humanoid-v4 --ep_len 1000 \
    --discount 0.99 -n 1000 -l 3 -s 256 -b 250000 -lr 0.001 \
    --baseline_gradient_steps 50 \
    -na --use_reward_to_go --use_baseline --gae_lambda 0.97 \
    --exp_name humanoid_v3
```

分析这两个图可以明显的看到，使用并行化训练的版本在相同的环境交互步数下，表现要优于非并行化版本。可以同时计算多个环境的样本，提升了样本效率，从而提升了训练效果。



(a) Eval Eplen Curve

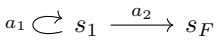


(b) Eval Return Curve

Figure 8: Learning Curves for parallel.

9 Analysis

Consider the following infinite-horizon MDP:



At each step, the agent stays in state s_1 and receives reward 1 if it takes action a_1, and receives reward 0 and terminates the episode otherwise. Parametrize the policy as stationary (not dependent on time) with a single parameter:

$$\pi_\theta(a_1|s_1) = \theta, \pi_\theta(a_2|s_1) = 1 - \theta$$

1. Applying policy gradients

   (a) Use policy gradients to compute the gradient of the expected return $R(\tau)$ with respect to the parameter θ. **Do not use discounting.**

   Hint: to compute $\sum_{k=1}^{\infty} k\alpha^{k-1}$, you can write:

$$\sum_{k=1}^{\infty} k\alpha^{k-1} = \sum_{k=1}^{\infty} \frac{d}{d\alpha}\alpha^k = \frac{d}{d\alpha}\sum_{k=1}^{\infty}\alpha^k$$

   Solution:

   先把这个小 MDP 说清楚：每一步在 s_1：

   * 选 a_1（概率 θ）：得到奖励 1，并留在 s_1；* 选 a_2（概率 $1-\theta$）：得到奖励 0，并终止。

   于是一次轨迹就是“连续做 a_1 共 k 次后做一次 a_2”：

   * 轨迹 τ_k 的概率：$p_\theta(\tau_k) = (1-\theta)\theta^k$（$k \geq 0$）* 回报：$R(\tau_k) = k$（做了 k 次 a_1，每次奖励 1）

   用 REINFORCE 的形式

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}\big[R(\tau)\,\nabla_\theta \log p_\theta(\tau)\big]$$

   这里

$$\log p_\theta(\tau_k) = k\log\theta + \log(1-\theta) \;\Rightarrow\; \nabla_\theta \log p_\theta(\tau_k) = \frac{k}{\theta} - \frac{1}{1-\theta}.$$

   因此

$$\nabla_\theta J(\theta) = \sum_{k=0}^{\infty}(1-\theta)\theta^k \; k\left(\frac{k}{\theta} - \frac{1}{1-\theta}\right) = \frac{1-\theta}{\theta}\sum_{k=0}^{\infty} k^2\theta^k - \sum_{k=0}^{\infty} k\theta^k.$$

用几何级数恒等式（提示）：

$$\sum_{k=0}^{\infty} k\alpha^k = \frac{\alpha}{(1-\alpha)^2}, \qquad \sum_{k=0}^{\infty} k^2\alpha^k = \frac{\alpha(1+\alpha)}{(1-\alpha)^3},$$

代入 $\alpha = \theta$ 得

$$\nabla_\theta J(\theta) = \frac{1-\theta}{\theta}\cdot\frac{\theta(1+\theta)}{(1-\theta)^3} - \frac{\theta}{(1-\theta)^2} = \frac{1+\theta}{(1-\theta)^2} - \frac{\theta}{(1-\theta)^2} = \boxed{\frac{1}{(1-\theta)^2}}.$$

(b) Compute the expected return of the policy $\mathbb{E}_{\tau\sim\pi_\theta} R(\tau)$ directly. Compute the gradient of this expression with respect to θ and verify that this matches the policy gradient.

Solution:

$$J(\theta) = \mathbb{E}[k] = \sum_{k=0}^{\infty} k \; \Pr(K=k) = \sum_{k=0}^{\infty} k\,(1-\theta)\theta^k = (1-\theta)\sum_{k=0}^{\infty} k\,\theta^k, \quad 0<\theta<1.$$

把几何级数对 θ 求导即可得到 $\sum k\theta^k$：

1. 先用基本几何级数

$$G(\theta) = \sum_{k=0}^{\infty}\theta^k = \frac{1}{1-\theta}.$$

2. 求导：

$$G'(\theta) = \sum_{k=0}^{\infty} k\,\theta^{k-1} = \frac{1}{(1-\theta)^2}.$$

3. 乘以 θ：

$$\theta G'(\theta) = \sum_{k=0}^{\infty} k\,\theta^k = \frac{\theta}{(1-\theta)^2}.$$

所以

$$J(\theta) = (1-\theta)\sum_{k=0}^{\infty} k\,\theta^k = (1-\theta)\cdot\frac{\theta}{(1-\theta)^2} = \frac{\theta}{1-\theta}.$$

直接求导可以，直接对 $J(\theta) = \dfrac{\theta}{1-\theta}$ 求导就行：

$$\frac{dJ}{d\theta} = \frac{(1-\theta)\cdot 1 - \theta\cdot(-1)}{(1-\theta)^2} = \frac{1}{(1-\theta)^2}, \qquad 0<\theta<1.$$

2. Compute the variance of the policy gradient in closed form and describe the properties of the variance with respect to θ. For what value(s) of θ is variance minimal? Maximal? (Once you have an exact expression for the variance you can eyeball the min/max).

   Hint: Once you have it expressed as a sum of terms $P(\theta)/Q(\theta)$ where P and Q are polynomials, you can use a symbolic computing program (Mathematica, SymPy, etc) to simplify to a single rational expression.

   Solution: 策略梯度的方差定义为：

$$\mathrm{Var}(\nabla_\theta J(\theta)) = \mathbb{E}_{\tau\sim\pi_\theta}\left[(R(\tau)\nabla_\theta \log p_\theta(\tau))^2\right] - \left(\mathbb{E}_{\tau\sim\pi_\theta}\left[R(\tau)\nabla_\theta \log p_\theta(\tau)\right]\right)^2$$

   其中：

   * $R(\tau)$ 是轨迹 τ 的回报（奖励的总和）。* $\nabla_\theta \log p_\theta(\tau)$ 是在策略 π_θ 下，轨迹 τ 的对数概率的梯度。

   步骤 1：计算 $\nabla_\theta \log p_\theta(\tau_k)$

   从之前的结果我们知道，轨迹 τ_k 的对数概率是：

$$\log p_\theta(\tau_k) = k\log\theta + \log(1-\theta)$$

   对 θ 求梯度得到：

$$\nabla_\theta \log p_\theta(\tau_k) = \frac{k}{\theta} - \frac{1}{1-\theta}$$

   步骤 2：方差公式中的第一项

   现在，我们计算 $R(\tau)\nabla_\theta \log p_\theta(\tau)$ 的平方的期望值：

$$\mathbb{E}_{\tau\sim\pi_\theta}\left[(R(\tau)\nabla_\theta \log p_\theta(\tau))^2\right]$$

   代入 $R(\tau_k) = k$ 和 $\nabla_\theta \log p_\theta(\tau_k) = \frac{k}{\theta} - \frac{1}{1-\theta}$，得到：

$$\mathbb{E}_{\tau\sim\pi_\theta}\left[\left(k\left(\frac{k}{\theta} - \frac{1}{1-\theta}\right)\right)^2\right]$$

   这可以化简为：

$$\mathbb{E}_{\tau\sim\pi_\theta}\left[k^2\left(\frac{k}{\theta} - \frac{1}{1-\theta}\right)^2\right]$$

   我们需要对所有可能的轨迹 τ_k 进行求和，其概率为 $p_\theta(\tau_k) = (1-\theta)\theta^k$。所以我们得到：

$$\mathbb{E}_{\tau\sim\pi_\theta}\left[\left(k^2\left(\frac{k}{\theta} - \frac{1}{1-\theta}\right)^2\right)\right] = \sum_{k=0}^{\infty}(1-\theta)\theta^k \cdot k^2\left(\frac{k}{\theta} - \frac{1}{1-\theta}\right)^2$$

   步骤 3：方差公式中的第二项

   方差的第二项是策略梯度的平方，我们之前计算过期望的策略梯度为：

$$\nabla_\theta J(\theta) = \frac{1}{(1-\theta)^2}$$

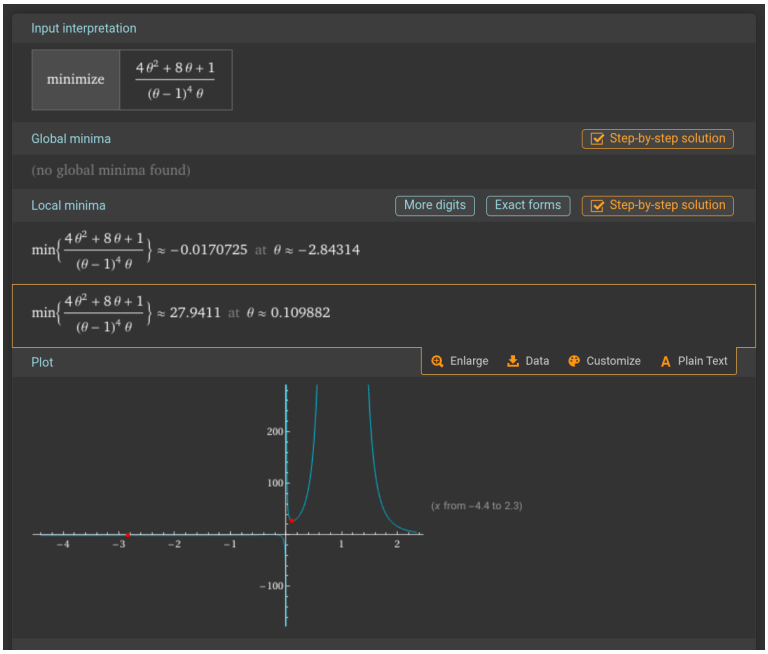


Figure 9: Variance of Policy Gradient with respect to θ. The variance is minimal near $\theta = 0.11$ and increases sharply as θ approaches 0 or 1.

因此，第二项是：

$$\left(\frac{1}{(1-\theta)^2}\right)^2 = \frac{1}{(1-\theta)^4}$$

步骤 4：完整的方差表达式

最终，方差就是第一项和第二项之差：

$$\mathrm{Var}(\nabla_\theta J(\theta)) = \sum_{k=0}^{\infty}(1-\theta)\theta^k \cdot k^2 \left(\frac{k}{\theta} - \frac{1}{1-\theta}\right)^2 - \frac{1}{(1-\theta)^4}$$

简化后

The simplified result of the expression is:

$$\frac{1+8\theta+4\theta^2}{\theta(1-\theta)^4}$$

带入 mathematica：

3. Apply return-to-go as an advantage estimator.

   (a) Write the modified policy gradient and confirm that it is unbiased.

   Solution: REINFORCE 的 reward-to-go 版：

$$\nabla_\theta J(\theta) = \mathbb{E}_\tau \left[\sum_t R_t \nabla_\theta \log \pi_\theta(a_t|s_t) \right], \quad R_t = \sum_{t' \geq t} r_{t'}$$

   对轨迹 τ_k：

   * 第 $t = 1, \ldots, k$ 步动作都是 a_1，其 **reward-to-go** 为

$$R_t = k - (t-1) = k - t + 1$$

   并且 $\nabla_\theta \log \pi_\theta(a_1|s_1) = \nabla_\theta \log \theta = \frac{1}{\theta}$。* 最后一步 $t = k+1$ 做 a_2，$R_{k+1} = 0$，而 $\nabla_\theta \log \pi_\theta(a_2|s_1) = \nabla_\theta \log(1-\theta) = -\frac{1}{1-\theta}$，但乘 0 仍为 0。

   于是，对固定的 k，和式只需要前 k 项：

$$\sum_{t=1}^{k} R_t \nabla_\theta \log \pi_\theta(a_t|s_t) = \sum_{t=1}^{k} (k-t+1) \cdot \frac{1}{\theta} = \frac{1}{\theta} \sum_{m=1}^{k} m = \frac{1}{\theta} \cdot \frac{k(k+1)}{2}.$$

   再对所有 k 取期望（按 $(1-\theta)\theta^k$ 加权）：

$$\begin{aligned} \nabla_\theta J(\theta) &= \sum_{k=0}^{\infty} (1-\theta)\theta^k \cdot \left(\frac{1}{\theta} \cdot \frac{k(k+1)}{2} \right) \\ &= \frac{1-\theta}{2\theta} \sum_{k=0}^{\infty} k(k+1)\theta^k. \end{aligned}$$

   用几何级数恒等式

$$\sum_{k=0}^{\infty} k\theta^k = \frac{\theta}{(1-\theta)^2}, \qquad \sum_{k=0}^{\infty} k^2\theta^k = \frac{\theta(1+\theta)}{(1-\theta)^3},$$

   得

$$\sum_{k=0}^{\infty} k(k+1)\theta^k = \sum (k^2+k)\theta^k = \frac{\theta(1+\theta)}{(1-\theta)^3} + \frac{\theta}{(1-\theta)^2} = \frac{2\theta}{(1-\theta)^3}.$$

   代回去：

$$\nabla_\theta J(\theta) = \frac{1-\theta}{2\theta} \cdot \frac{2\theta}{(1-\theta)^3} = \boxed{\frac{1}{(1-\theta)^2}}.$$

   (b) Compute the variance of the return-to-go policy gradient and plot it on $[0, 1]$ alongside the variance of the original estimator.

   Solution: 计算方差

$$\mathrm{Var}[g] = \mathbb{E}[g^2] - \big(\mathbb{E}[g]\big)^2.$$

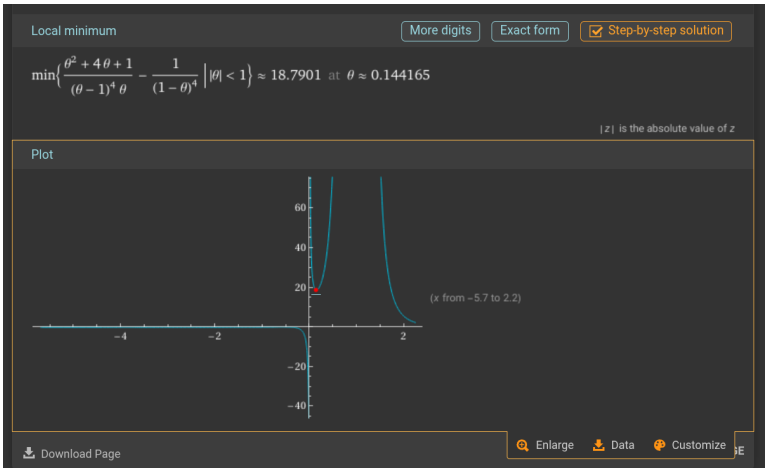


Figure 10: Variance of Policy Gradient with rtg. The variance is minimal near $\theta = 0.14$ and increases sharply as θ approaches 0 or 1.

先算二阶矩：

$$\mathbb{E}[g^2] = \sum_{k=0}^{\infty}(1-\theta)\theta^k \left(\frac{k(k+1)}{2\theta}\right)^2 = \frac{1-\theta}{4\theta^2}\sum_{k=0}^{\infty}\theta^k\,(k(k+1))^2.$$

把 $(k(k+1))^2 = k^4 + 2k^3 + k^2$，用标准几何求和恒等式

$$\sum_{k\geq 0} k^2\alpha^k = \frac{\alpha(1+\alpha)}{(1-\alpha)^3}, \quad \sum_{k\geq 0} k^3\alpha^k = \frac{\alpha(1+4\alpha+\alpha^2)}{(1-\alpha)^4}, \quad \sum_{k\geq 0} k^4\alpha^k = \frac{\alpha(1+11\alpha+11\alpha^2+\alpha^3)}{(1-\alpha)^5},$$

代入 $\alpha = \theta$ 可得

$$\mathbb{E}[g^2] = \frac{\theta^2+4\theta+1}{\theta(1-\theta)^4}.$$

于是

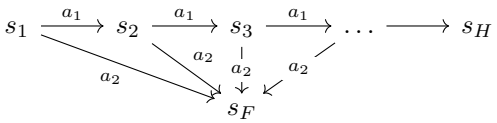
$$\boxed{\mathrm{Var}[g] = \mathbb{E}[g^2] - \big(\mathbb{E}[g]\big)^2 = \frac{\theta^2+4\theta+1}{\theta(1-\theta)^4} - \frac{1}{(1-\theta)^4} = \boxed{\frac{\theta^2+3\theta+1}{\theta(1-\theta)^4}}}.$$

直观解释

* θ 很小时：很少继续，轨迹大多很短，但 $\nabla \log \pi(a_1|s_1) = 1/\theta$ 系数巨大，导致方差暴涨；* θ 很接近 1 时：轨迹很长，累加的 RtG（以及 $(1-\theta)^{-4}$）把方差再次推高；* 在中间存在最佳折中（约 0.144），方差最小。

带入 mathematica：

4. Consider a finite-horizon H-step MDP with sparse reward:



The agent receives reward $R_{\max}$ if it arrives at s_H and reward 0 if it arrives at s_F (a terminal state). In other words, the return for a trajectory τ is given by:

$$R(\tau) = \begin{cases} 1 & \tau \text{ ends at } s_H \\ 0 & \tau \text{ ends at } s_F \end{cases}$$

Using the same policy parametrization as above, consider off-policy policy gradients via importance sampling. Assume we want to compute policy gradients for a policy π_θ with samples drawn from $\pi_{\theta'}$.

(a) Write the policy gradient with importance sampling.

Solution: * 每一步都有两个可选动作：a_1：继续往右走；a_2：直接掉到失败终止态 s_F。

* **只有到达最右侧的 s_H**（连续执行 a_1 直到最后一步）时才会得到奖励 $R_{\max}$（题目中取 1）。

* 如果中途任何一步选了 a_2，轨迹立即结束在 s_F，奖励 $= 0$。

—

所以，确实：

> 除非整条轨迹全是 a_1，否则 $R(\tau) = 0$。

换句话说：

$$R(\tau) = \begin{cases} 1, & \text{if } \tau = (a_1, a_1, \dots, a_1) \text{ (length } H), \\ 0, & \text{otherwise.} \end{cases}$$

—

对应概率和策略

若每一步采取 a_1 的概率为 $\pi_\theta(a_1|s_t) = \theta$，则到达成功轨迹的概率为：

$$p_\theta(\text{success}) = \theta^H.$$

其余所有轨迹（至少一个 a_2）的 $R(\tau) = 0$。

—

Off-policy 情形

我们想计算目标策略 π_θ 的梯度，但样本是从别的策略 $\pi_{\theta'}$ 采的。

于是根据 **importance sampling** 修正分布差异：

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau\sim\pi_{\theta'}}\Big[R(\tau)\,\frac{p_\theta(\tau)}{p_{\theta'}(\tau)}\,\nabla_\theta \log p_\theta(\tau)\Big].$$

—

具体到这道题

因为只有一条轨迹（全 a_1）有奖励，所以可以直接写出：

* 成功轨迹的概率 $p_{\theta'}(\tau_{\text{succ}}) = (\theta')^H$ * 目标策略下的概率 $p_\theta(\tau_{\text{succ}}) = \theta^H$ * 对数梯度 $\nabla_\theta \log p_\theta(\tau_{\text{succ}}) = \frac{H}{\theta}$

代入：

$$\nabla_\theta J(\theta) = R_{\max} \frac{p_\theta(\tau_{\text{succ}})}{p_{\theta'}(\tau_{\text{succ}})} \nabla_\theta \log p_\theta(\tau_{\text{succ}})\, p_{\theta'}(\tau_{\text{succ}}) = R_{\max}\, \theta^H \frac{H}{\theta} = \boxed{R_{\max}\, H\, \theta^{H-1}}.$$

(b) Compute its variance.

Solution:

* 有限地平线 H。* 只在“全程都选 a_1”的成功轨迹时 $R(\tau) = 1$，否则 $R(\tau) = 0$。* 目标策略：$\pi_\theta(a_1|s) = \theta$，行为策略：$\pi_{\theta'}(a_1|s) = \theta'$。* 重要性权重（整条轨迹）：$w(\tau) = \frac{p_\theta(\tau)}{p_{\theta'}(\tau)} = \prod_{t=1}^{H} \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta'}(a_t|s_t)}$。

成功轨迹（全为 a_1）时

$$w(\tau_{\text{succ}}) = \left(\tfrac{\theta}{\theta'}\right)^H, \qquad \nabla_\theta \log p_\theta(\tau_{\text{succ}}) = \frac{H}{\theta}.$$

失败轨迹时 $R = 0$，估计量为 0。

—

Off-policy 梯度估计量与方差

用 REINFORCE 的 off-policy 形式

$$\hat{g} \;=\; R(\tau)\, w(\tau)\, \nabla_\theta \log p_\theta(\tau), \qquad \tau \sim \pi_{\theta'}.$$

因此

$$\hat{g} \;=\; \begin{cases} \dfrac{H}{\theta}\Big(\dfrac{\theta}{\theta'}\Big)^H = \dfrac{H\,\theta^{H-1}}{(\theta')^H}, & \text{若成功（概率 } (\theta')^H) \\ 0, & \text{若失败（概率 } 1-(\theta')^H) \end{cases}$$

* 期望（无偏性）：

$$\mathbb{E}[\hat{g}] = (\theta')^H \cdot \frac{H\,\theta^{H-1}}{(\theta')^H} = \boxed{H\,\theta^{H-1}}.$$

* 二阶矩：

$$\mathbb{E}[\hat{g}^2] = (\theta')^H \cdot \Big(\frac{H\,\theta^{H-1}}{(\theta')^H}\Big)^2 = \frac{H^2\,\theta^{2H-2}}{(\theta')^H}.$$

* **方差**：

$$\mathrm{Var}[\hat{g}] = \mathbb{E}[\hat{g}^2] - \big(\mathbb{E}[\hat{g}]\big)^2 = \boxed{H^2\,\theta^{2H-2}\Big(\frac{1}{(\theta')^H} - 1\Big)}.$$

性质（随 θ' 与 θ 的变化）

* 固定 θ：方差是 θ' 的单调函数，$\theta' \to 1$ 最小、$\theta' \to 0$ 爆炸。

* **最小值**：$\theta' = 1 \Rightarrow \mathrm{Var} = 0$（行为策略一定成功，估计量是常数）。* **最大值**：$\theta' \to 0^+ \Rightarrow \mathrm{Var} \to \infty$（极端分布失配）。

* 固定 θ'：方差随 θ^{2H-2} 缩放，$\theta \to 0$ 方差趋 0（但此时真梯度也 $\to 0$）。

10 Survey

Please estimate, in minutes, for each problem, how much time you spent (a) writing code and (b) waiting for the results. This will help us calibrate the difficulty for future homeworks.

- **Policy Gradients:**
- **Neural Network Baseline:**
- **Generalized Advantage Estimation:**
- **Hyperparameters and Sample Efficiency:**
- **Humanoid:**
- **Humanoid:**
- **Analysis – applying policy gradients:**
- **Analysis – PG variance:**
- **Analysis – return-to-go:**
- **Analysis – importance sampling:**