CS342 - Section 10

- Homework 9
  - Due last night, can still turn it in with penalties
  - How much is your loss?
- Homework 10
  - Will be posted later today
  - Train agent to act!
  - Setting up SuperTux important
- Homework 8
  - Grades are out

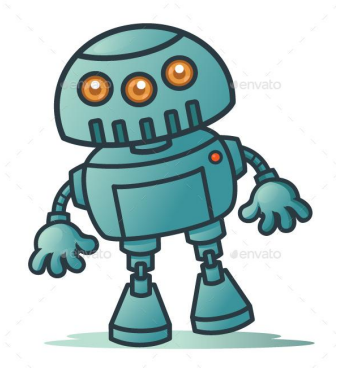
Questions?

Agenda

- Imitation Learning
- Recap of Reinforcement Learning

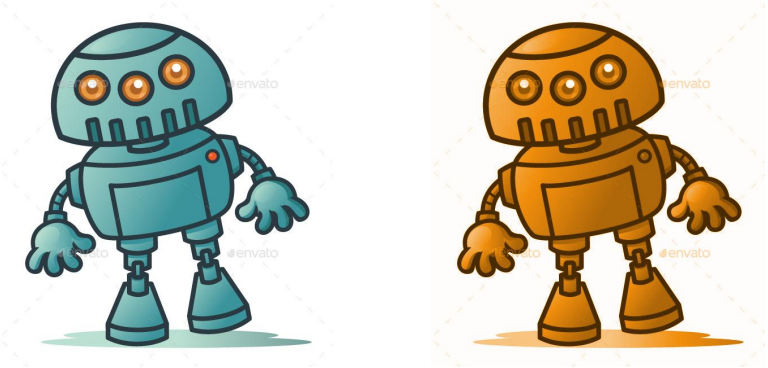
Learning to Act

- How do agents learn to act?

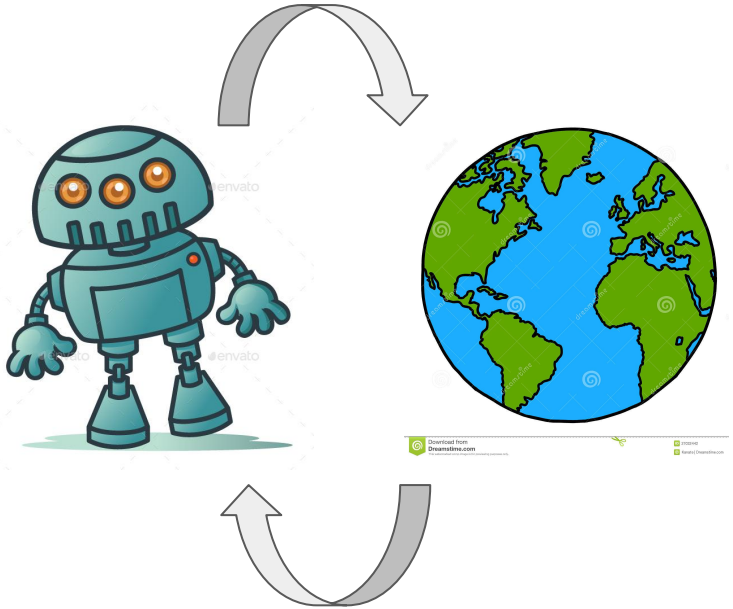


Learning to Act

- From Imitation

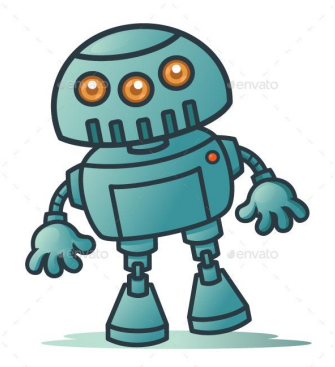


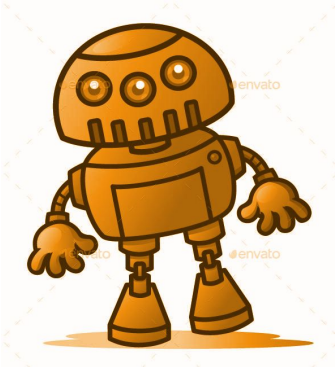
- From Interaction



Imitation Learning

- Given expert trajectories, train agent to mimic expert behavior





Imitation Learning

1. Large state space in RL requires a lot of data to train
2. Allows for direct supervision

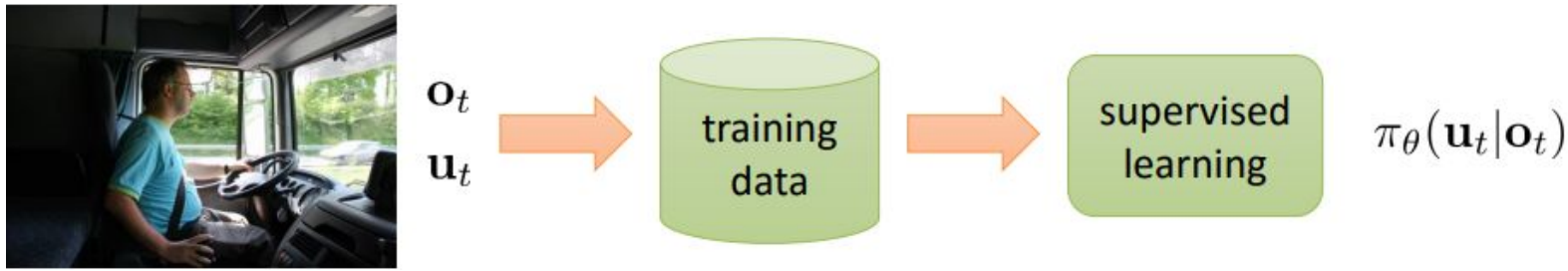


Image Credit: https://katefvision.github.io/katefSlides/immitation_learning_I_katef.pdf

Problems with Imitation Learning

- Lack of enough expert training data
- Mismatch in distribution of states for expert policy and the model policy
  - Similar to mismatch in training and test data.
- Errors in a sequence grow quadratically[1]
- How can we get around these problems?

[1] Ross, Stéphane, Geoffrey Gordon, and Drew Bagnell. "A reduction of imitation learning and structured prediction to no-regret online learning." In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 627-635. 2011.

DAgger: Dataset Aggregation

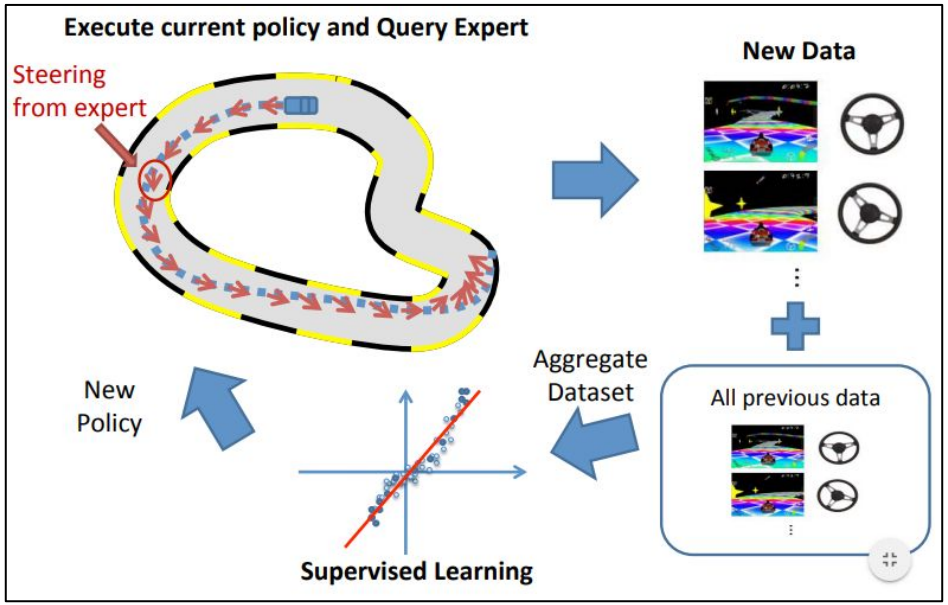


Image Credit: https://katefvision.github.io/katefSlides/immitation_learning_I_katef.pdf

DAgger: Dataset Aggregation

1. Train a policy P_i using the given dataset D,
2. Generate new samples using the policy P_i,
3. Get Labels for new samples and add to the dataset D
4. Go to 1.

[1] Ross, Stéphane, Geoffrey Gordon, and Drew Bagnell. "A reduction of imitation learning and structured prediction to no-regret online learning." In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 627-635. 2011.

DAgger: Applications

- Super Tux Kart racing game
- Car stays on track after 15 iterations.
- Do consider this for your final project

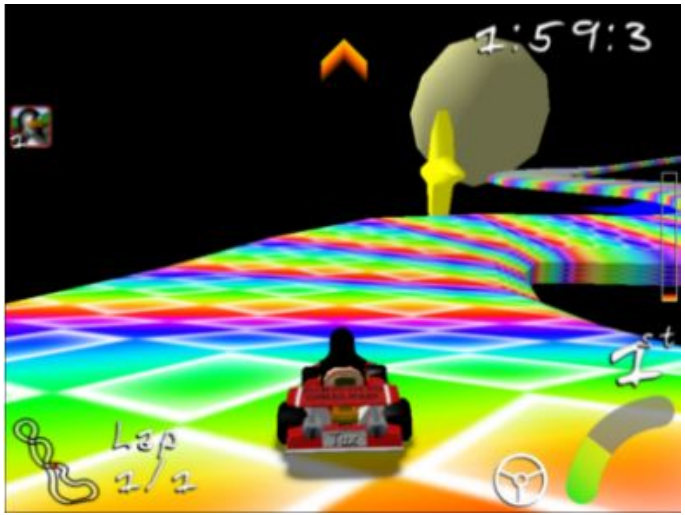
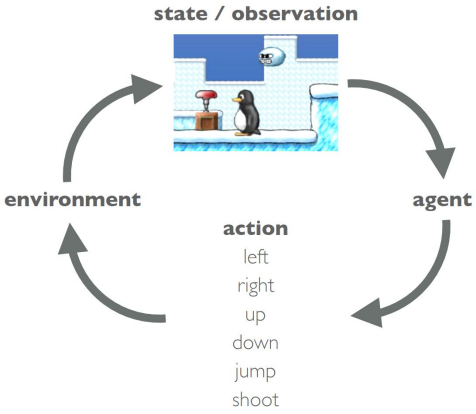


Image Credit: Ross, Stéphane, Geoffrey Gordon, and Drew Bagnell. "A reduction of imitation learning and structured prediction to no-regret online learning." In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 627-635. 2011.

Recap of RL from lecture

- Problem: Find the policy that takes actions in the environment to maximize cumulative reward



Recap of RL from lecture

- Actions: Actions (a) that agent can take in the environment
- States: States (s) that represent the internal state of the environment/agent
- Policy: A function (π) that maps states/observations to actions
- Reward: A function that maps (s,a) to a number, meaning how desirable is the state-action transition

Recap of RL from lecture

RL is a framework, and its goal is to find the optimal policy

$$\theta^* = \text{argmax}_\theta E_{a\sim p(\tau)}[\sum_t r(s_t, a_t)]$$

where $p(\tau)=p(s_0)\prod_t \pi_\theta(a_t|s_t)T(s_{t+1}|s_t,a_t)$, $\tau=\{(s_1,a_1),(s_2,a_2), \dots\}$

π_θ is the policy distribution, T is the transition probability, r is the reward function

Recap of RL from lecture

- How to find the policy?
  - Model-based : Explicitly model the environment
  - Model-free : Do not explicitly model the environment
    - On-policy
      - Policy Gradient, Actor-Critic
    - Off-policy
      - Value function approximation

Recap of RL from lecture

- How to find the policy?
  - Model-based : given or estimate the transition model, and use it to plan
    - Policy/Value Iteration : need T(s,a,s') transition probability
    - Linear Quadratic Regulator (LQR) : need closed-form and linear forward model $f(s_t,a_t)$ and quadratic reward (back when how missiles are controlled!)
    - Guided Policy Search (GPS) : iLQR + Supervised Learning, still need model (learn door opening in 10 trajectories!)
    - AlphaGo : Use the learned value function to plan using Monte Carlo Tree Search (beat the human champion!)

Recap of RL from lecture

- How to find the policy?
  - Model-based : given or estimate the transition model, and use it to plan

- Problems:
  - Finding the optimal policy often does not need a full model. E.x. You don’t need to know law of physics to run, jump, pour water into a mug etc.
  - Lots of model-based methods has strong assumptions
    - Guided Policy Search need full robot kinematics model! Soft robots cannot use this method.
    - AlphaGo types of tree search methods requires a known highly structured forward model (E.x, gomoku)

Recap of RL from lecture

- How to find the policy?
  - Model-free : On-policy
    - Policy gradient : directly backpropagate the gradient of objective into policy
    - E.x REINFORCE, Trust Region Policy Optimization (TRPO)

$\text{argmax}_\theta J(\theta)$, where $J(\theta)$ is the expected cumulative reward

$$\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$$

Recap of RL from lecture

- How to find the policy?
  - Model-free : Off-policy
    - Q-Learning: Estimate Q(s,a), which represents the cumulative reward if taking action a at state s and acting optimally later
    - E.x Deep Q-Networks (DQN), Deep Deterministic Policy Gradient (DDPG) (DDPG is not policy gradient!)

$\text{argmax}_\theta J(\theta)$, where $J(\theta)$ is the expected cumulative reward

$Q(s,a) = r(s,a) + \max_{a'} Q(s',a')$

set $\pi(s)$ to max $Q(s,a)$

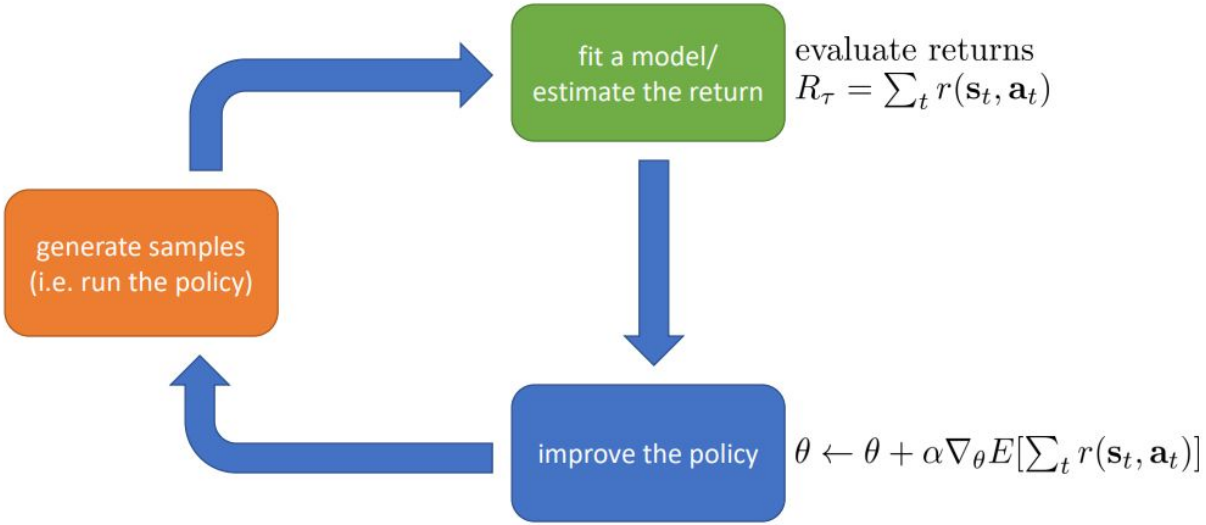
Recap of RL from lecture

- How to find the policy?
  - Model-free : On-policy
    - Actor-Critic: Policy gradient+Value function, approximate V(s) or Q(s,a) of the current policy to get gradient
    - E.x. Async Actor-Critic (A3C), Soft Actor-Critic (SAC)

$\text{argmax}_\theta J(\theta)$, where $J(\theta)$ is the expected cumulative reward

$\theta \leftarrow \theta + \alpha \nabla_\theta E(Q(s,a))$

Policy Gradient



Slides from CS294-112

Policy Gradient

- We improve the policy by directly backpropagating the gradient

$$\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$$

$$J(\theta) = E_{\tau \sim \pi\theta}[r(\tau)]$$

- $\nabla_\theta J(\theta) = \nabla_\theta \int \pi_\theta(\tau) r(\tau) d\tau$

$$= \int \nabla_\theta \pi_\theta(\tau) r(\tau) d\tau$$

$$= \int \pi_\theta(\tau) \nabla_\theta \log \pi_\theta(\tau) r(\tau) d\tau$$

$$= E_{\tau \sim \pi\theta}[\nabla_\theta \log \pi_\theta(\tau) r(\tau)] \Longrightarrow$$

$\nabla_x \log f(x) = 1/f(x) \nabla_x f(x)$

weight actions by reward

Policy Gradient

- We improve the policy by directly backpropagating the gradient

$$\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$$

$$J(\theta) = E_{\tau \sim \pi\theta}[r(\tau)]$$

- $\nabla_\theta J(\theta) = E_{\tau \sim \pi\theta}[\nabla_\theta \log \pi_\theta(\tau) r(\tau)]$

$$= E_{\tau \sim \pi\theta}[\nabla_\theta (p(s_0) + \sum_t \log \pi_\theta(s_t|a_t) + \sum_t \log T(s_{t+1}|s_t,a_t)) r(\tau)]$$

$$= E_{\tau \sim \pi\theta}[(\nabla_\theta \sum_t \log \pi_\theta(s_t|a_t)) (\sum_t r(s_t,a_t))]$$

Policy Gradient

- We improve the policy by directly backpropagating the gradient

$$\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$$

$$J(\theta) = E_{\tau \sim \pi\theta}[r(\tau)]$$

- $\nabla_\theta J(\theta) = E_{\tau \sim \pi\theta}[\nabla_\theta \log \pi_\theta(\tau) r(\tau)]$

$$= E_{\tau \sim \pi\theta}[\nabla_\theta (p(s_0) + \sum_t \log \pi_\theta(s_t|a_t) + \sum_t \log T(s_{t+1}|s_t,a_t)) r(\tau)]$$

$$= E_{\tau \sim \pi\theta}[(\nabla_\theta \sum_t \log \pi_\theta(s_t|a_t)) (\sum_t r(s_t,a_t))]$$

$$\approx 1/N \sum_i (\nabla_\theta \sum_t \log \pi_\theta(s_t|a_t)) (\sum_t r(s_{i,t},a_{i,t})) \Longrightarrow$$ REINFORCE!

Policy Gradient

- We improve the policy by directly backpropagating the gradient

$$\theta \leftarrow \theta + \nabla_{\theta} J(\theta)$$

$$J(\theta) = E_{\tau \sim \pi\theta}[r(\tau)]$$

- Problem
  - log π(s|a) weighted directly by rewards, has high variance
  - Every data point used only once then thrown away

Policy Gradient

- We improve the policy by directly backpropagating the gradient

$$\theta \leftarrow \theta + \nabla_\theta J(\theta)$$

$$J(\theta) = E_{\tau \sim \pi\theta}[r(\tau)]$$

- Variance reduction
  - $\nabla_\theta J(\theta) = E_{\tau \sim \pi\theta}[\nabla_\theta \log \pi_\theta(\tau) r(\tau)] \rightarrow \nabla_\theta J(\theta) = E_{\tau \sim \pi\theta}[\nabla_\theta \log \pi_\theta(\tau)(r(\tau) - b)]$
  - Are we actually allowed to do that?

Policy Gradient

- We improve the policy by directly backpropagating the gradient

$$\theta \leftarrow \theta + \nabla_\theta J(\theta)$$

$$J(\theta) = E_{\tau \sim \pi\theta}[r(\tau)]$$

- Variance reduction
  - $\nabla_\theta J(\theta) = E_{\tau \sim \pi\theta}[\nabla_\theta \log \pi_\theta(\tau) r(\tau)] \rightarrow \nabla_\theta J(\theta) = E_{\tau \sim \pi\theta}[\nabla_\theta \log \pi_\theta(\tau)(r(\tau) - b)]$
  - Still an unbiased gradient estimator, because

$$E_{\tau \sim \pi\theta}[\nabla_\theta \log \pi_\theta(\tau)(r(\tau) - b)] = \int \pi_\theta(\tau) \nabla_\theta \log \pi_\theta(\tau)(r(\tau) - b) d\tau$$

$$= \int \pi_\theta(\tau) \nabla_\theta \log \pi_\theta(\tau) r(\tau) d\tau - \int \pi_\theta(\tau) \nabla_\theta \log \pi_\theta(\tau) b d\tau$$

$$= E_{\tau \sim \pi\theta}[\nabla_\theta \log \pi_\theta(\tau) r(\tau)] - \int \nabla_\theta \pi_\theta(\tau) b d\tau$$

$$= E_{\tau \sim \pi\theta}[\nabla_\theta \log \pi_\theta(\tau) r(\tau)] - \nabla_\theta \int \pi_\theta(\tau) b d\tau = E_{\tau \sim \pi\theta}[\nabla_\theta \log \pi_\theta(\tau) r(\tau)]$$

Policy Gradient

- We improve the policy by directly backpropagating the gradient

$$\theta \leftarrow \theta + \nabla_\theta J(\theta)$$

$$J(\theta) = E_{\tau \sim \pi\theta}[r(\tau)]$$

- Variance reduction
  - $\nabla_\theta J(\theta) = E_{\tau \sim \pi\theta}[\nabla_\theta \log \pi_\theta(\tau) r(\tau)] \rightarrow \nabla_\theta J(\theta) = E_{\tau \sim \pi\theta}[\nabla_\theta \log \pi_\theta(\tau)(r(\tau) - b)]$
  - Variance is reduced, if we choose b to be reward weighed by gradient magnitude

$$\mathrm{Var}_{\tau \sim \pi\theta}[\nabla_\theta \log \pi_\theta(\tau)(r(\tau) - b)] = E_{\tau \sim \pi\theta}[\nabla_\theta \log \pi_\theta(\tau)(r(\tau) - b)^2] - E_{\tau \sim \pi\theta}[\nabla_\theta \log \pi_\theta(\tau) r(\tau)]^2$$

$$= \int \pi_\theta(\tau)(\nabla_\theta \log \pi_\theta(\tau)(r(\tau) - b))^2 d\tau - E_{\tau \sim \pi\theta}[\nabla_\theta \log \pi_\theta(\tau) r(\tau)]^2$$

$$= \int \pi_\theta(\tau)(\nabla_\theta \pi_\theta(\tau)(r(\tau) - b))^2 d\tau - E_{\tau \sim \pi\theta}[\nabla_\theta \log \pi_\theta(\tau) r(\tau)]^2$$

$$= E_{\tau \sim \pi\theta}[\nabla_\theta \pi_\theta(\tau)^2 (r(\tau) - b)^2] - E_{\tau \sim \pi\theta}[\nabla_\theta \log \pi_\theta(\tau) r(\tau)]^2$$

Policy Gradient

- We improve the policy by directly backpropagating the gradient

$$\theta \leftarrow \theta + \nabla_\theta J(\theta)$$

$$J(\theta) = E_{\tau \sim \pi\theta}[r(\tau)]$$

- Variance reduction
  - $\nabla_\theta J(\theta) = E_{\tau \sim \pi\theta}[\nabla_\theta \log \pi_\theta(\tau) r(\tau)] \rightarrow \nabla_\theta J(\theta) = E_{\tau \sim \pi\theta}[\nabla_\theta \log \pi_\theta(\tau)(r(\tau) - b)]$
  - Variance is reduced, if we choose b to be reward weighed by gradient magnitude

Note that $\mathrm{Var}_{\tau \sim \pi\theta}[\nabla_\theta \log \pi_\theta(\tau)(r(\tau) - b)] = E_{\tau \sim \pi\theta}[\nabla_\theta \pi_\theta(\tau)^2 (r(\tau) - b)^2] - E_{\tau \sim \pi\theta}[\nabla_\theta \log \pi_\theta(\tau) r(\tau)]^2$

$$\frac{d\mathrm{Var}(b)}{db} = \frac{dE_{\tau \sim \pi\theta}[\nabla_\theta \pi_\theta(\tau)^2 (r(\tau) - b)^2]}{db}$$

$$= dE_{\tau \sim \pi\theta}[\nabla_\theta \pi_\theta(\tau)^2 r(\tau)^2]/db - 2dE_{\tau \sim \pi\theta}[\nabla_\theta \pi_\theta(\tau)^2 r(\tau) b]/db + dE_{\tau \sim \pi\theta}[\nabla_\theta \pi_\theta(\tau)^2 b^2]/db$$

$$= 0 - 2d\, E_{\tau \sim \pi\theta}[\nabla_\theta \pi_\theta(\tau)^2 r(\tau) b]/db + 2b\, dE_{\tau \sim \pi\theta}[\nabla_\theta \pi_\theta(\tau)^2]$$

Policy Gradient

- We improve the policy by directly backpropagating the gradient

$$\theta \leftarrow \theta + \nabla_\theta J(\theta)$$

$$J(\theta) = E_{\tau \sim \pi\theta}[r(\tau)]$$

- Variance reduction
  - $\nabla_\theta J(\theta) = E_{\tau \sim \pi\theta}[\nabla_\theta \log \pi_\theta(\tau) r(\tau)] \rightarrow \nabla_\theta J(\theta) = E_{\tau \sim \pi\theta}[\nabla_\theta \log \pi_\theta(\tau)(r(\tau) - b)]$
  - Variance is reduced, if we choose b to be reward weighed by gradient magnitude

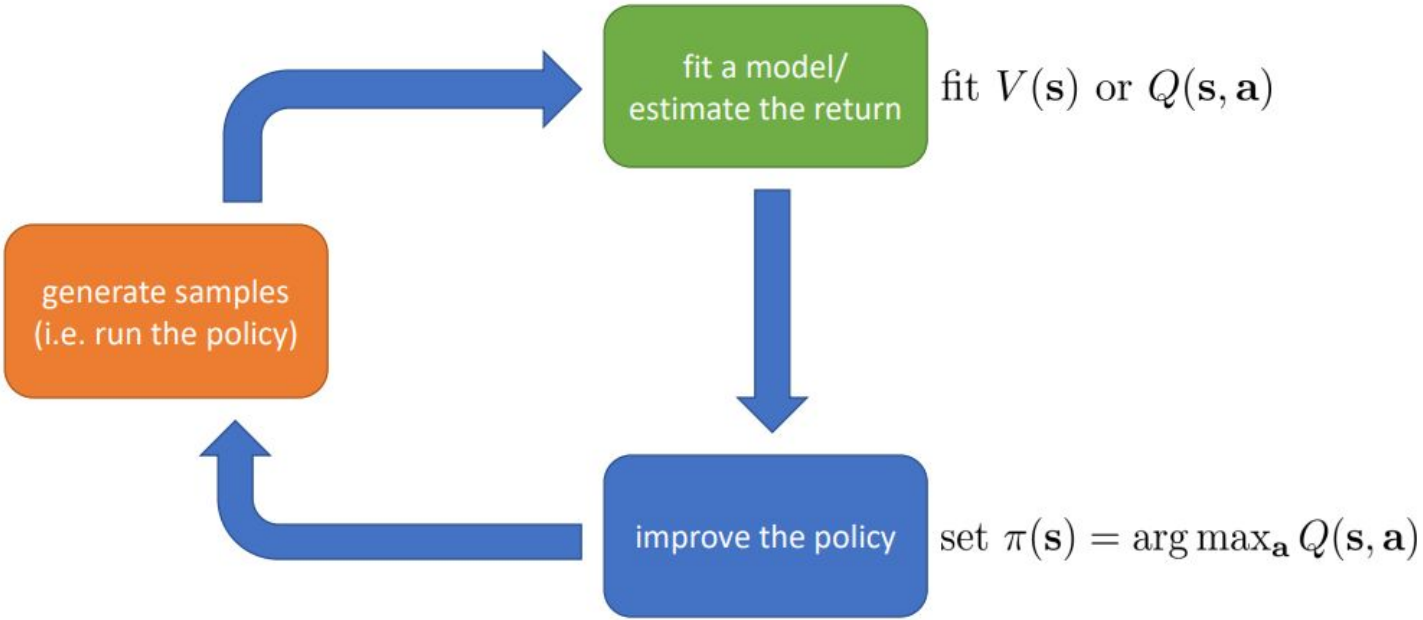
Note that $Var_{\tau \sim \pi\theta}[\nabla_\theta \log \pi_\theta(\tau)(r(\tau) - b)] = E_{\tau \sim \pi\theta}[\nabla_\theta \pi_\theta(\tau)^2 (r(\tau) - b)^2] - E_{\tau \sim \pi\theta}[\nabla_\theta \log \pi_\theta(\tau) r(\tau)]^2$

$dVar(b)/db = d\,(-2bE_{\tau \sim \pi\theta}[\nabla_\theta \pi_\theta(\tau)^2 r(\tau)] + b^2 E_{\tau \sim \pi\theta}[\nabla_\theta \pi_\theta(\tau)^2]) / db$

$= -2E_{\tau \sim \pi\theta}[\nabla_\theta \pi_\theta(\tau)^2 r(\tau)] + 2b\, E_{\tau \sim \pi\theta}[\nabla_\theta \pi_\theta(\tau)^2]$

Variance is minimized when $dVar(b)/db = 0$, i.e $b = E_{\tau \sim \pi\theta}[\nabla_\theta \pi_\theta(\tau)^2 r(\tau)] / E_{\tau \sim \pi\theta}[\nabla_\theta \pi_\theta(\tau)^2]$

Value functions/Q Learning



Slides from CS294-112

Value functions/Q-Learning

- We improve the policy by maximizing Q(s,a)

$$\text{Set } \pi(s) \text{ s.t. } \operatorname{argmax}_a Q(s,a)$$

$$Q(s,a) = r(s,a) + \operatorname{argmax}_{a'} Q(s',a')$$

Value functions/Q-Learning

- We improve the policy by maximizing Q(s,a)

$$\text{Set } \pi(s) \text{ s.t. } \text{argmax}_a\ Q(s,a)$$

$$Q(s,a) = r(s,a) + \text{argmax}_{a'} Q(s',a')$$

- How do we train Q?
  - Bellman Update:

Given (s,a,s',r)

$$y = r(s,a) + \max_{a'} Q_\Phi(s',a')$$

$$\Phi \leftarrow \text{argmin}_\Phi ||y - Q_\Phi(s,a)||_2$$

Value functions/Q-Learning

- We improve the policy by maximizing Q(s,a)

$$\text{Set } \pi(s) \text{ s.t. } \operatorname{argmax}_a Q(s,a)$$

$$Q(s,a) = r(s,a) + \operatorname{argmax}_{a'} Q(s',a')$$

- How do we train policy?
    - $\pi(s) = \operatorname{argmax}_a Q(s,a)$
        - Discrete case: directly argmax a
        - Continuous case : argmax of a is intractable

Value functions/Q-Learning

- We improve the policy by maximizing Q(s,a)

$$\text{Set } \pi(s) \text{ s.t. } \operatorname{argmax}_a Q(s,a)$$

$$Q(s,a) = r(s,a) + \operatorname{argmax}_{a'} Q(s',a')$$

- How do we train policy?
  - $\pi_\theta(s) = \operatorname{argmax}_a Q_\theta(s,a)$
    - Continuous case : $\theta \leftarrow \theta + \nabla_\theta Q(s,a)|_{s=s',\ a=a'}$

Recap of RL from lecture

- How to find the policy?
  - Model-free : On-policy
    - Policy Gradient
    - Value function/Q-Learning
    - Actor-Critic
- Problem
  - Model-free methods, especially the on-policy ones, requires too many samples!
  - Model-free methods have worse generalization potentials than model-based ones