Project Presentations

- Logistics
  - Apr. 1, 3, 5 – 10 minutes per person
  - Sign up at Google sheets link
    - First-come-first-serve; feel free to negotiate among yourselves
- Purpose
  - Share what you have learned with the class
  - Stimulate research ideas and collaborations
  - Participation marks for the project
- Content
  - Project does not need to be finished at this point
    - Summarize what you have done, and what remains to be done
  - Position your presentation in the context of this course
- Audience
  - Your peers
  - Engineers, mathematicians, and computer scientists who know the basics about the algorithms presented in this course

Policy-Based and Actor-Critic RL

CMPT 882

Mar. 21

Outline For The Week

- Basic ideas in RL
  - Value functions and value iteration
  - Policy evaluation and policy improvement
- Model-free RL
  - Monte-Carlo and temporal differencing policy evaluation
  - ϵ-greedy policy improvement
- Function Approximation
- Policy Gradient

Categories

- Model-based
  - Explicitly involves an MDP model
- Model-free
  - Does not involve an MDP model

- Value based
  - Learns value function, and derives policy from value function
- Policy based
  - Learns policy without value function
- Actor critic
  - Incorporates both value function and policy

Policy Gradients

- If we executed a policy π_θ from state s_0, we obtain a trajectory
  - $\tau \coloneqq (s_0, a_0, s_1, a_1, \dots)$
  - Note: this is a random variable

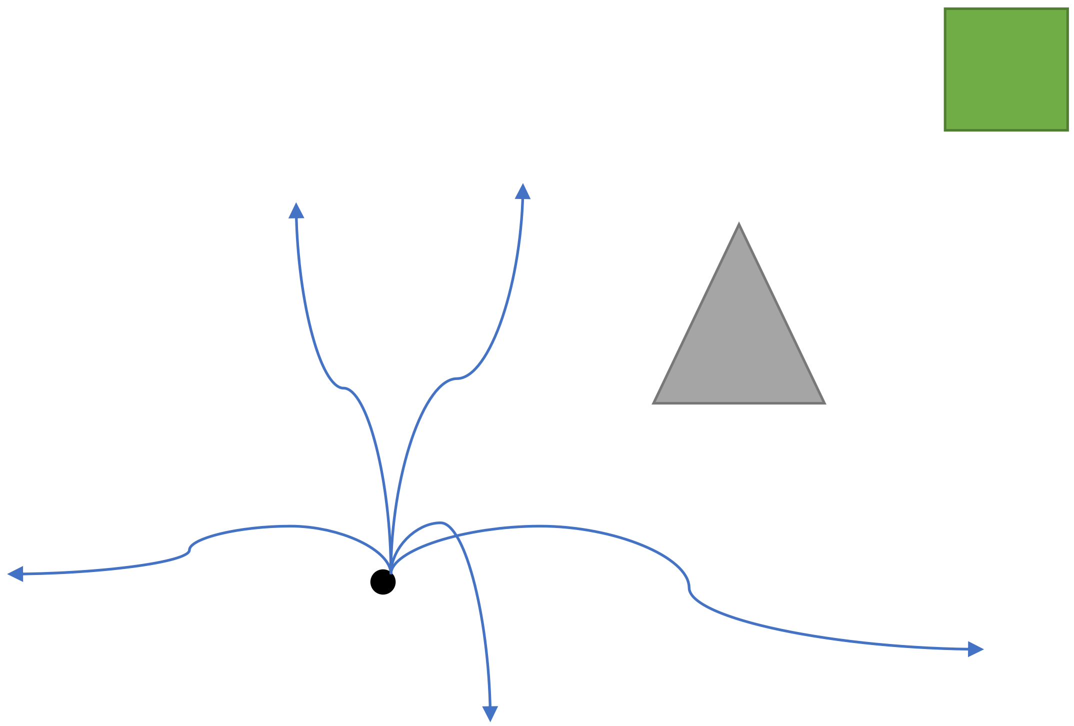
- The return is given by $R(\tau) \coloneqq \sum_{t \geq 0} \gamma^t r(s_t, a_t)$
  - Also a random variable

- Expected return given parameters θ: $J(\theta) \coloneqq \mathbb{E}_{\tau \sim p(\tau;\theta)}[R(\tau)]$

- Parameters for the optimal policy:
  - $\theta^* = \arg\max_\theta \mathbb{E}_{\tau \sim p(\tau;\theta)}[R(\tau)]$

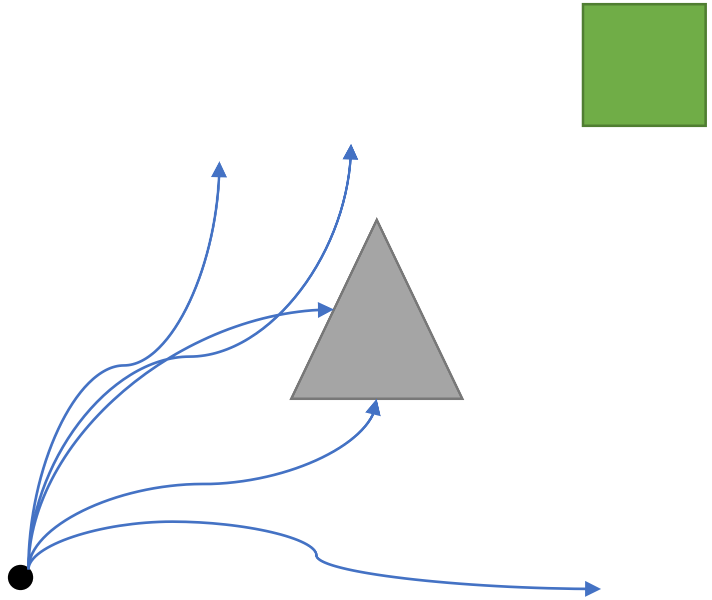
Policy Gradients

- Strategy: differentiate $J(\theta)$ w.r.t. θ and perform stochastic gradient ascent
  - Do this in a way that is model-free and computationally tractable



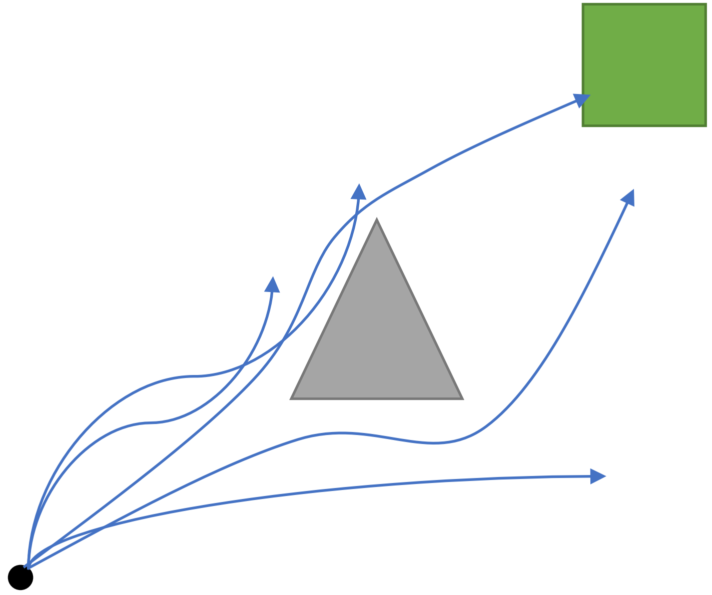
Policy Gradients

- Strategy: differentiate $J(\theta)$ w.r.t. θ and perform stochastic gradient ascent
  - Do this in a way that is model-free and computationally tractable



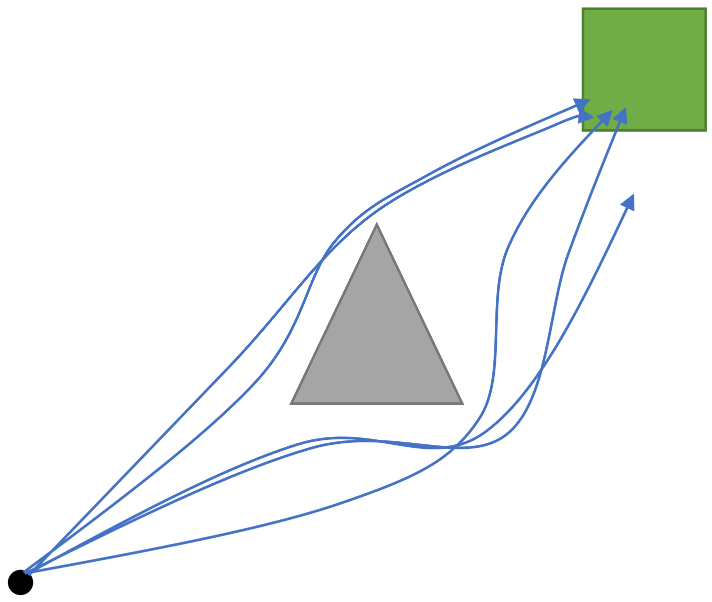
Policy Gradients

- Strategy: differentiate $J(\theta)$ w.r.t. θ and perform stochastic gradient ascent
  - Do this in a way that is model-free and computationally tractable



Policy Gradients

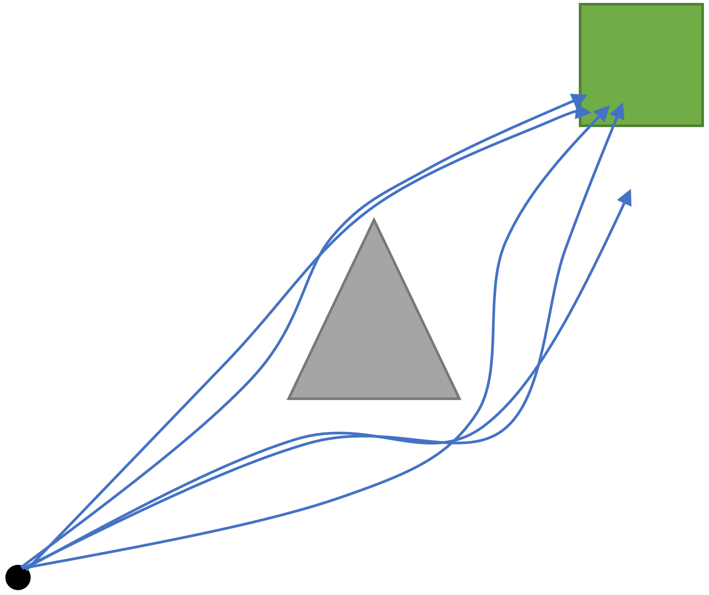
- Strategy: differentiate $J(\theta)$ w.r.t. θ and perform stochastic gradient ascent
  - Do this in a way that is model-free and computationally tractable



Policy Gradients

- Strategy: differentiate $J(\theta)$ w.r.t. θ and perform stochastic gradient ascent
  - Do this in a way that is model-free and computationally tractable

- To achieve this
  - Write out $J(\theta)$
  - Take gradient
  - Do a math trick
  - Obtain gradient expression that can be estimated easily



Write Out $J(\theta)$ and Take Gradient

- $J(\theta) \coloneqq \mathbb{E}_{\tau \sim p(\tau;\theta)}[R(\tau)]$

- $J(\theta) = \int_\tau R(\tau) p(\tau;\theta) d\tau$

- $\nabla_\theta J(\theta) = \int_\tau R(\tau) \nabla_\theta p(\tau;\theta) d\tau$
  - Hard…

Log Gradient Trick

- $\nabla_\theta J(\theta) = \int_\tau R(\tau)\nabla_\theta p(\tau;\theta)d\tau$

- Trick:
  - $\nabla_\theta p(\tau;\theta) = p(\tau;\theta)\frac{\nabla_\theta p(\tau;\theta)}{p(\tau;\theta)} = p(\tau;\theta)\nabla_\theta \log p(\tau;\theta)$
  - $\nabla_\theta J(\theta) = \int_\tau R(\tau)p(\tau;\theta)\nabla_\theta \log p(\tau;\theta)\, d\tau$
  - $\nabla_\theta J(\theta) = \mathbb{E}_{\tau\sim p(\tau;\theta)}[R(\tau)\nabla_\theta \log p(\tau;\theta)]$
  - Gradient is an expectation – can estimate this using techniques we learned before!

Model-Free Estimate of Gradient

- $\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim p(\tau;\theta)}[R(\tau)\nabla_\theta \log p(\tau;\theta)]$

- $p(\tau;\theta) = \prod_{t\geq 0} p(s_{t+1}|s_t, a_t)\pi_\theta(a_t|s_t)$
- $\log p(\tau;\theta) = \sum_{t\geq 0}[\log p(s_{t+1}|s_t, a_t) + \log \pi_\theta(a_t|s_t)]$
- $\nabla_\theta \log p(\tau;\theta) = \sum_{t\geq 0} \nabla_\theta \log \pi_\theta(a_t|s_t)$
  - Amazingly, model-free
  - Markov property is not used
  - $\nabla_\theta \log \pi_\theta(a_t|s_t)$ is known: since the form of π_θ is known
    - Eg. Backprop if π_θ is a neural network

Monte-Carlo Gradient Estimate

- Results so far:
  - $\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim p(\tau;\theta)}[R(\tau)\nabla_\theta \log p(\tau;\theta)]$
  - $\nabla_\theta \log p(\tau;\theta) = \sum_{t\geq 0} \nabla_\theta \log \pi_\theta(a_t|s_t)$

- Some more algebra to write out gradient of $\nabla_\theta J(\theta)$
  - $\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim p(\tau;\theta)}[R(\tau)\sum_{t\geq 0} \nabla_\theta \log \pi_\theta(a_t|s_t)]$
  - $\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim p(\tau;\theta)}[\sum_{t\geq 0} R(\tau)\nabla_\theta \log \pi_\theta(a_t|s_t)]$
  - $\nabla_\theta J(\theta) \approx \frac{1}{N}\sum_{i=1}^{N}\left[\sum_{t\geq 0} R(\tau_i)\nabla_\theta \log \pi_\theta\left(a_{t,i}|s_{t,i}\right)\right]$

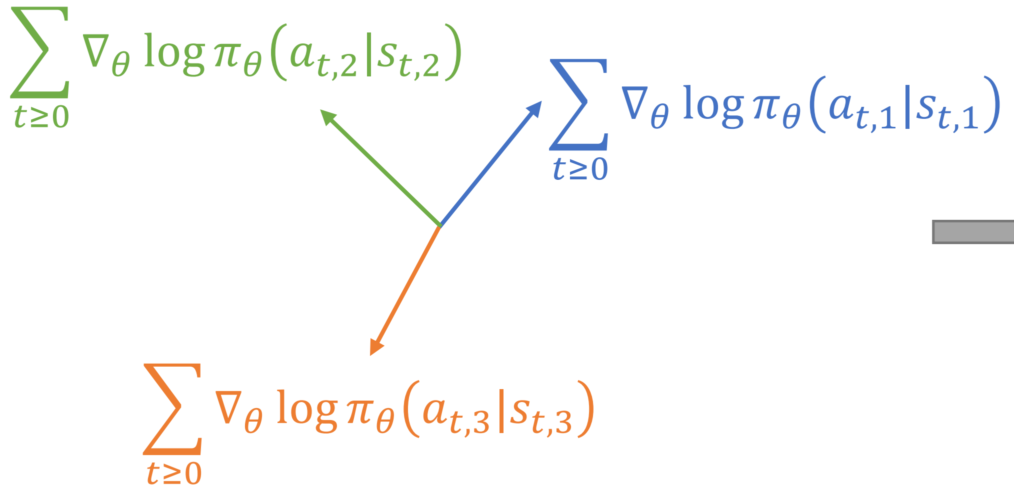
REINFORCE Algorithm

- (Monte-Carlo Policy Gradient)
- Use policy $\pi_\theta(a|s)$ to obtain a trajectory $\tau = \{s_0, a_0, \dots\}$
- Estimate the gradient of the reward
  - $\nabla_\theta J(\theta) \approx \sum_{i=1}^{N}\left[\sum_{t\geq 0} R(\tau_i)\nabla_\theta \log \pi_\theta\left(a_{t,i}|s_{t,i}\right)\right]$
- Update policy parameters via (stochastic) gradient ascent
  - $\theta \leftarrow \theta + \alpha\, \nabla_\theta J(\theta)$

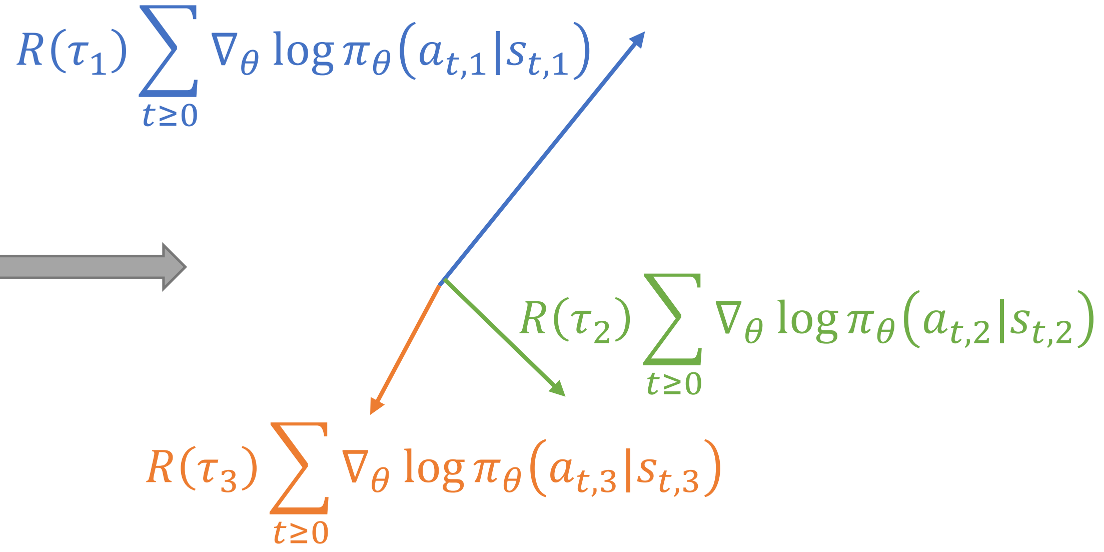
Observation 1

- Gradient estimate:
  - $\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim p(\tau;\theta)}\left[\sum_{t\geq 0} R(\tau)\nabla_\theta \log \pi_\theta(a_t|s_t)\right]$
  - $\nabla_\theta J(\theta) \approx \sum_{i=1}^{N}\left[\sum_{t\geq 0} R(\tau_i)\nabla_\theta \log \pi_\theta\left(a_{t,i}|s_{t,i}\right)\right]$

- Gradient estimate also works for POMDPs without modification



Suppose
- $R(\tau_1) = 2$
- $R(\tau_2) = -1$
- $R(\tau_3) = 1$



Observation 1

- Gradient estimate:
  - $\nabla_\theta J(\theta) \approx \mathbb{E}_{\tau \sim p(\tau;\theta)}\left[\sum_{t\geq 0} R(\tau)\nabla_\theta \log \pi_\theta(a_t|s_t)\right]$
  - $\nabla_\theta J(\theta) \approx \sum_{i=1}^{N}\left[\sum_{t\geq 0} R(\tau_i)\nabla_\theta \log \pi_\theta\left(a_{t,i}|s_{t,i}\right)\right]$

- Gradient estimate also works for POMDPs without modification

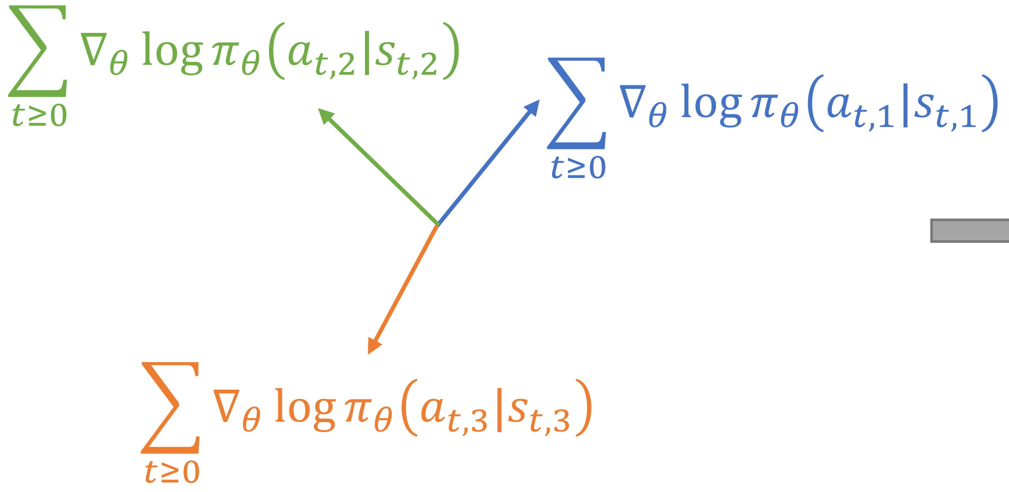
- Parameter updates: $\theta \leftarrow \theta + \alpha\, \nabla_\theta J(\theta)$
  - Trajectories have high reward will be made more likely
  - Trajectories with low reward will be made less likely
  - A high-reward trajectory has good actions… **on average**

Observation 2

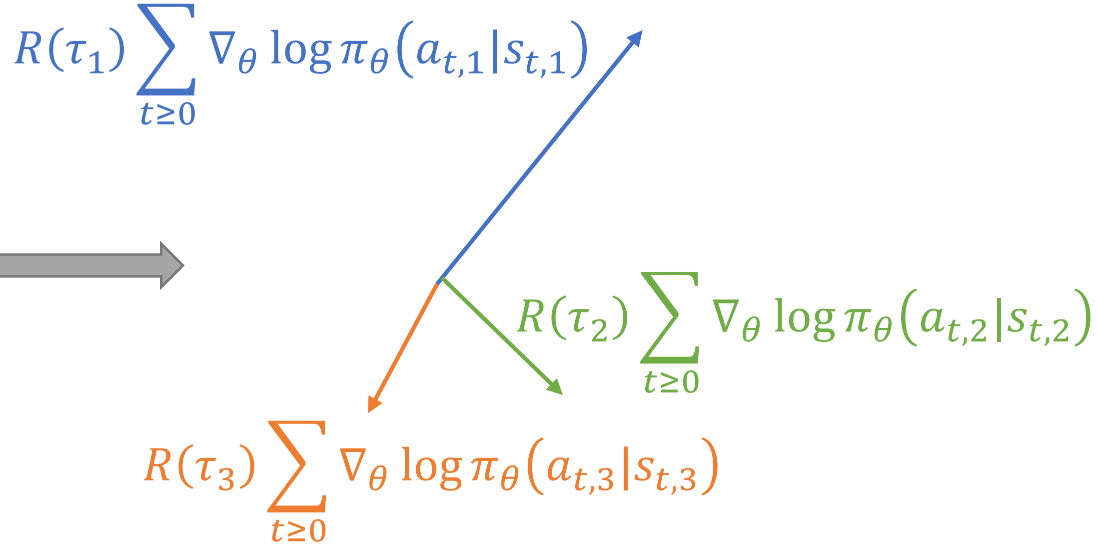
- Gradient estimate:
  - $\nabla_\theta J(\theta) \approx \mathbb{E}_{\tau \sim p(\tau;\theta)}[\sum_{t \geq 0} R(\tau) \nabla_\theta \log \pi_\theta(a_t | s_t)]$
- Causality?
  - $R(\tau)$ is the reward of the entire trajectory
  - $R(\tau)$ is multiplied in every term of the sum
  - τ includes times before t
  - So, according to the above, the weight of $\nabla_\theta \log \pi_\theta(a_t | s_t)$ depends on times prior to t?
- Simple fix:
  - $\nabla_\theta J(\theta) \approx \mathbb{E}_{\tau \sim p(\tau;\theta)}\left[\sum_{t \geq 0}\left[\left(\sum_{t' \geq t} \gamma^{t'-t} r(s_t, a_t)\right) \nabla_\theta \log \pi_\theta(a_t | s_t)\right]\right]$

Observation 3

- Gradient estimate:
  - $\nabla_\theta J(\theta) \approx \mathbb{E}_{\tau\sim p(\tau;\theta)}[\sum_{t\geq 0} R(\tau)\nabla_\theta \log \pi_\theta(a_t|s_t)]$

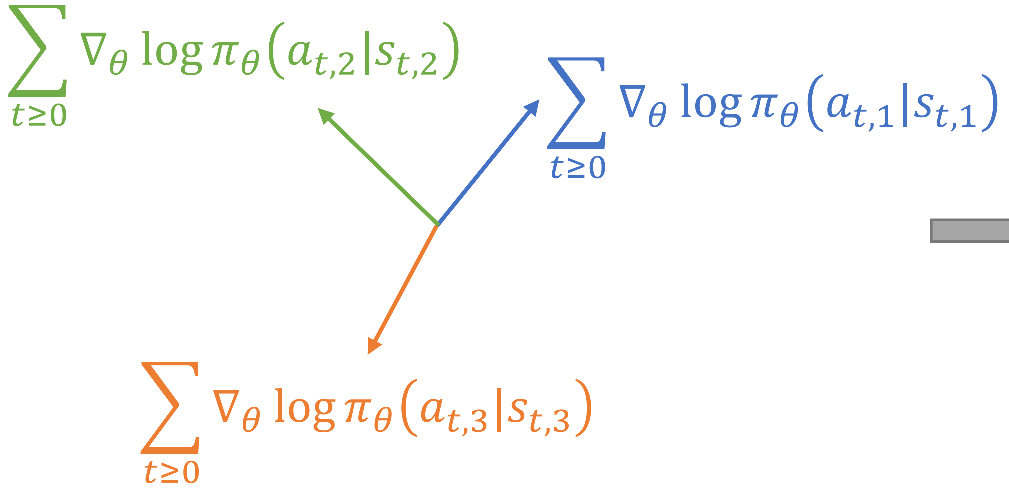


Suppose
- $R(\tau_1) = 2$
- $R(\tau_2) = -1$
- $R(\tau_3) = 1$

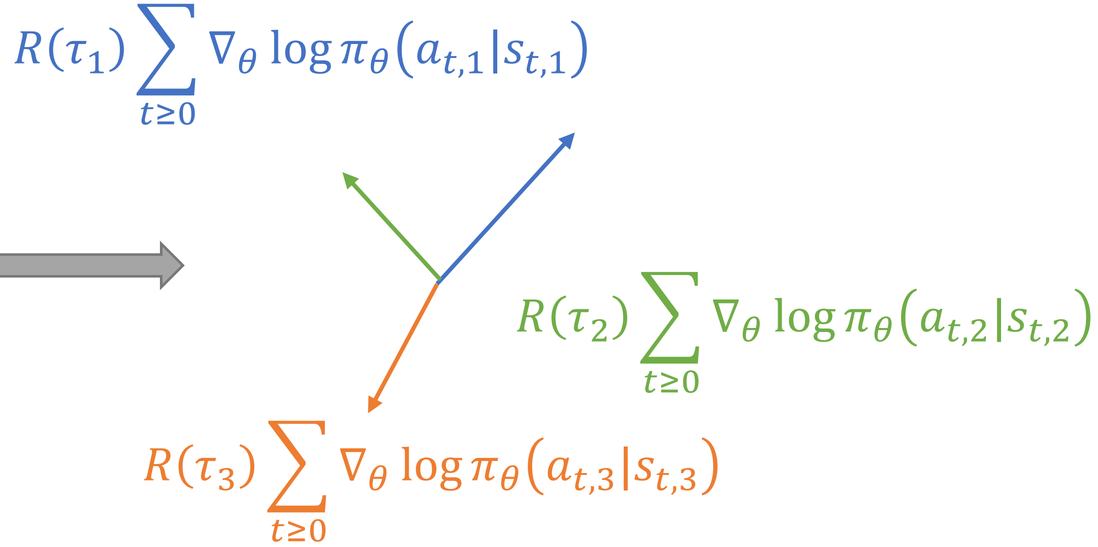


Observation 3

- Gradient estimate:
  - $\nabla_\theta J(\theta) \approx \mathbb{E}_{\tau\sim p(\tau;\theta)}[\sum_{t\geq 0} R(\tau)\nabla_\theta \log \pi_\theta(a_t|s_t)]$

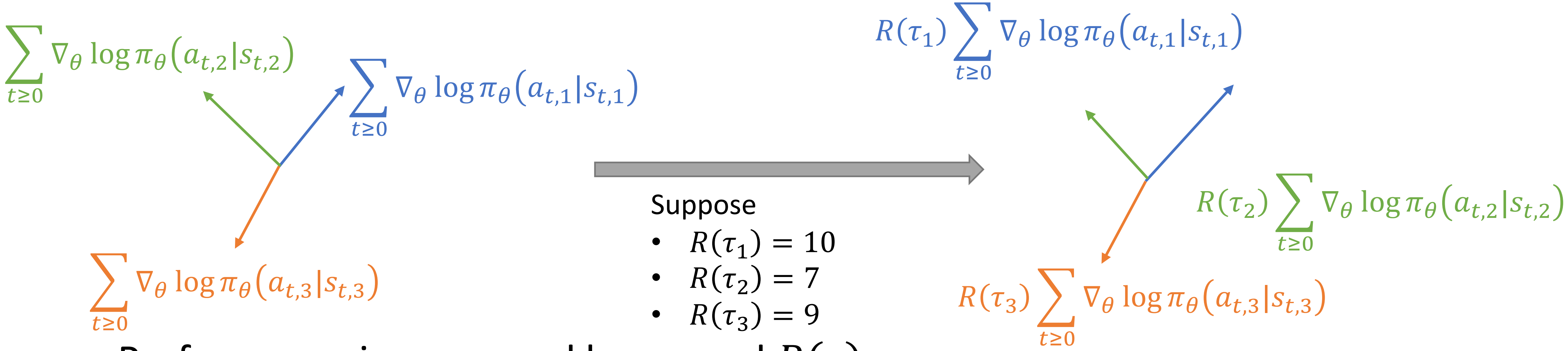


Suppose
- $R(\tau_1) = 10$
- $R(\tau_2) = 7$
- $R(\tau_3) = 9$



Observation 3

- Gradient estimate:
  - $\nabla_\theta J(\theta) \approx \mathbb{E}_{\tau \sim p(\tau;\theta)}[\sum_{t\geq 0} R(\tau) \nabla_\theta \log \pi_\theta(a_t|s_t)]$



- Performance is measured by reward $R(\tau)$
  - But what is considered "good"?
  - Need a baseline of comparison!
  - $\nabla_\theta J(\theta) \approx \mathbb{E}_{\tau \sim p(\tau;\theta)}[\sum_{t\geq 0} (R(\tau) - b) \nabla_\theta \log \pi_\theta(a_t|s_t)]$
  - Fact: expectation is unchanged as long as b does not depend on θ

Revised REINFORCE

- (Monte-Carlo Policy Gradient)
- Use policy $\pi_\theta(a|s)$ to obtain a trajectory $\tau = \{s_0, a_0, \dots\}$
- Estimate the gradient of the reward
  - $\nabla_\theta J(\theta) \approx \mathbb{E}_{\tau \sim p(\tau;\theta)}\left[\sum_{t \ge 0}\left[\left(\sum_{t' \ge t} \gamma^{t'-t} r(s_t, a_t) - b\right) \nabla_\theta \log \pi_\theta(a_t|s_t)\right]\right]$
- Update policy parameters via (stochastic) gradient ascent
  - $\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$

Picking a Baseline

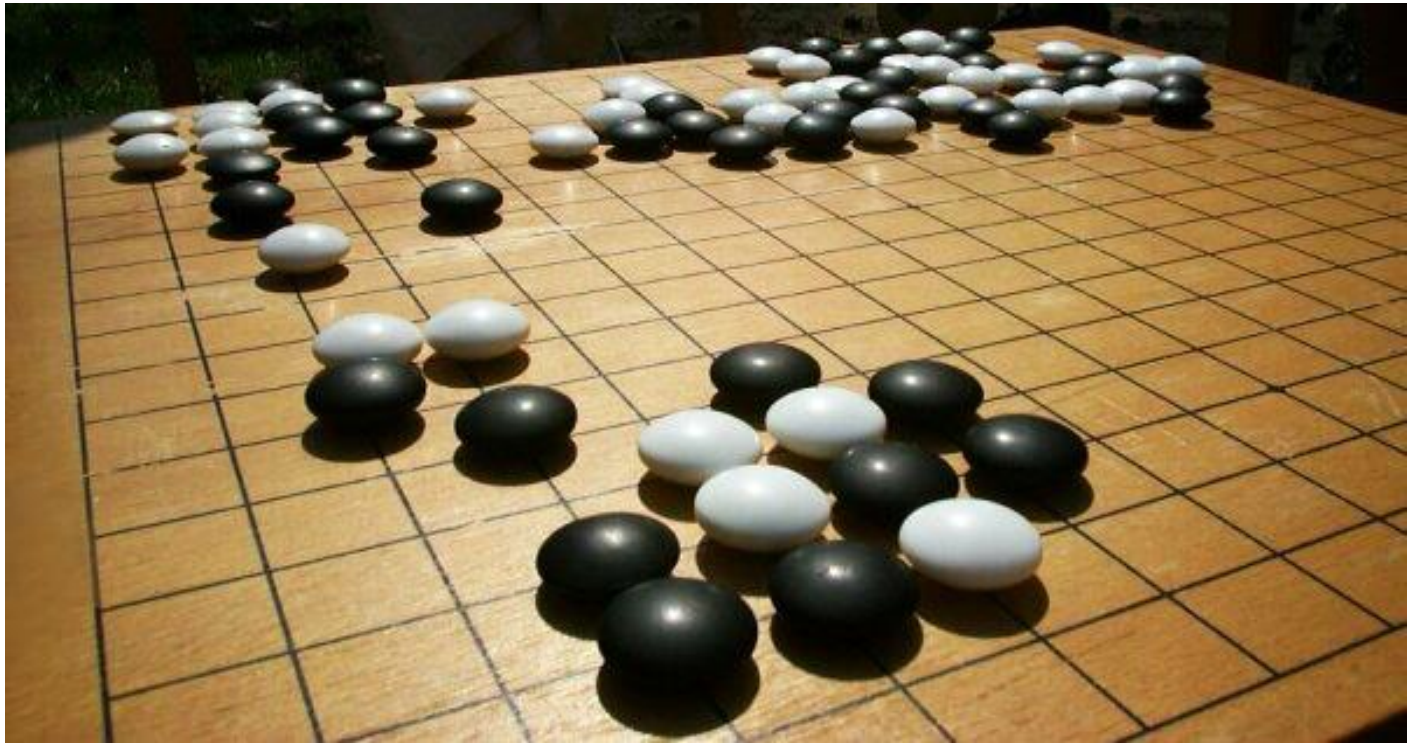
- Many choices

- Basic, intuitive choice
  - $b = \mathcal{A}_\pi(s, a) \coloneqq r(s, a) + \gamma V_\pi(s') - V_\pi(s)$
    - Good action: one that gives a return that is large relative to V
    - Bad action: one that gives a return that is small relative to V
  - $\mathcal{A}_\pi(s, a)$ -- "**advantage function**"

- But we don't know V…
  - Learn it!

Actor-Critic Methods

- Actor (policy π) decides which actions to take
- Critic (value function V) decides how good the action is



Actor-Critic Methods

- Basic algorithm, combining everything we've learned:
  1. Start with some initial policy π_θ and value function $\hat{V}(s; w)$
     - θ and w are parameters
  2. Collect data S, R, S' by executing policy
  3. Update V_ϕ: $\underset{w}{\text{minimize}} \left\| \tilde{R} + \gamma \hat{V}\left(\tilde{S}'; w^-\right) - V\left(\tilde{S}; w\right) \right\|_2^2$
     - Many methods (eg. stochastic gradient descent)
  4. Estimate policy gradient: $\nabla_\theta J(\theta) \approx \mathbb{E}_{\tau \sim p(\tau;\theta)} \left[\sum_{t \geq 0} \left(\tilde{R} + \gamma V_\pi\left(\tilde{S}'\right) - V_\pi\left(\tilde{S}\right) \right) \nabla_\theta \log \pi_\theta(a_t | s_t) \right]$
  5. Improve policy via gradient ascent: $\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$
  6. Repeat 2-5 many times

State-of-the-Art Policy Gradient Methods

- Trust region policy optimization (TRPO)
  - https://arxiv.org/abs/1502.05477

- Proximal policy optimization (PPO)
  - https://arxiv.org/abs/1707.06347

Current Robotics Research

- Additional challenge: lack of data
- Transfer learning
  - Learn in simulation, transfer knowledge to real-life
  - Build better simulators
- Curriculum learning
  - Learn easier tasks first, and increase difficulty gradually
  - Lesson plans from reachability analysis
- Reward shaping: how to design reward
  - Inverse reinforcement learning (figure out expert's reward)
  - Time-to-reach functions for simplified system, using optimal control (Xubo Lyu)

Current Robotics Research

- Transfer learning
  - Taylor, Stone. "Transfer Learning for Reinforcement *Learning* Domains: A Survey," https://dl.acm.org/citation.cfm?doid=1577069.1755839
  - Harrison et al. "ADAPT: Zero-Shot Adaptive Policy Transfer for Stochastic Dynamical Systems," https://arxiv.org/abs/1707.04674
- Curriculum learning
  - Florensa et al. "Reverse Curriculum Generation for Reinforcement Learning," http://proceedings.mlr.press/v78/florensa17a.html
  - Ivanovic et al. "BaRC: Backward Reachability Curriculum for Robotic Reinforcement Learning," https://arxiv.org/abs/1806.06161
- Reward shaping: how to design reward
  - Abbeel, Ng. "Apprenticeship learning via inverse reinforcement learning," https://dl.acm.org/citation.cfm?id=1015430
  - Time-to-reach function for simplified system, using optimal control (Xubo Lyu)