Theory and Methods for Reinforcement Learning

Prof. Volkan Cevher
volkan.cevher@epfl.ch

Lecture 7: Policy Gradient Methods I

Laboratory for Information and Inference Systems (LIONS)
École Polytechnique Fédérale de Lausanne (EPFL)

EE-618 (Spring 2020)

License Information for Reinforcement Learning Slides

Outline

- This lecture
  1. Policy gradient methods
  2. Policy gradient theorem and its proof

- Next lecture
  1. Trust Region Policy Optimization
  2. Proximal Policy Optimization

Recommended Reading

- Chapter 13 in S. Sutton, and G. Barto, *Reinforcement Learning: An Introduction*, MIT Press, 2018.
- "Policy gradient methods for reinforcement learning with function approximation", Sutton et al., NIPS, 2000

Recap

- Value-based methods:
  1. Policy evaluation: Learns estimates of either value or state-value functions

$$v_{t+1}(s) \leftarrow \sum_{a} \pi(a|s) \sum_{s',r} p(s',r|s,a)[r + \gamma v_t(s')]$$

$$q_{t+1}(a,s) \leftarrow \sum_{s',r} p(s',r|s,a) \sum_{a'} \pi(a'|s)[r + \gamma q_t(s',a')]$$

  2. Policy improvement: Design the policy based on these estimates (e.g., greedy policy, or ϵ-greedy)

$$\pi_{t+1}(s) = \arg\max_{a} \sum_{s',r} p(s',r|s,a)[r + \gamma v_\pi(s')]$$

$$\pi_{t+1}(s) = \arg\max_{a} q_\pi(a,s)$$

MC, TD, SARSA, Q-learning methods all follow this precise scheme !

Limitations of Value-Based Methods

- Do not scale to high dimensional action spaces (cannot handle continuous actions)
- Lose convergence guarantees when using function approximator of the Q function [1]
- Produce policies that are inherently deterministic

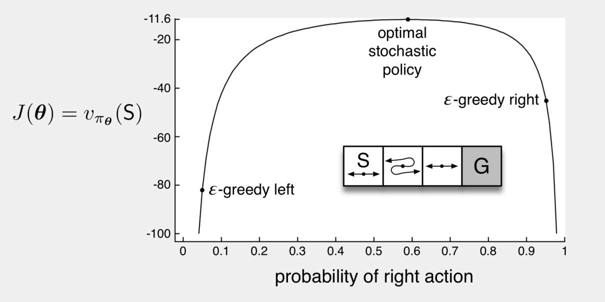


Figure: Short-corridor gridworld

Motivation

Motivation

Instead of building the policy out of a learned estimated value function, is it possible to somehow learn the policy directly?

Parameterizing the Policy

- Parameterized policy methods:

  1. **parameterize** the policy $\pi : \mathcal{S} \to \Delta(\mathcal{A}) \equiv \{p \in \mathbb{R}^{|\mathcal{A}|} : \sum_{a \in \mathcal{A}} p_a = 1\}$ using some parameter vector $\theta \in \mathbb{R}^d$ (e.g., via a neural network)
  2. Define a **performance measure** $J(\theta)$ over the policy parameters
  3. **Maximize** this function over the parameter space

Policy Gradient Methods

- General (non-convex) optimization problem:

$$\max_{\theta} J(\theta)$$

- A common method for optimizing the performance measure is the **Stochastic Gradient Method**:

$$\theta_{t+1} = \theta_t + \alpha \widehat{\nabla J(\theta_t)} \tag{1}$$

where $\widehat{\nabla J(\theta_t)}$ is an unbiased stochastic estimate of the gradient of J at θ_t.

- Methods following this general scheme are called **policy gradient methods**.

Policy Parameterization

- How to choose the policy parameterization $\pi(a|s,\theta)$?
  - Must be differentiable with respect to the components of θ
  - Always assigns non-zero probability to each action, i.e., $\pi(a|s,\theta) > 0 \; \forall a, s, \theta$
  - Can make use of domain knowledge

- Example: Soft-max in action preferences

$$\pi(a|s,\theta) = \frac{e^{\theta^T \mathbf{x}(s,a)}}{\sum_b e^{\theta^T \mathbf{x}(s,b)}} \tag{2}$$

for some feature vector $\mathbf{x}(s,a)$

- Most common parameterization: Neural networks

Performance Measure

- Assume undiscounted episodic case, each episode starting at some particular non-random state s_0 and having length T.

- Define a performance measure as

$$\begin{aligned} J(\theta) &= v_{\pi_\theta}(s_0) \\ &= \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{T-1} R_t \middle| S_0 = s_0\right] \end{aligned}$$

- v_π smooth in $\pi \Rightarrow v_{\pi_\theta}(s)$ smooth in θ (provided smooth parametrization π_θ)

Policy Gradient Theorem

- $\nabla_\theta \pi(a|s, \theta)$: easy to compute.

- What about $\nabla J(\theta)$? The effect of policy on the state distribution is generally unknown !

Policy gradient theorem (undiscounted case)

$$\nabla J(\theta) \propto \sum_s \mu(s) \sum_a q_\pi(s, a) \nabla \pi(a|s, \theta) \tag{3}$$

where μ is the state distribution under policy π, i.e., the fraction of time spent in each state normalized to sum to one.

Policy Gradient Theorem

Proof

To keep the notation simple, we omit the dependence of π on θ.

Using the expression for v_π from chapter 2, we can write for all state $s \in \mathcal{S}$:

$$
\begin{aligned}
\nabla v_\pi(s) &= \nabla \left[\sum_a \pi(a|s) q_\pi(s,a) \right] \\
&= \sum_a \left[\nabla \pi(a|s) q_\pi(s,a) + \pi(a|s) \nabla q_\pi(s,a) \right] \\
&= \sum_a \left[\nabla \pi(a|s) q_\pi(s,a) + \pi(a|s) \nabla \sum_{s',r} p(s',r|s,a)(r + v_\pi(s')) \right] \\
&= \sum_a \left[\nabla \pi(a|s) q_\pi(s,a) + \pi(a|s) \sum_{s'} p(s'|s,a) \nabla v_\pi(s') \right]
\end{aligned}
$$

Policy Gradient Theorem

Proof (cont'd)

By unrolling once the previous computation, we get

$$\nabla v_\pi(s) = \sum_a \Bigg[\nabla\pi(a|s) q_\pi(s,a) + \pi(a|s) \sum_{s'} p(s'|s,a) \sum_{a'} \Bigg[\nabla\pi(a'|s') q_\pi(s',a') + \pi(a'|s') \sum_{s''} p(s''|s',a') \nabla v_\pi(s'') \Bigg]\Bigg]$$

Similarly, by unrolling it infinitely many times, we obtain

$$\nabla v_\pi(s) = \sum_{x\in\mathcal{S}} \sum_{k=0}^{\infty} Pr(s \to x, k, \pi) \sum_a \nabla\pi(a|x) q_\pi(x,a),$$

where $Pr(s \to x, k, \pi)$ is the probability of transitioning from state s to state x in exactly k steps under policy π.

Policy Gradient Theorem

Proof (cont'd)

By definition of $J(\theta) = v_\pi(s_0)$, it is then immediate that

$$\nabla J(\theta) = \sum_s \left(\sum_{k=0}^{\infty} Pr(s_0 \to s, k, \pi) \right) \sum_a \nabla \pi(a|s) q_\pi(s,a)$$
$$\propto \sum_s \mu(s) \sum_a \nabla \pi(a|s) q_\pi(s,a),$$

where the proportionality constant is the average length of the episode.

Computing a Stochastic Estimate $\widehat{\nabla J(\theta)}$

- $\nabla J(\theta) \propto \sum_s \mu(s) \sum_a q_\pi(s,a) \nabla \pi(a|s,\theta) \rightarrow$ computationally intractable !

- Rewrite the full gradient expression as follows:

$$\begin{aligned}
\nabla J(\theta) &\propto \mathbb{E}_{S_t \sim \pi} \left[\sum_a q_\pi(S_t, a) \nabla \pi(a|S_t,\theta) \right] \\
&= \mathbb{E}_{S_t, A_t \sim \pi} \left[q_\pi(S_t, A_t) \frac{\nabla \pi(A_t|S_t,\theta)}{\pi(A_t|S_t,\theta)} \right] \text{ where } A_t \sim \pi(\cdot|S_t) \\
&= \mathbb{E}_{S_t, A_t \sim \pi, R_t} \left[G_t \frac{\nabla \pi(A_t|S_t,\theta)}{\pi(A_t|S_t,\theta)} \right] \text{ where } G_t = \sum_{t'=t+1}^{T} R_{t'}
\end{aligned}$$

- We can thus define an unbiased estimate for $\nabla J(\theta_t)$ as:

$$\widehat{\nabla J(\theta)} \propto G_t \frac{\nabla \pi(A_t|S_t,\theta)}{\pi(A_t|S_t,\theta)} \text{ where } A_t \sim \pi(\cdot|S_t) \tag{4}$$

REINFORCE: Monte Carlo Policy Gradient [2]

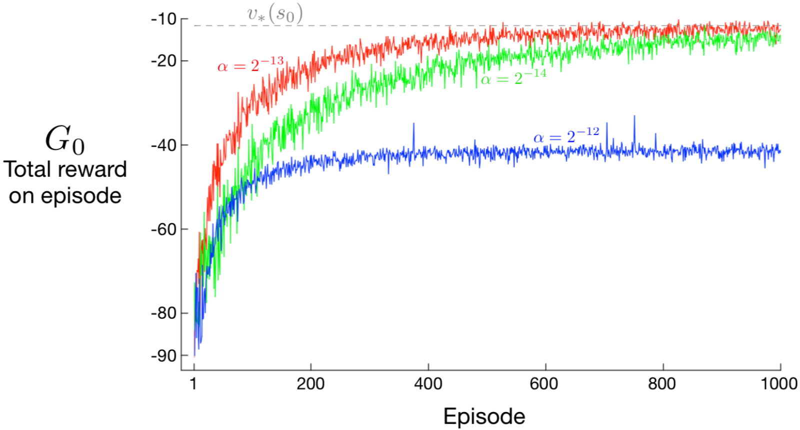
REINFORCE: Monte-Carlo Policy-Gradient Control (episodic) for π^*

Algorithm parameter: step size $\alpha > 0$.

Initialize policy parameter $\theta \in \mathbb{R}^d$ (e.g. to **0**))
for each episode **do**
  Generate an episode $S_0, A_0, R_1, ..., S_{T-1}, A_{T-1}, R_T$ following $\pi(.|., \theta)$
  for each step of the episode t=0,1,...,T-1 **do**
    $G \leftarrow \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k$
    $\theta \leftarrow \theta + \alpha\gamma^t G \frac{\nabla\pi(A_t|S_t,\theta)}{\pi(A_t|S_t,\theta)}$
  end for
end for

• The use of stochastic gradient method ensures the convergence to a local optimum when choosing a decreasing step α_t such that $\sum_{t=0}^{\infty} \alpha_t = \infty$ and $\sum_{t=0}^{\infty} \alpha_t^2 < \infty$.

Example: Short-corridor gridworld using REINFORCE



- Small enough step size $\Rightarrow$ moves towards a local optimum
- Main drawback: The MC stochastic estimate of the gradient typically have **high variance**, leading to slow learning.

Variance Reduction: Including a Baseline

- Observe that for any function $b(s)$ over states (not depending on the actions),

$$\sum_a b(s)\nabla\pi(a|s,\theta) = b(s)\nabla\sum_a \pi(a|s,\theta) = b(s)\nabla 1 = 0$$

- Therefore, we can generalize the policy gradient theorem (3) by including the baseline $b(s)$ as follows:

$$\nabla J(\theta) \propto \sum_s \mu(s)\sum_a (q_\pi(s,a) - b(s))\nabla\pi(a|s,\theta) \tag{5}$$

without modifying its expected value. Using the same procedure as previously, this gives rise to the new update rule:

$$\theta_{t+1} = \theta_t + \alpha(G_t - b(S_t))\frac{\nabla\pi(A_t|S_t,\theta)}{\pi(A_t|S_t,\theta)}. \tag{6}$$

- Baseline leaves the expected value of the update unchanged, but can have a large effect on its variance if carefully chosen.

Baseline Choices

$$\theta_{t+1} = \theta_t + \alpha(G_t - b(S_t))\frac{\nabla\pi(A_t|S_t,\theta)}{\pi(A_t|S_t,\theta)}$$

- The baseline $b(s)$ must be chosen so as to reduce the variance of the policy gradient stochastic estimate coming from the stochasticity of the return G_t.

- A natural choice is an estimate of the state value v_π.

- Similarly as in Lecture 7, we parameterize this function as $\hat{v}(S_t, \mathbf{w})$, where **w** is a parameter vector that must be learned during the algorithm.

REINFORCE with Baseline

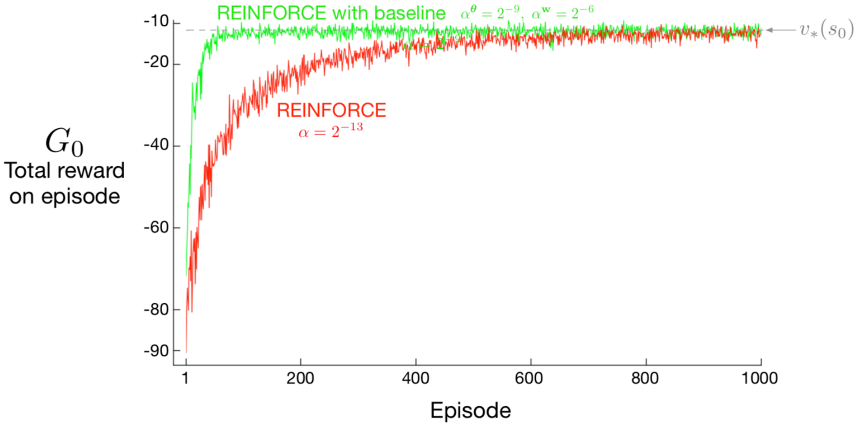
REINFORCE with baseline (episodic) for estimating $\pi_\theta = \pi_*$

Algorithm parameters: step sizes $\alpha^{\mathbf{w}} > 0, \alpha^\theta > 0$.

Initialize policy parameter $\theta \in \mathbb{R}^d$ and state-value weights $\mathbf{w} \in \mathbb{R}^d$ (e.g. to $\mathbf{0}$))
for each episode **do**
  Generate an episode $S_0, A_0, R_1, ..., S_{T-1}, A_{T-1}, R_T$ following $\pi(.|., \theta)$
  for each step of the episode t=0,1,...,T-1 **do**
    $G \leftarrow \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k$
    $\delta \leftarrow G - \hat{v}(S_t, \mathbf{w})$
    $\mathbf{w} \leftarrow \mathbf{w} + \alpha^{\mathbf{w}} \gamma^t \delta \nabla \hat{v}(S_t, \mathbf{w})$
    $\theta \leftarrow \theta + \alpha^\theta \gamma^t \delta \frac{\nabla \pi(A_t|S_t,\theta)}{\pi(A_t|S_t,\theta)}$
  end for
end for

- Rule of thumb for setting $\alpha^{\mathbf{w}}$: $\alpha^{\mathbf{w}} = \frac{0.1}{\mathbb{E}\|\nabla \hat{v}(S_t,\mathbf{w})\|_\mu^2}$ (Lecture 7).
- Drawbacks:
  - Hard to choose α^θ
  - MC method $\to$ slow
  - Inconvenient to implement online or for continuing problems

Example: Short-corridor gridworld using REINFORCE with baseline



- Baseline reduces gradient variance $\Rightarrow$ can use larger step-size $\Rightarrow$ faster convergence

Actor-Critic (AC) Methods

- Actor: refers to the policy (which decides on which action to make)

- Critic: refers to the state-value function (which influences the update rule during policy optimization)

- One-step AC method: Replace the full return of REINFORCE with the one-step return:

$$\theta_{t+1} = \theta_t + \alpha(R_{t+1} + \gamma\hat{v}(S_{t+1}, \mathbf{w}) - \hat{v}(S_t, \mathbf{w}))\frac{\nabla\pi(A_t|S_t, \theta_t)}{\pi(A_t|S_t, \theta_t)} \tag{7}$$

- AC method introduces bias, but reduces variance and accelerates learning.

- This approach can be generalized to include eligibility traces, and thus allows for flexibility in the degree of bootstrapping.

One-step Actor-Critic Algorithm

One-step Actor-Critic (episodic), for estimating $\pi_\theta = \pi_*$

Algorithm parameters: step sizes $\alpha^{\mathbf{w}} > 0, \alpha^{\theta} > 0$.

Initialize policy parameter $\theta \in \mathbb{R}^d$ and state-value weights $\mathbf{w} \in \mathbb{R}^d$ (e.g. to **0**))
for each episode **do**
  Initialize S_0 (first state of episode)
  for each step of the episode t=0,1,...,T-1 **do**
    $A_t \sim \pi_\theta(\cdot|S_t, \theta)$
    Take action A_t, observe S_{t+1}, R
    $\delta \leftarrow R + \gamma \hat{v}(S_{t+1}, \mathbf{w}) - \hat{v}(S_t, \mathbf{w})$
    $\mathbf{w} \leftarrow \mathbf{w} + \alpha^{\mathbf{w}} \gamma^t \delta \nabla \hat{v}(S_r, \mathbf{w})$
    $\theta \leftarrow \theta + \alpha^{\theta} \gamma^t \delta \frac{\nabla \pi(A_t|S_t,\theta)}{\pi(A_t|S_t,\theta)}$
  end for
end for

Actor-Critic Algorithm with Eligibility Traces

Actor-Critic with Eligibility Traces (episodic), for estimating $\pi_\theta = \pi_*$

Algorithm parameters: trace-decay rates $\lambda^{\mathbf{w}} \in [0,1], \lambda^\theta \in [0,1]$, step sizes $\alpha^{\mathbf{w}} > 0, \alpha^\theta > 0$.

Initialize policy parameter $\theta \in \mathbb{R}^d$ and state-value weights $\mathbf{w} \in \mathbb{R}^d$ (e.g. to $\mathbf{0}$))

for each episode **do**
  Initialize S_0 (first state of episode)
  $\mathbf{z}^{\mathbf{w}} \leftarrow \mathbf{0}$
  $\mathbf{z}^\theta \leftarrow \mathbf{0}$
  for each step of the episode t=0,1,...,T-1 **do**
    $A_t \sim \pi_\theta(\cdot|S_t, \theta)$
    Take action A_t, observe S_{t+1}, R
    $\delta \leftarrow R + \gamma \hat{v}(S_{t+1}, \mathbf{w}) - \hat{v}(S_t, \mathbf{w})$
    $\mathbf{z}^{\mathbf{w}} \leftarrow \gamma \lambda^{\mathbf{w}} \mathbf{z}^{\mathbf{w}} + \gamma^t \nabla \hat{v}(S, \mathbf{w})$
    $\mathbf{z}^\theta \leftarrow \gamma \lambda^\theta \mathbf{z}^\theta + \gamma^t \frac{\nabla \pi(A_t|S_t,\theta)}{\pi(A_t|S_t,\theta)}$
    $\mathbf{w} \leftarrow \mathbf{w} + \alpha^{\mathbf{w}} \delta \mathbf{z}^{\mathbf{w}}$
    $\theta \leftarrow \theta + \alpha^\theta \delta \mathbf{z}^\theta$
  end for
end for

Policy Gradients for Continuing Problems

- Similar results can be proved for the continuing case, although some quantities need to be re-defined.

- Performance measure:

$$
\begin{aligned}
J(\theta) &= \lim_{t\to\infty} \mathbb{E}[R_t | A_0, A_1, ..., A_{t-1} \sim \pi] \\
&= \sum_s \mu(s) \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a) r,
\end{aligned}
$$

where μ is the steady distribution under π, i.e.,
$\mu(s) \equiv \lim_{t\to\infty} \mathrm{Pr}(S_t = s | A_0, A_1, ..., A_{t-1} \sim \pi)$.

- State-action value function $q_\pi(s, a) = \mathbb{E}_\pi[G_t | S_t = s, A_t = a]$ defined with respect to the differential return

$$
G_t \equiv R_{t+1} - J(\pi) + R_{t+2} - J(\pi) + R_{t+3} - J(\pi) + \ldots
$$

- Using these definitions, the policy gradient theorem remains valid:

$$
\nabla J(\theta) = \sum_s \mu(s) \sum_a q_\pi(s, a) \nabla \pi(a|s, \theta).
$$

Actor-Critic algorithm with Eligibility Traces (continuing)

Actor-Critic with Eligibility Traces (continuing), for estimating $\pi_\theta = \pi_*$

Algorithm parameters: $\lambda^{\mathbf{w}} \in [0,1], \lambda^{\theta} \in [0,1], \alpha^{\mathbf{w}} > 0, \alpha^{\theta} > 0, \alpha^{\bar{R}} > 0.$

Initialize $\bar{R} \in \mathbb{R}$ (e.g. 0), policy parameter $\theta \in \mathbb{R}^d$ and state-value weights $\mathbf{w} \in \mathbb{R}^d$ (e.g. to $\mathbf{0}$))
Initialize S_0 (initial state)
$\mathbf{z}^{\mathbf{w}} \leftarrow \mathbf{0}$
$\mathbf{z}^{\theta} \leftarrow \mathbf{0}$
for each step of the episode t=0,1,... **do**
  $A_t \sim \pi_\theta(\cdot|S_t, \theta)$
  Take action A_t, observe S_{t+1}, R
  $\delta \leftarrow R - \bar{R} + \hat{v}(S_{t+1}, \mathbf{w}) - \hat{v}(S_t, \mathbf{w})$
  $\bar{R} \leftarrow \bar{R} + \alpha^{\bar{R}}\delta$
  $\mathbf{z}^{\mathbf{w}} \leftarrow \gamma\lambda^{\mathbf{w}}\mathbf{z}^{\mathbf{w}} + \nabla\hat{v}(S, \mathbf{w})$
  $\mathbf{z}^{\theta} \leftarrow \gamma\lambda^{\theta}\mathbf{z}^{\theta} + \frac{\nabla\pi(A_t|S_t,\theta)}{\pi(A_t|S_t,\theta)}$
  $\mathbf{w} \leftarrow \mathbf{w} + \alpha^{\mathbf{w}}\delta\mathbf{z}^{\mathbf{w}}$
  $\theta \leftarrow \theta + \alpha^{\theta}\delta\mathbf{z}^{\theta}$
end for

Policy Parameterization for Continuous Actions

- Continuous action space: what parametrization to use ?

- Solution 1: learn a deterministic policy $\pi : \mathcal{S} \to \mathcal{A}$ and parameterize it directly (Lecture 10).

- Solution 2: restrict the class of possible probability distributions on the action space

Example: Gaussian Parameterization for Continuous Policy

- Common choice for parameterized policy over continuous action spaces:

$$\pi(a|s,\theta_\mu,\theta_\sigma) = \frac{1}{\sigma(s,\theta_\sigma)\sqrt{2\pi}} \exp\left(-\frac{(a-\mu(s,\theta_\mu))^2}{s\sigma(s,\theta_\sigma)^2}\right),$$

where μ, σ are two differentiable function approximators, parameterized by $\theta_\mu, \theta_\sigma$ respectively.

- Using the chain rule, we can obtain the policy gradient as follows:

$$\begin{aligned}
\frac{\nabla_{\theta_\mu}\pi(a|s,\theta)}{\pi(a|s,\theta)} &= \frac{(a-\mu(s,\theta_\mu))}{\sigma(s,\theta_\sigma)^2}\nabla\mu(s,\theta_\mu) \\
\frac{\nabla_{\theta_\sigma}\pi(a|s,\theta)}{\pi(a|s,\theta)} &= \left(\frac{(a-\mu(s,\theta_\mu))^2}{\sigma(s,\theta_\sigma)^2} - 1\right)\frac{\nabla\sigma(s,\theta_\sigma)}{\sigma(s,\theta_\sigma)},
\end{aligned}$$

where $\theta = [\theta_\mu, \theta_\sigma]^T$.

Advantages of Policy Gradient Methods

- Can deal with continuous action space (very useful in robotics)

- The policy parameterization can exploit the representation power of neural network and incorporate domain knowledge

- Can exploit a large class of optimization algorithms coming from the literature

- Convergence at least to a local optimum is guaranteed

- Policy may be easier to parameterize than the value function

Disadvantages of Policy Gradient Methods

- Black-box policy
- Can be very sensitive to step-size selection
- Only converge to local optimum solution
- Harder to train off-policy
- Performance quite depends on chosen parameterization (need knowledge about the system)
- Policy may be harder to parameterize than the value function

Conclusion

- We introduced policy gradient methods by rephrasing the RL problem as a continuous optimization problem.

- We proved the policy gradient theorem, which provides a way of computing unbiased estimate of the objective function gradient.

- Orthogonal strengths-weaknesses compared to value-based methods

- This opens the box of a variety of new RL methods, by exploiting the wide literature of continuous optimization.

References

[1] Leemon Baird.
Residual algorithms: Reinforcement learning with function approximation.
In *Machine Learning Proceedings 1995*, pages 30–37. Elsevier, 1995.

[2] Ronald J Williams.
Simple statistical gradient-following algorithms for connectionist reinforcement learning.
Machine learning, 8(3-4):229–256, 1992.