Theory and Methods for Reinforcement Learning

Prof. Volkan Cevher
volkan.cevher@epfl.ch

Lecture 7: Policy Gradient 3

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

EE-618 (Spring 2022)

License Information for Theory and Methods for Reinforcement Learning (EE-618)

Recap: Policy optimization

$$\max_\theta J(\pi_\theta) := \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) | s_0 \sim \mu, \pi_\theta\right] = \mathbb{E}_{s\sim\mu}[V^{\pi_\theta}(s)]$$

Tabular parametrization

- Direct:

$$\pi_\theta(a|s) = \theta_{s,a}, \text{ with } \theta_{s,a} \geq 0, \sum_a \theta_{s,a} = 1$$

- Softmax:

$$\pi_\theta(a|s) = \frac{\exp(\theta_{s,a})}{\sum_{a' \in \mathcal{A}} \exp(\theta_{s,a'})}$$

Non-tabular parametrization

- Softmax:

$$\pi_\theta(a|s) = \frac{\exp(f_\theta(s,a))}{\sum_{a' \in \mathcal{A}} \exp(f_\theta(s,a'))}$$

- Gaussian:

$$\pi_\theta(a|s) \sim \mathcal{N}\left(\mu_\theta(s), \sigma_\theta^2(s)\right)$$

Recap: Policy gradient methods

$$\max_\theta J(\pi_\theta) := \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) | s_0 \sim \mu, \pi_\theta\right] = \mathbb{E}_{s\sim\mu}[V^{\pi_\theta}(s)]$$

Exact policy gradient method

$$\theta_{t+1} \longleftarrow \theta_t + \alpha_t \nabla_\theta J(\pi_{\theta_t}),$$

where $\nabla_\theta J(\pi_{\theta_t})$ is the full gradient of the performance objective.

Stochastic policy gradient method

$$\theta_{t+1} \longleftarrow \theta_t + \alpha_t \hat{\nabla}_\theta J(\pi_{\theta_t}),$$

where $\hat{\nabla}_\theta J(\pi_{\theta_t})$ is a stochastic estimate of the full gradient of the performance objective and is used in

- REINFORCE [9]
- REINFORCE with baseline
- Actor-Critic [5]
- ...

Previous lecture

Question 1 (Non-concavity)

When do policy gradient methods converge to an optimal solution? If so, how fast?

Question 2 (Vanishing gradient)

How to avoid vanishing gradients and further improve the convergence?

Previous lecture

Question 1 (Non-concavity)

When do policy gradient methods converge to an optimal solution? If so, how fast?

Remarks:
- Optimization wisdom: GD/SGD can converge to the global optima for "convex-like" functions:

$$J(\pi^\star) - J(\pi) = O(\|\nabla J(\pi)\|) \text{ or } O(\|G(\pi)\|)$$

- Take-away: Despite nonconcavity, PG converges to the optimal policy, in a sublinear or linear rate.

Question 2 (Vanishing gradient)

How to avoid vanishing gradients and further improve the convergence?

Previous lecture

Question 1 (Non-concavity)

When do policy gradient methods converge to an optimal solution? If so, how fast?

Remarks:
- Optimization wisdom: GD/SGD can converge to the global optima for "convex-like" functions:

$$J(\pi^\star) - J(\pi) = O(\|\nabla J(\pi)\|) \text{ or } O(\|G(\pi)\|)$$

- Take-away: Despite nonconcavity, PG converges to the optimal policy, in a sublinear or linear rate.

Question 2 (Vanishing gradient)

How to avoid vanishing gradients and further improve the convergence?

Remarks:
- Optimization wisdom: Use divergence with good curvature information.
- Take-away: Natural policy gradient achieves a faster convergence with better constants.

This lecture

Question 3 (theory)

- Why does NPG achieve a better convergence?
- How can we further improve the algorithm?

Question 4 (practice)

- How do we extend the algorithms to function approximation settings?
- How do we extend the algorithms to online settings without computing exact gradient?
- How do we extend the algorithms to off-policy settings?

Revisit gradient descent

∘ Consider the optimization problem $\min_{\mathbf{x}\in\mathbb{R}^d} \; f(\mathbf{x})$.

- Gradient descent (GD):

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \eta\nabla f(\mathbf{x}_t).$$

- Equivalent regularized form:

$$\mathbf{x}_{t+1} = \arg\min_{\mathbf{x}} \left\{\nabla f(\mathbf{x}_t)^\top(\mathbf{x}-\mathbf{x}_t) + \frac{1}{2\eta}\|\mathbf{x}-\mathbf{x}_t\|_2^2\right\}.$$

- Equivalent trust region form:

$$\mathbf{x}_{t+1} = \arg\min_{\mathbf{x}} \nabla f(\mathbf{x}_t)^\top(\mathbf{x}-\mathbf{x}_t), \text{ s.t. } \|\mathbf{x}-\mathbf{x}_t\|_2^2 \leq \delta.$$

Question: ∘ Would GD give the same trajectory under invertible linear transformations ($x \to Ay$)?

Gradient descent revisited

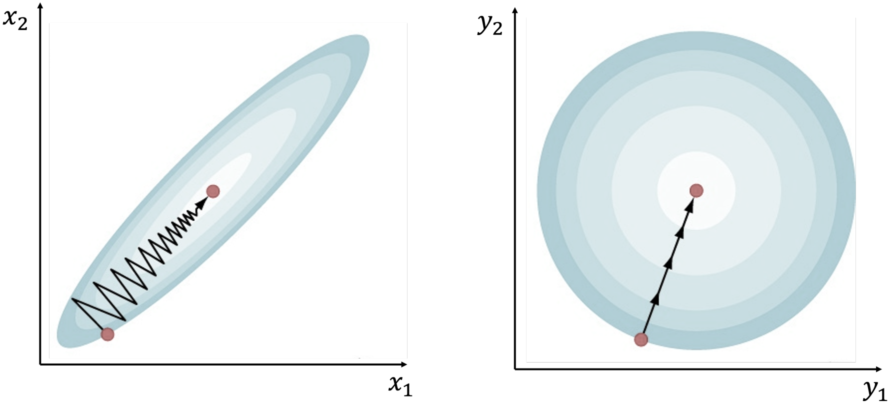


Figure: GD is not invariant w.r.t. linear transformation.

Recall Bregman divergences

Bregman divergence

Let $\omega : \mathcal{X} \to \mathbb{R}$ be continuously differentiable and 1-strongly convex w.r.t. some norm $\|\cdot\|$ on $\mathcal{X}$. The Bregman divergence D_ω associated to ω is defined as

$$D_\omega(\mathbf{x}, \mathbf{y}) = \omega(\mathbf{x}) - \omega(\mathbf{y}) - \nabla\omega(\mathbf{y})^T(\mathbf{x} - \mathbf{y}),$$

for any $\mathbf{x}, \mathbf{y} \in \mathcal{X}$.

Examples:

- Euclidean distance: $\omega(\mathbf{x}) = \frac{1}{2}\|\mathbf{x}\|_2^2$, $D_\omega(\mathbf{x}, \mathbf{y}) = \frac{1}{2}\|\mathbf{x} - \mathbf{y}\|_2^2$.
- Mahalanobis distance: $\omega(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T Q\mathbf{x}$ (where $Q \succeq I$), $D_\omega(\mathbf{x}, \mathbf{y}) = \frac{1}{2}(\mathbf{x} - \mathbf{y})^T Q(\mathbf{x} - \mathbf{y})$.
- Kullback-Leibler divergence: $\mathcal{X} = \{\mathbf{x} \in \mathbb{R}_+^d : \sum_{i=1}^d x_i = 1\}$, $\omega(\mathbf{x}) = \sum_{i=1}^d x_i \log x_i$

$$D_\omega(\mathbf{x}, \mathbf{y}) = \mathrm{KL}(\mathbf{x}\|\mathbf{y}) := \sum_{i=1}^d x_i \log \frac{x_i}{y_i}.$$

Background: Mirror descent

Mirror descent (Nemirovski & Yudin, 1983)

For a given strongly convex function ω, the iterates of mirror descent [2] are given by

$$\mathbf{x}_{t+1} = \arg\min_{\mathbf{x}\in\mathcal{X}} \{D_\omega(\mathbf{x}, \mathbf{x}_t) + \eta_t \langle \nabla f(\mathbf{x}_t), \mathbf{x} - \mathbf{x}_t \rangle\}.$$

Examples:

- Gradient descent: $\mathcal{X} \subseteq \mathbb{R}^d$, $\omega(\mathbf{x}) = \frac{1}{2}\|\mathbf{x}\|_2^2$, $D_\omega(\mathbf{x}, \mathbf{x}_t) = \frac{1}{2}\|\mathbf{x} - \mathbf{x}_t\|_2^2$.

$$\mathbf{x}_{t+1} = \Pi_{\mathcal{X}}(\mathbf{x}_t - \eta_t \nabla f(\mathbf{x}_t)).$$

- Entropic mirror descent [2]: $\mathcal{X} = \Delta_d$, $\omega(\mathbf{x}) = \sum_{i=1}^d x_i \log x_i$, $D_\omega(\mathbf{x}, \mathbf{x}_t) = \mathrm{KL}(\mathbf{x}\|\mathbf{x}_t)$.

$$\mathbf{x}_{t+1} \propto \mathbf{x}_t \odot \exp(-\eta_t \nabla f(\mathbf{x}_t)),$$

  where $\odot$ is element-wise multiplication and $\exp(\cdot)$ is applied element-wise.

- Entropic Mirror Descent attains nearly dimension-free convergence (Chapter 4 of [3]).
- See Lecture 4 Supplementary Material for more details and examples.

Background: Fisher information and KL divergence

Fisher Information Matrix

Consider a smooth parametrization of distributions $\theta \mapsto p_\theta(\cdot)$, the Fisher information matrix is defined as

$$F_\theta = \mathbb{E}_{z \sim p_\theta}[\nabla_\theta \log p_\theta(z) \nabla_\theta \log p_\theta(z)^\top].$$

Remarks:

- It is an invariant metric on the space of the parameters.
- Fisher information matrix is the Hessian of KL divergence.

$$F_{\theta_0} = \frac{\partial^2}{\partial \theta^2} \mathrm{KL}(p_{\theta_0} \| p_\theta)\Big|_{\theta=\theta_0}.$$

- The second-order Taylor expansion of KL divergence is given by

$$\mathrm{KL}(p_{\theta_0} \| p_\theta) \approx \frac{1}{2}(\theta - \theta_0)^\top F_{\theta_0}(\theta - \theta_0).$$

Background: Natural gradient descent

- Consider the optimization problem $\min_{\mathbf{x}\in\Delta}\ f(\mathbf{x})$ and represent $\mathbf{x}$ by $p_\theta(\cdot)$.
  - Natural gradient descent (Amari, 1998):
  $$\theta_{t+1} = \theta_t - \eta(F_{\theta_t})^{\dagger}\nabla f(\theta_t).$$
  - Equivalent regularized form:
  $$\theta_{t+1} = \arg\min_{\theta}\left\{\nabla f(\theta_t)^\top(\theta-\theta_t) + \frac{1}{2\eta}(\theta-\theta_t)^\top F_{\theta_t}(\theta-\theta_t)\right\}.$$
  - Equivalent trust region form:
  $$\theta_{t+1} = \arg\min_{\theta}\nabla f(\theta_t)^\top(\theta-\theta_t),\ \text{s.t.}\ \frac{1}{2}(\theta-\theta_t)^\top F_{\theta_t}(\theta-\theta_t) \le \delta.$$

Natural policy gradient method for policy optimization

$$\max_{\theta} J(\pi_\theta) := \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) | s_0 \sim \mu, \pi_\theta\right] = \mathbb{E}_{s\sim\mu}[V^{\pi_\theta}(s)]$$

Natural policy gradient (Kakade, 2002)[4]

For a stepsize $\eta > 0$, the iterates of natural policy gradient are given by

$$\theta_{t+1} = \theta_t + \eta (F_{\theta_t})^{\dagger} \nabla_\theta J(\pi_{\theta_t}).$$

Remarks:

- F_θ is the Fisher Information Matrix:

$$F_\theta = \mathbb{E}_{s\sim\lambda_\mu^{\pi_\theta}, a\sim\pi_\theta(\cdot|s)}\left[\nabla_\theta \log \pi_\theta(a|s) \nabla_\theta \log \pi_\theta(a|s)^\top\right].$$

- $\nabla_\theta J(\pi_\theta)$ is the policy gradient:

$$\nabla_\theta J(\pi_\theta) = \frac{1}{1-\gamma} \mathbb{E}_{s\sim\lambda_\mu^{\pi_\theta}, a\sim\pi_\theta(\cdot|s)}\left[\nabla_\theta \log \pi_\theta(a|s) A^{\pi_\theta}(s,a)\right].$$

- $C^\dagger$ is the Moore–Penrose inverse of the matrix C.

Interpretation of NPG

◦ NPG can be viewed as repeatedly solving the quadratic approximation of the subproblem:

$$\theta_{t+1} \approx \arg\max_{\theta} \left\{ J(\pi_\theta), \text{ s.t. } \mathrm{KL}\left(p_{\theta_t}(\tau) \| p_\theta(\tau)\right) \leq \delta \right\},$$

where $p_\theta(\tau)$ is the probability of the random trajectory $\tau = (s_0, a_0, r_1, \ldots, ...)$.

Explanation: ◦ Approximate the objective with the first-order Taylor expansion: $\nabla J(\pi_{\theta_t})^\top (\theta - \theta_t)$.

◦ Approximate the constraint with the second-order Taylor expansion:

$$\mathrm{KL}\left(p_{\theta_t}(\tau) \| p_\theta(\tau)\right) \leq \delta \quad \rightarrow \quad \frac{1}{2}(\theta - \theta_t)^\top F_{\theta_t}(\theta - \theta_t) \leq \delta.$$

Question: ◦ How can we compute the iterates of natural policy gradient efficiently?

Computing natural policy gradient

Equivalent form of NPG (Appendix C.3 [1])

Let $w^\star(\theta)$ be such that

$$(1-\gamma)(F_\theta)^\dagger \nabla_\theta J(\pi_\theta) = w^\star(\theta).$$

Then, $w^\star(\theta)$ is the solution to the following least squares minimization problem:

$$w^\star(\theta) \in \arg\min_{w} \mathbb{E}_{s\sim\lambda_\mu^{\pi_\theta}, a\sim\pi_\theta(\cdot|s)} \left[\left(w^\top \nabla_\theta \log \pi_\theta(a|s) - A^{\pi_\theta}(s,a) \right)^2 \right]. \tag{1}$$

Remarks:

- The proof follows immediately by first-order optimality condition.
- Equivalently, we can rewrite NPG as:

$$\theta_{t+1} = \theta_t + \frac{\eta}{1-\gamma} w^\star(\theta_t).$$

- $w^\star(\theta_t)$ can be obtained by solving (1) via conjugate gradients, SGD, and other solvers.

Side story

Compatible function approximation (Sutton et al., 1999)[8]

Let $A_{w^\star}(s,a)$ be defined as w

$$A_{w^\star}(s,a) := w^\star \cdot \nabla_\theta \log \pi_\theta(a|s)$$

where $w^\star$ is as defined in (1). Then we have

$$\nabla_\theta J(\pi_\theta) = \frac{1}{1-\gamma} F_\theta \cdot w^\star = \frac{1}{1-\gamma} \mathbb{E}_{s\sim\lambda_\mu^{\pi_\theta}, a\sim\pi_\theta(\cdot|s)} \left[\nabla_\theta \log \pi_\theta(a|s) A_{w^\star}(s,a)\right].$$

Remarks:

- One can obtain unbiased policy gradient with $A_{w^\star}(s,a)$
  - This is the best linear approximation of $A^{\pi_\theta}(s,a)$ using feature maps $\nabla \log \pi_\theta(s,a)$.
- Advantage value function approximation $A^{\pi_\theta}(s,a) \approx w^\top \phi(s,a)$ can introduce bias.

Example 1: Tabular NPG under softmax parameterization

NPG parameter update

Consider the softmax parameterization $\pi_\theta(a|s) = \frac{\exp(\theta_{s,a})}{\sum_{a'} \exp(\theta_{s,a'})}$ and denote $\pi_t = \pi_{\theta_t}$, the induced NPG parameter update corresponds to the following

$$\theta_{t+1} = \theta_t + \frac{\eta}{1-\gamma} A^{\pi_t}.$$

NPG policy update = policy mirror descent

In policy space, the induced update corresponds to the following

$$\pi_{t+1}(a|s) = \pi_t(a|s) \frac{\exp(\eta A^{\pi_t}(s,a)/(1-\gamma))}{Z_t(s)}.$$

Example 2: NPG with linear function approximation

NPG parameter update

Consider $\pi_\theta(a|s) = \frac{\exp(\theta^\top \phi(s,a))}{\sum_{a'} \exp(\theta^\top \phi(s,a'))}$ and denote $\pi_t = \pi_{\theta_t}$. The induced NPG parameter update corresponds to the following

$$\theta_{t+1} = \theta_t + \frac{\eta}{1-\gamma} w_t, \text{ where } w_t = \arg\min_w \mathbb{E}_{s \sim \lambda_\mu^{\pi_\theta}, a \sim \pi_\theta(\cdot|s)} \left[\left(w^\top \bar{\phi}(s,a) - A^{\pi_\theta}(s,a) \right)^2 \right].$$

NPG policy update = policy mirror descent

Notice that the parameterizations can be chosen to result in the familiar mirror descent updates on policies:

$$\pi_{t+1}(a|s) = \pi_t(a|s) \frac{\exp(\eta w_t^\top \phi(s,a)/(1-\gamma))}{Z_t(s)}$$

Convergence of tabular NPG with softmax parametrization

NPG policy update = policy mirror descent

$$\pi_{t+1}(a|s) = \pi_t(a|s)\frac{\exp(\eta A^{\pi_t}(s,a)/(1-\gamma))}{Z_t(s)}$$

Convergence of tabular NPG [1]

In the tabular setting, for any $\eta \geq (1-\gamma)^2 \log|\mathcal{A}|$ and $T > 0$, the tabular NPG satisfies

$$J(\pi^\star) - J(\pi_T) \leq \frac{2}{(1-\gamma)^2 T}.$$

Remarks:

- Nearly dimension-free convergence, no dependence on $|\mathcal{A}|, |\mathcal{S}|$.
- No dependence on distribution mismatch coefficient.

Question:

- What is the computational cost of this (nearly) dimension-free method?

Proof of tabular NPG convergence

Lemma (Policy Improvement)

$$J(\pi) - J(\pi_t) = \frac{1}{\eta}\mathbb{E}_{s\sim\lambda_\mu^\pi}\left[KL(\pi(\cdot|s)\|\pi_t(\cdot|s)) - KL(\pi(\cdot|s)\|\pi_{t+1}(\cdot|s)) + \log Z_t(s)\right].$$

Proof sketch:

- Recall from Performance Difference Lemma:

$$J(\pi) - J(\pi_t) = \frac{1}{1-\gamma}\mathbb{E}_{s\sim\lambda_\mu^\pi, a\sim\pi(a|s)}[A^{\pi_t}(s,a)].$$

- From the update rule $\pi_{t+1}(a|s) = \pi_t(a|s)\frac{\exp(\eta A^{\pi_t}(s,a)/(1-\gamma))}{Z_s}$, we have

$$A^{\pi_t}(s,a) = \frac{1-\gamma}{\eta}\log\frac{\pi_{t+1}(a|s)Z_t(s)}{\pi_t(a|s)}.$$

- Combing these two equations, we have the above lemma.

Proof of Tabular NPG convergence

Proof.

- Setting $\pi = \pi^\star$ in the previous lemma and telescoping from $t = 0, \ldots, T-1$

$$\frac{1}{T}\sum_{t=0}^{T-1} J(\pi^\star) - J(\pi_t) \le \frac{1}{\eta T}\mathbb{E}_{s\sim\lambda_\mu^{\pi^\star}}\left[\mathrm{KL}(\pi^\star(\cdot|s)\|\pi_0(\cdot|s))\right] + \frac{1}{\eta T}\sum_{t=0}^{T}\mathbb{E}_{s\sim\lambda_\mu^{\pi^\star}}\left[\log Z_t(s)\right].$$

- Setting $\pi = \pi_{t+1}$ in the previous lemma, we have

$$J(\pi_{t+1}) - J(\pi_t) \ge \frac{1}{\eta}\mathbb{E}_{s\sim\lambda_\mu^{\pi_{t+1}}}\left[\log Z_t(s)\right] \ge \frac{1-\gamma}{\eta}\mathbb{E}_{s\sim\mu}\left[\log Z_t(s)\right] \ge 0, \forall \mu.$$

- Combining these two equations and the fact that $J(\pi) \ge \frac{1}{1-\gamma}$ implies that

$$\frac{1}{T}\sum_{t=0}^{T-1} J(\pi^\star) - J(\pi_t) \le \frac{\log|\mathcal{A}|}{\eta T} + \frac{1}{(1-\gamma)^2 T}.$$

$\square$

Sample-based NPG

Sample-based NPG (informal)

- Use N-step SGD to estimate $w_t \approx w^\star(\theta_t)$
- Update $\theta_{t+1} = \theta_t + \frac{\eta}{1-\gamma} w_t$

Sample-based NPG

Initialize policy parameter $\theta_0 \in \mathbb{R}^d$, step size $\eta > 0$, $\alpha > 0$
for $t = 0, 1, \ldots, T-1$ **do**
  Initialize w_0, denote $\pi_t = \pi_{\theta_t}$
  for $n = 0, 1, \ldots, N-1$ **do**
    Obtain sample $s \sim \lambda_\mu^{\pi_t}$, $a \sim \pi_t(\cdot|s)$
    Obtain an unbiased estimate $\hat{A}(s,a)$ for $A^{\pi_t}(s,a)$
    Update w: $w \leftarrow w - \alpha(w^\top \nabla_\theta \log \pi_t(a|s) - \hat{A}(s,a)) \cdot \nabla_\theta \log \pi_t(a|s)$
  end for
  Set $w_t = w$ (or the average)
  Update $\theta_{t+1} = \theta_t + \frac{\eta}{1-\gamma} w_t$
end for

Convergence of sample-based NPG with function approximation

Convergence of sampled-based NPG (informal)

$$\mathbb{E}\left[\min_{t<T} J(\pi_{\theta_\star}) - J(\pi_{\theta_t})\right] \leq O\left(\frac{1}{1-\gamma}\sqrt{\frac{2\log|A|}{T}} + \sqrt{\epsilon_{\text{stat}}} + \sqrt{\epsilon_{\text{bias}}}\right),$$

where ϵ_{stat} is how close w_t is to a $w^\star(\theta_t)$ (statistical error) and ϵ_{bias} is how good the best policy in the class is (function approximation error).

Popular Baselines

Trust Region Policy Optimization

John Schulman JOSCHU@EECS.BERKELEY.EDU
Sergey Levine SLEVINE@EECS.BERKELEY.EDU
Philipp Moritz PCMORITZ@EECS.BERKELEY.EDU
Michael Jordan JORDAN@CS.BERKELEY.EDU
Pieter Abbeel PABBEEL@CS.BERKELEY.EDU
University of California, Berkeley, Department of Electrical Engineering and Computer Sciences

TRPO (ICML, 2015)

Proximal Policy Optimization Algorithms

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, Oleg Klimov
OpenAI
`{joschu, filip, prafulla, alec, oleg}@openai.com`

PPO (arXiv, 2017)

OpenAI implementation: `https://github.com/openai/baselines`

Trust Region Policy Optimization (TRPO)

TRPO (key idea) [6]

$$\begin{aligned} \max_{\theta} \quad & \mathbb{E}_{s \sim \lambda_{\mu}^{\pi_{\theta_t}}, a \sim \pi_{\theta_t}(\cdot|s)} \left[\frac{\pi_{\theta}(a \mid s)}{\pi_{\theta_t}(a \mid s)} A^{\pi_{\theta_t}}(s,a) \right], \\ \text{s.t.} \quad & \mathbb{E}_{s \sim \lambda_{\mu}^{\pi_{\theta_t}}} [\mathrm{KL}(\pi_{\theta}(\cdot \mid s) \| \pi_{\theta_t}(\cdot \mid s))] \leq \delta. \end{aligned}$$

Remarks:

- The surrogate objective can be viewed as linear approximation in π of $J(\pi_\theta)$:

$$J(\pi) = J(\pi_t) + \frac{1}{1-\gamma} \mathbb{E}_{s \sim \lambda_{\mu}^{\pi}, a \sim \pi(a|s)} [A^{\pi_t}(s,a)]. \qquad \text{(PDL)}$$

- It can be approximated by a natural policy gradient step.
- Line-search can ensure performance improvement and no constraint violation.

Proximal Policy Optimization (PPO2)

PPO (key idea) [7]

$$\max_{\theta} \quad \mathbb{E}_{s' \sim \lambda_{\mu}^{\pi_{\theta_t}}, a \sim \pi_{\theta_t}(\cdot|s)} \min \left\{ \frac{\pi_\theta(a|s)}{\pi_{\theta_t}(a|s)} A^{\pi_{\theta_t}}(s,a), \mathrm{clip}\left(\frac{\pi_\theta(a|s)}{\pi_{\theta_t}(a|s)}; 1-\epsilon; 1+\epsilon \right) A^{\pi_{\theta_t}}(s,a) \right\}$$

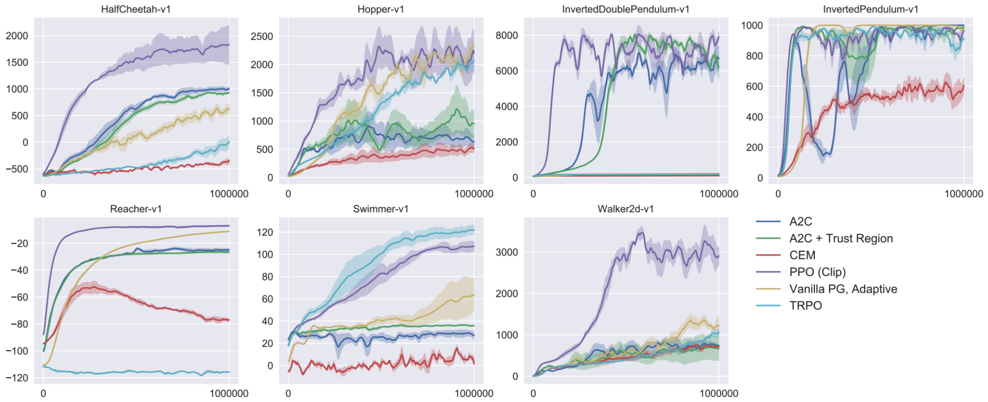
Remarks:

∘ PPO penalizes large deviation from the current policy directly inside the objective function through clipping the ratio $\frac{\pi_\theta}{\pi_{\theta_t}}$.

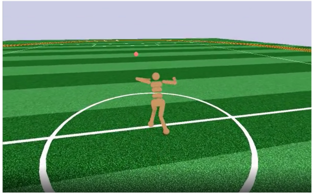
$$\mathrm{clip}(x; 1-\epsilon; 1+\epsilon) = \begin{cases} 1-\epsilon, & \text{if } x < 1-\epsilon \\ 1+\epsilon, & \text{if } x > 1+\epsilon \\ x, & \text{otherwise} \end{cases}$$

∘ Run SGD. No need to deal with the KL divergence or trust region constraints.

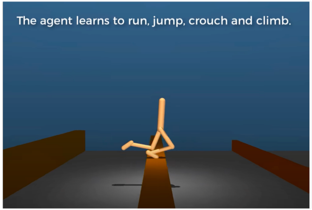
Numerical Performance [7]



More Applications



Robots



Locomotion

Muti-agent Games

Figure: PPO performs well in many locomotion task and games.

- Some links:
  - https://www.youtube.com/watch?v=hx_bgoTF7bs
  - https://openai.com/blog/openai-baselines-ppo/

Summary

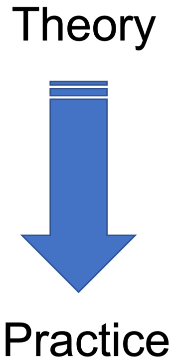
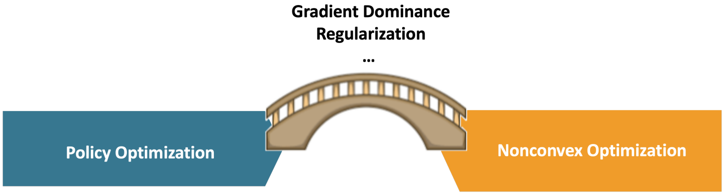


Figure from Schulman's slide on PPO in 2017.

Summary



Vanilla Policy Gradient	Gradient Descent
REINFORCE	Stochastic Gradient Descent
Natural Policy Gradient TRPO PPO	Mirror Descent
Conservative Policy Iteration ...	Frank Wolfe ...

References I

[1] Alekh Agarwal, Sham M Kakade, Jason D Lee, and Gaurav Mahajan.
Optimality and approximation with policy gradient methods in markov decision processes.
In *Conference on Learning Theory*, pages 64–66. PMLR, 2020.

[2] Amir Beck and Marc Teboulle.
Mirror descent and nonlinear projected subgradient methods for convex optimization.
Operations Research Letters, 31(3):167–175, 2003.

[3] Sébastien Bubeck.
Convex optimization: Algorithms and complexity.
Foundations and Trends in Machine Learning, 8(3-4):231–358, 2015.

[4] S. Kakade.
A natural policy gradient.
In *Advances in Neural Information Processing Systems (NeurIPS)*, 2001.

[5] Vijay R Konda and John N Tsitsiklis.
On actor-critic algorithms.
SIAM journal on Control and Optimization, 42(4):1143–1166, 2003.

[6] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz.
Trust region policy optimization.
In *International conference on machine learning*, pages 1889–1897. PMLR, 2015.

References II

[7] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov.
Proximal policy optimization algorithms.
arXiv preprint arXiv:1707.06347, 2017.

[8] Richard S Sutton, David A McAllester, Satinder P Singh, Yishay Mansour, et al.
Policy gradient methods for reinforcement learning with function approximation.
In *Conference on Neural Information Processing Systems*, pages 1057–1063, 1999.

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