NeurIPS tutorial on RL and optimization

Part 1: RL as black-box optimization

Nicolas Le Roux, Google AI - Mila

Reinforcement learning

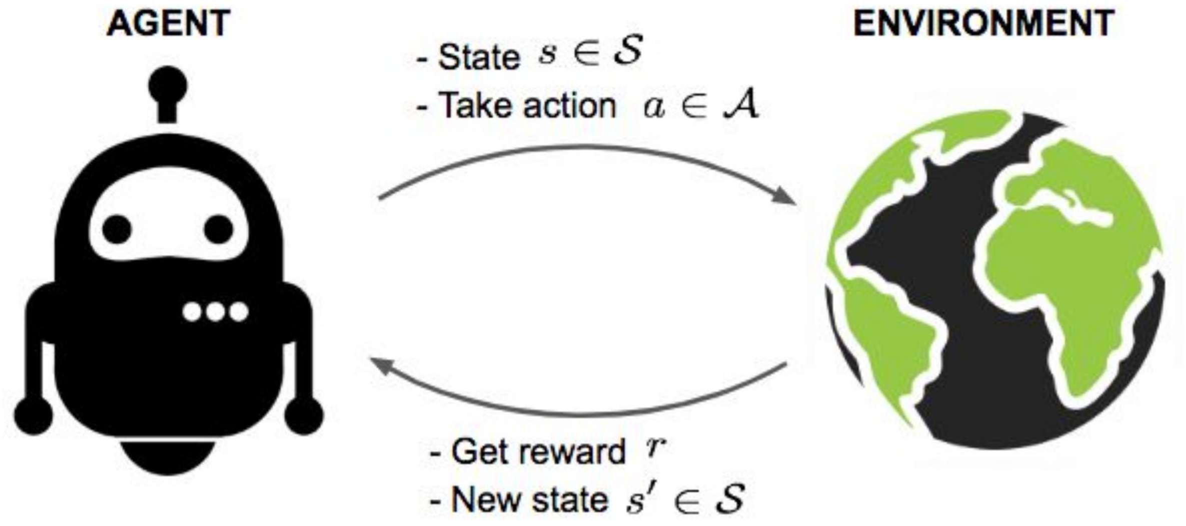


Image from Lilian Weng

Agent is governed by a policy π

The policy gradient objective

$$\tau = (s_0, a_0, s_1, a_1, \ldots, s_H)$$

$$G(\tau) = \sum_{h=0}^{H-1} \gamma^h r(s_h, a_h)$$

Q- and value functions

If I am in state s, take action a, then follow policy π, how much reward will I accumulate?

$$Q^\pi(s,a) = r(s,a) + \gamma \sum_{s',a'} P(s'|s,a)\pi(a'|s')r(s',a') + \ldots$$

$$= r(s,a) + \gamma \sum_{s',a'} P(s'|s,a)\pi(a'|s')Q^\pi(s',a')$$

$$V^\pi(s) = \sum_a \pi(a|s)Q^\pi(s,a)$$

Chapter 3.5 of Reinforcement learning, Sutton and Barto

The policy gradient objective

$$V^{\pi}(s) = \sum_{a} \pi(a|s) Q^{\pi}(s,a)$$

$$V^{\pi}(\mu) = \mathbb{E}_{s_0 \sim \mu}[V^{\pi}(s_0)]$$

- μ Is the starting state distribution
- π Is the policy being optimized

Optimization in parameter space

In practice, π is parametrized by θ and we solve

$$\begin{aligned}\theta^* &= \arg\max_\theta V^{\pi_\theta}(\mu) \\ &= \arg\max_\theta J(\theta)\end{aligned}$$

This can be solved using standard gradient ascent

$$\theta_{t+1} = \theta_t + \eta \nabla_\theta J(\theta_t)$$

Chapter 13 of Reinforcement learning, Sutton and Barto

Optimization as an iterative procedure

$$\theta_{t+1} = \theta_t + \eta \nabla_\theta J(\theta_t)$$

$$\mathbb{E}\left[\|\nabla J(\theta)\|^2\right] \leq C_1 \frac{C_{\text{curv}}}{N} + C_2 \frac{\sigma}{\sqrt{N}}$$

Curvature is the limiting factor at first, then noise kicks in

Why does curvature hurt?

$$\theta_{t+1} = \theta_t + \eta \nabla_\theta J(\theta_t)$$

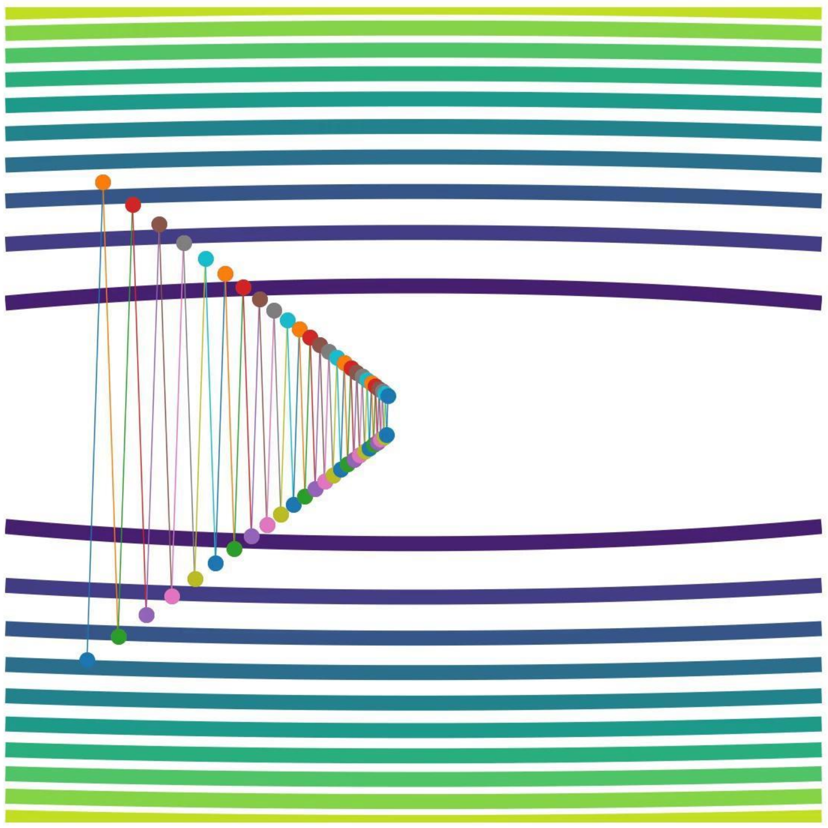
$$J(\theta) \approx J(\theta_t) + (\theta - \theta_t)^\top \nabla_\theta J(\theta_t) - \frac{1}{2\eta} \|\theta - \theta_t\|^2$$

- We make a small move because we do not trust the linear approximation
- How accurate is the approximation depends on how quickly the derivative changes

- If $\|\nabla_\theta J(\theta) - \nabla_\theta J(\theta')\| \leq L\|\theta - \theta'\|$
- Then

$$J(\theta) \geq J(\theta_t) + (\theta - \theta_t)^\top \nabla_\theta J(\theta_t) - \frac{L}{2}\|\theta - \theta_t\|^2$$

$$\theta_{t+1} = \theta_t + \frac{1}{L}\nabla_\theta J(\theta_t)$$



$$\theta_{t+1} = \arg\max_{\theta} (\theta - \theta_t)^\top \nabla_\theta J(\theta_t) - \frac{1}{2\eta} \|\theta - \theta_t\|^2$$

Parameter linearity

Divergence

Newton method

$$\theta_{t+1} = \arg\max_{\theta} (\theta - \theta_t)^\top \nabla_\theta J(\theta_t) - \frac{1}{2\eta}\|\theta - \theta_t\|^2$$

Newton method

$$\theta_{t+1} = \arg\max_{\theta} (\theta - \theta_t)^\top \nabla_\theta J(\theta_t) - \frac{1}{2\eta}(\theta - \theta_t)^\top H(\theta - \theta_t)$$

$$= \theta_t + \eta H^{-1} \nabla_\theta J(\theta_t)$$

Newton method

$$
\begin{aligned}
\theta_{t+1} &= \arg\max_{\theta} \, (\theta - \theta_t)^\top \nabla_\theta J(\theta_t) - \frac{1}{2\eta}(\theta - \theta_t)^\top H (\theta - \theta_t) \\
&= \theta_t + \eta H^{-1} \nabla_\theta J(\theta_t)
\end{aligned}
$$

Equivalent to standard gradient descent on $\theta' = H^{1/2}\theta$

Changing the divergence is equivalent to changing the representation

Mirror descent

$$\theta_{t+1} = \arg\max_{\theta} (\theta - \theta_t)^\top \nabla_\theta J(\theta_t) - \frac{1}{2\eta}\|\theta - \theta_t\|^2$$

Mirror descent

$$\theta_{t+1} = \arg\max_{\theta} (\theta - \theta_t)^\top \nabla_\theta J(\theta_t) - \frac{1}{2\eta} D_\Phi(\theta, \theta_t)$$

- D_Φ is called a *Bregman divergence*
- We want to find a divergence w/ good curvature properties

Chapter 4 of Convex optimization, Bubeck

Reasoning with policies

- $$\theta_{t+1} = \arg\max_{\theta} (\theta - \theta_t)^\top \nabla_\theta J(\theta_t) - \frac{1}{2\eta}\|\theta - \theta_t\|^2$$

Reasoning with policies

-

$$\theta_{t+1} = \arg\max_{\theta} (\theta - \theta_t)^\top \nabla_\theta J(\theta_t) - \frac{1}{2\eta} KL(P_{\theta_t} || P_\theta)$$

$$P_\theta(\tau) = \mu(s_0) \prod_{h=0}^{H} \pi_\theta(a_h | s_h) P(s_{h+1} | s_h, a_h)$$

$$\theta_{t+1} = \arg\max_{\theta} (\theta - \theta_t)^\top \nabla_\theta J(\theta_t) - \frac{1}{2\eta} KL(P_{\theta_t} || P_\theta)$$

- This problem does not have a closed form solution
- Two potential solutions:
  - Replace the KL with a quadratic approximation
  - Use a better approximation and do multiple optimization steps

$$\theta_{t+1} = \arg\max_{\theta} (\theta - \theta_t)^\top \nabla_\theta J(\theta_t) - \frac{1}{2\eta} KL(P_{\theta_t} || P_\theta)$$

- This problem does not have a closed form solution
- Two potential solutions:
  - Replace the KL with a quadratic approximation: NPG / TRPO
  - Use a better approximation and do multiple optimization steps: PPO

$$\theta_{t+1} = \arg\max_{\theta} (\theta - \theta_t)^\top \nabla_\theta J(\theta_t) - \frac{1}{2\eta} KL(P_{\theta_t} || P_\theta)$$

- Exactly minimizing the KL can be expensive

- Can we replace it with an approximation?

$$KL(P_{\theta_t} || P_\theta) = (\theta - \theta_t)^\top F(\theta_t)(\theta - \theta_t) + o(\|\theta - \theta_t\|^2)$$

- The minimization gives $\theta_{t+1} = \theta_t + \eta F(\theta_t)^{-1} \nabla_\theta J(\theta_t)$

Natural policy gradient (Kakade, 2001)

Replacing the penalty with a constraint

$$\theta_{t+1} = \arg\max_{\theta} (\theta - \theta_t)^\top \nabla_\theta J(\theta_t) - \frac{1}{2\eta} KL(P_{\theta_t} || P_\theta)$$

Replacing the penalty with a constraint

$$\theta_{t+1} = \arg\max_{\theta} (\theta - \theta_t)^\top \nabla_\theta J(\theta_t) \text{ s.t. } KL(P_{\theta_t} || P_\theta) \leq C$$

- This does not immediately give a stepsize: line search!

$$\theta_{t+1} = \theta_t + \eta_t F(\theta_t)^{-1} \nabla_\theta J(\theta_t)$$

With η_t such that the constraint is satisfied

TRPO (Schulman et al., 2015)

From "$\boldsymbol{\theta}$-linear" to "$\boldsymbol{\pi}$-linear"

$$\theta_{t+1} = \arg\max_{\theta} (\theta - \theta_t)^\top \nabla_\theta J(\theta_t) - \frac{1}{2\eta} KL(P_{\theta_t} || P_\theta)$$

If the divergence deals with policies, why have an approximation linear in θ ?

From “$\boldsymbol{\theta}$-linear” to “$\boldsymbol{\pi}$-linear”

$$J(\pi) = J(\pi_t) + \sum_{s,a} d^{\pi}(s)\pi(a|s)Q^{\pi_t}(s,a)$$

- $d^{\pi}(s)$ is the stationary distribution induced by π
- It has a complex derivative w.r.t. π

From “$\boldsymbol{\theta}$-linear” to “$\boldsymbol{\pi}$-linear”

$$J(\pi) = J(\pi_t) + \sum_{s,a} d^{\pi}(s)\pi(a|s)Q^{\pi_t}(s,a)$$

$$\approx J(\pi_t) + \sum_{s,a} d^{\pi_t}(s)\pi(a|s)Q^{\pi_t}(s,a)$$

Linear in $\boldsymbol{\pi}$!

From “$\boldsymbol{\theta}$-linear” to “$\boldsymbol{\pi}$-linear”

$$\pi_{t+1} = \arg\max_{\pi} \sum_{s,a} d^{\pi_t}(s)\pi(a|s)Q^{\pi_t}(s,a)$$

- Actions are sampled from $\pi_t(a|s)$
- We use the importance ratio $\dfrac{\pi_t(a|s)}{\pi(a|s)}$
- It is clipped to prevent poor estimation

PPO (Schulman et al., 2017)

The curse of deterministic policies

$$\mathbb{E}\left[\|\nabla J(\theta_N)\|^2\right] \leq C_1 \frac{C_{\mathbf{curv}}}{N} + C_2 \frac{\sigma}{\sqrt{N}}$$

- Signal to noise ratio of deterministic policies is very small
- Second-order methods cannot address that
- We need to add curvature near the boundaries

Adding curvature near the boundaries

- $H(P_\theta) = -\sum_\tau P_\theta(\tau) \log P_\theta(\tau)$: entropy regularization, small effect
- $KL(P_{\theta_t} || P_\theta)$: local KL, slows down movement
- $KL(P_{\theta_0} || P_\theta)$: relative entropy, acts as a log barrier, faster rates

What about variance?

- Getting the true gradient is difficult
- Stochastic estimates of the gradient

$$\mathbb{E}\left[\|\nabla J(\theta)\|^2\right] \leq C_1 \frac{C_{\mathbf{curv}}}{N} + C_2 \frac{\sigma}{\sqrt{N}}$$

Control variates

- We want to estimate $\mathbb{E}_\xi[\nabla_\theta J(\theta, \xi)]$ from samples $\nabla_\theta J(\theta, \xi_i)$
- Imagine we know $\mathbb{E}_\xi[z(\xi)]$ for some variable z
- Then $\nabla_\theta J(\theta, \xi_i) - z(\xi_i) + \mathbb{E}_\xi[z(\xi)]$ has the correct expectation
- It also has lower variance if $z(\xi_i)$ is positively correlated with $\nabla_\theta J(\theta, \xi_i)$

Example: stochastic variance reduction methods

$$\theta_{t+1} = \theta_t - \alpha \left(\nabla_\theta J(\theta, \xi_i) - g_i + \frac{1}{N} \sum_j g_j \right)$$

- g_i is the gradient for datapoint i stored in memory
- We store the last computed gradient for each datapoint in memory
- This has (roughly) the same convergence rate as a batch method

Control variates in RL: baselines

$$\nabla_\theta J(\theta) = \sum_{s,a} d^\pi(s)\pi(a|s)Q^\pi(s,a)\nabla_\theta \log \pi(a|s)$$

$$\nabla_\theta J(\theta) = \sum_{s,a} d^\pi(s)\pi(a|s)\Big(Q^\pi(s,a) - z(s)\Big)\nabla_\theta \log \pi(a|s)$$

- $z(s)$ is called a (state-dependent) baseline
- A carefully chosen baseline can reduce the variance
- Other techniques exist but they bias the gradient (more w/ Martha)

Conclusion

- Policy gradient methods can be tackled as a pure optimization problem
- One needs to deal with curvature and noise
- Curvature: careful choice of the parametrization can help
- Entropy and relative entropy also add curvature
- Noise: baselines are an easy tool but they have limited impact