Policy Search: Actor-Critic Methods

Matteo Pirotta
Facebook AI Research

Reinforcement Learning Summer School (RLSS)
I will add the parts presented on the whiteboard soon.
Value Iteration as Gradient Descent (optional)
Value Iteration

**Optimal Bellman Operator**

\[ L v(s) = \max_a \{ r(s, a) + \gamma \sum_y p(y|s, a) v(y) \} \]

**Value Iteration**

\[ v_{n+1} = L v_n \]

**Guarantees** [Puterman, 1994, Sec. 6.3.2]

greedy policy \( \pi^+(s) \in \arg \max_a \{ r(s, a) + \gamma \sum_y p(y|s, a) v_{n+1}(y) \} \)

\[ \| v_{n+1} - v_n \|_\infty \leq \frac{\epsilon (1 - \gamma)}{2\gamma} \implies \| v^{\pi^+} - v^* \| \leq \epsilon \]

thus \( \pi^+ \) is an \( \epsilon \)-optimal policy

\( \epsilon \)-optimal policy in \( O \left( \frac{1}{1 - \gamma} \log \left( \frac{1}{\epsilon (1 - \gamma)} \right) \right) \) iterations
Value Iteration

**Optimal Bellman Operator**

\[ Lu(s) = \max_{a} \{ r(s, a) + \gamma \sum_{y} p(y|s, a)v(y) \} \]

**Value Iteration**

\[ v_{n+1} = Lu \]

**Guarantees** [Puterman, 1994, Sec. 6.3.2]

Greedy policy \( \pi^+(s) \in \arg \max_{a} \{ r(s, a) + \gamma \sum_{y} p(y|s, a)v_{n+1}(y) \} \)

\[ \|v_{n+1} - v_{n}\|_{\infty} \leq \frac{\epsilon(1 - \gamma)}{2\gamma} \implies \|v^{\pi^+} - v^{*}\| \leq \epsilon \]

stopping condition

thus \( \pi^+ \) is an \( \epsilon \)-optimal policy

\( \epsilon \)-optimal policy in \( O \left( \frac{1}{1 - \gamma} \log \left( \frac{1}{\epsilon(1 - \gamma)} \right) \right) \) iterations
Relaxation Value Iteration (R-VI)

R-VI is a Krasnoselskii-Mann (KM) iteration

\[ v_{n+1} = v_n - \alpha_n (v_n - Lv_n) \]

- this is a smooth version of VI
  - \( \alpha_n = 1 \) is VI
- \( v_n - Lv_n \) is the gradient of an unknown function \( f : \mathbb{R}^n \to \mathbb{R}^n \)

why? \( \|v^* - Lv^*\|_\infty = 0 \) (vanishing gradient at the optimum)
Relaxation Value Iteration (R-VI)

R-VI is a Krasnoselskii-Mann (KM) iteration

\[ v_{n+1} = v_n - \alpha_n (v_n - L v_n) \]

- this is a smooth version of VI
  - \( \alpha_n = 1 \) is VI
- \( v_n - L v_n \) is the gradient of an unknown function \( f : \mathbb{R}^n \rightarrow \mathbb{R}^n \)
  - why? \( \| v^* - L v^* \|_\infty = 0 \) (vanishing gradient at the optimum)

**Guarantees** \( \forall \alpha_n = \alpha \in (0, 2/(1 - \gamma)) \)

\[ \| v_n - v^* \|_\infty \leq (\gamma \alpha + |1 - \alpha|)^n \cdot \| v_0 - v^* \|_\infty \]

**Optimal rate:** \( \alpha = 1 \implies \text{VI} \)

Not faster than VI but interesting connections with gradient descent
Gradient Descent

\[ v_{n+1} = v_n - \alpha_n \nabla f(v_n) \]

- Linear convergence rate when \( f \) is \( \mu \)-strongly convex and \( L \)-Lipschitz continuous \((L > \mu > 0)\)
- Optimal rate is obtained for \( \alpha_n = \alpha = \frac{2}{L + \mu} \)

\[ \exists C > 0, \quad \|v_n - v^*\|_2 \leq C \left( \frac{L - \mu}{L + \mu} \right)^n \]

Can we map \((L, \mu)\) to parameters of VI?
R-VI as Gradient Descent
[Goyal and Grand-Clement, 2019]

\[(GD) \quad \mu \|v - w\|_2 \leq \|\nabla f(v) - \nabla f(w)\|_2 \leq L\|v - w\|_2\]

\[\mu \mapsto 1 - \gamma \quad L \mapsto 1 + \gamma\]

Recall that optimal rate of R-VI is obtained for

\[\alpha = 1 = \frac{2}{(1 + \gamma) + (1 - \gamma)} = \frac{2}{L + \gamma}\]

as in gradient descent

and the optimal rate is \(\gamma\):

\[\gamma = \frac{(1 + \gamma) - (1 - \gamma)}{(1 + \gamma) + (1 - \gamma)} = \frac{L - \mu}{L + \mu}\]

Strong connection between VI and gradient (simply different norms)
R-VI as Gradient Descent

[Goyal and Grand-Clement, 2019]

\[(GD) \quad \mu \|v - w\|_2 \leq \|\nabla f(v) - \nabla f(w)\|_2 \leq L\|v - w\|_2\]

\[(VI) \quad (1 - \gamma)\|v - w\|_\infty \leq \|(v - Lv) - (w - Lw)\|_\infty \leq (1 + \gamma)\|v - w\|_\infty\]

\[\mu \mapsto 1 - \gamma \quad L \mapsto 1 + \gamma\]

Recall that optimal rate of R-VI is obtained for

\[\alpha = 1 = \frac{2}{(1 + \gamma) + (1 - \gamma)} = \frac{2}{L + \gamma}\]

as in gradient descent

and the optimal rate is \(\gamma\):

\[\gamma = \frac{(1 + \gamma) - (1 - \gamma)}{(1 + \gamma) + (1 - \gamma)} = \frac{L - \mu}{L + \mu}\]

Strong connection between VI and gradient (simply different norms)
Accelerated Value Iteration (A-VI)
[Goyal and Grand-Clement, 2019]

**Nesterov Acceleration for VI**
\[\forall v_0, v_1 \in \mathbb{R}^S, n \geq 1\]

\[h_n = v_n + \beta_n(v_n - v_{n-1})\]
\[v_{n+1} = h_n - \alpha_n(h_n - Lh_n)\]

When \(\beta_n = \gamma\) and \(\alpha_n = 1/(1 + \gamma)\)

\[\epsilon\text{-optimal policy in } O\left(\frac{\sqrt{1+\gamma}}{\sqrt{1-\gamma}} \log\left(\frac{1}{\epsilon(1-\gamma)}\right)\right)\text{ iterations}\]
From Policy Iteration to Policy Search
Policy Iteration: recap

Let $\pi_0$ be an arbitrary stationary policy

while $k = 1, \ldots, K$ do

- **Policy Evaluation:** given $\pi_k$ compute $v_k = v^{\pi_k}$

- **Policy Improvement:** find $\pi_{k+1}$ that is better than $\pi_k$
  - e.g., compute the greedy policy

\[
\pi_{k+1}(s) \in \arg \max_{a \in A} \left\{ r(s, a) + \gamma \sum_y p(y|s, a) v^{\pi_k}(y) \right\}
\]

return the last policy $\pi_K$

depth
Let $\pi_0$ be an arbitrary stationary policy

while $k = 1, \ldots, K$ do

\begin{itemize}
  \item **Policy Evaluation**: given $\pi_k$ compute $v_k = v^{\pi_k}$
  \item **Policy Improvement**: find $\pi_{k+1}$ that is better than $\pi_k$
    - e.g., compute the greedy policy
      \[ \pi_{k+1}(s) \in \arg \max_{a \in A} \left\{ r(s, a) + \gamma \sum_y p(y|s, a)v^{\pi_k}(y) \right\} \]
\end{itemize}

return the last policy $\pi_K$

end

- Convergence is finite and monotonic [Bertsekas, 2007] (in exact settings)

? **Issues**: Function approximation for $v^{\pi_k}$ $\implies$ Is it still converging? Continuous actions?
Approximate Policy Iteration

**Issue:** is no longer guaranteed to converge!

**Proposition**

The asymptotic performance of the policies $\pi_k$ generated by the API algorithm is related to the approximation error as:

$$\limsup_{k \to +\infty} \|v^* - v^{\pi_k}\|_\infty \leq \frac{2\gamma}{(1 - \gamma)^2} \limsup_{k \to +\infty} \|v_k - v^{\pi_k}\|_\infty$$

performance loss

approximation error
Approximate Policy Iteration

**Issue:** is no longer guaranteed to converge!

**Proposition**

The asymptotic performance of the policies $\pi_k$ generated by the API algorithm is related to the approximation error as:

$$\limsup_{k \to +\infty} \| v^* - v^{\pi_k} \|_\infty \leq \frac{2\gamma}{(1 - \gamma)^2} \limsup_{k \to +\infty} \| v_k - v^{\pi_k} \|_\infty$$

**Diagram:**

- **Transitional phase**
- **Stationary phase**
- **Asymptotic error**
Approximate Policy Iteration: Issues

Potential pathologies in policy-iteration with function approximation

1. Exploration
2. Policy evaluation: bias, simulation bias/error
3. Policy improvement: policy oscillation
   - *local attractors*, e.g., local maxima
Approximate Policy Iteration: Issues

Potential pathologies in policy-iteration with function approximation

1. Exploration
2. Policy evaluation: bias, simulation bias/error
3. Policy improvement: policy oscillation
   • *local attractors*, e.g., local maxima
Approximate Policy Iteration: Issues

Potential pathologies in policy-iteration with function approximation

1. Exploration
2. Policy evaluation: bias, simulation bias/error
3. Policy improvement: policy oscillation
   - *local attractors*, e.g., local maxima
Policy oscillation with linear function approximation [Koller and Parr, 2000, Lagoudakis and Parr, 2003a]

Tetris [Bertsekas and Ioffe, 1996] very pathological [e.g., Scherrer et al., 2015]

Figure 9: The problematic MDP.
Approximate a stochastic policy directly using function approximation

$$\pi_\theta : S \to \mathcal{P}(A) \text{ with } \theta \in \mathbb{R}^d$$

Let $J(\pi_\theta)$ denote the policy performance of policy $\pi_\theta$

Policy optimization problem

$$\max_{\pi_\theta} J(\pi_\theta)$$
Approximate a *stochastic policy* directly using function approximation

\[ \pi_\theta : S \rightarrow \mathcal{P}(A) \text{ with } \theta \in \mathbb{R}^d \]

Let \( J(\pi_\theta) \) denote the *policy performance* of policy \( \pi_\theta \)

Policy optimization problem

\[
\max_{\pi_\theta} J(\pi_\theta)
\]

**Solution 1: Policy Search/Black-box optimization:**
Use global optimizers or gradient by finite-difference methods

Policy \( \pi_\theta \) can also be *not differentiable* w.r.t. \( \theta \)
Approximate a *stochastic policy* directly using function approximation

\[ \pi_\theta : S \rightarrow \mathcal{P}(A) \text{ with } \theta \in \mathbb{R}^d \]

Let \( J(\pi_\theta) \) denote the *policy performance* of policy \( \pi_\theta \)

Policy optimization problem

\[ \max_{\pi_\theta} J(\pi_\theta) \]

**Solution 1: Policy Search/Black-box optimization:**

Use global optimizers or gradient by finite-difference methods

Policy \( \pi_\theta \) can also be *not differentiable* w.r.t. \( \theta \)

**Solution 2: Policy gradient optimization:**

Compute the gradient \( \nabla_\theta J(\theta) \) and follow the ascent direction

\( \nabla_\theta \pi_\theta(s, a) \) should exist
Policy Gradient as Policy Update

Approximate Policy Iteration

\[ \pi_{\theta_{k+1}} = \arg \max_{\pi_\theta} q^{\pi_\theta}(s, \pi_\theta(s)) \]

Unstable (fast)

Policy Gradient

\[ \theta_{k+1} = \theta_k + \alpha_k \nabla J(\theta_k) \]

Smooth, fine control (slow)

How do we compute \( \nabla_{\theta} J(\theta) \)?

(recap on optimality criteria)
Finite Horizon
Given an MDP \( M = (S, A, p, r, H, \rho) \) and a policy \( \pi \)

\[
J(\pi) = \mathbb{E} \left[ \sum_{t=1}^{H} r_t | \pi, M \right] = \mathbb{E}_{\tau \sim P(\tau | \pi, M)} \left[ R(\tau) \right]
\]

where \( \tau = (s_1, a_1, r_1, \ldots, s_{H+1}) \) is a trajectory and \( R(\tau) \) its return (sum of returns).
Policy Gradient: finite-horizon

**Theorem** ([Williams, 1992, Sutton et al., 2000])

For any finite-horizon MDP $M = (S, A, p, r, H, \rho)$ and differentiable policy $\pi_\theta$

$$\nabla_\theta J(\pi_\theta) = \mathbb{E}_{\tau \sim P(\cdot | \pi, M)} \left[ R(\tau) \sum_{t=1}^{H} \nabla_\theta \log \pi_\theta(s_t, a_t) \right]$$
Proof

- The objective is an *expectation*. Want to compute the gradient w.r.t. \( \theta \)

\[
\nabla_\theta J(\theta) = \nabla_\theta \mathbb{E}_\tau[R(\tau)] = \nabla_\theta \int \mathbb{P}(\tau|\theta)R(\tau)d\tau
\
= \int \nabla_\theta \mathbb{P}(\tau|\theta)R(\tau)d\tau
\
= \int \mathbb{P}(\tau|\theta) \nabla_\theta \log \mathbb{P}(\tau|\theta) R(\tau)d\tau
\
= \mathbb{E}_\tau[R(\tau)\nabla_\theta \log \mathbb{P}(\tau|\theta)]
\]

* log trick

\[
\nabla_\theta \log \mathbb{P}(\tau|\theta) = \frac{\nabla_\theta \mathbb{P}(\tau|\theta)}{\mathbb{P}(\tau|\theta)}
\]
The objective is an expectation. Want to compute the gradient w.r.t. $\theta$

$$
\nabla_\theta J(\theta) = \nabla_\theta \mathbb{E}_\tau [R(\tau)] = \nabla_\theta \int \mathbb{P}(\tau|\theta) R(\tau) d\tau

= \int \nabla_\theta \mathbb{P}(\tau|\theta) R(\tau) d\tau

= \int \mathbb{P}(\tau|\theta) \nabla_\theta \log \mathbb{P}(\tau|\theta) \cdot R(\tau) d\tau

= \mathbb{E}_\tau [R(\tau) \nabla_\theta \log \mathbb{P}(\tau|\theta)]

$$

Last expression is an unbiased gradient estimator. Just sample $\tau_i \sim \mathbb{P}(\tau|\theta)$, and compute $\hat{g}_i = R(\tau_i) \nabla_\theta \log \mathbb{P}(\tau|\theta)$
Proof

- The objective is an *expectation*. Want to compute the gradient w.r.t. $\theta$

$$\nabla_\theta J(\theta) = \nabla_\theta \mathbb{E}_\tau[R(\tau)] = \nabla_\theta \int P(\tau|\theta)R(\tau)d\tau$$

$$= \int \nabla_\theta P(\tau|\theta)R(\tau)d\tau$$

$$= \int P(\tau|\theta) \nabla_\theta \log P(\tau|\theta) R(\tau)d\tau$$

$$= \mathbb{E}_\tau[R(\tau)\nabla_\theta \log P(\tau|\theta)]$$

- Last expression is an *unbiased* gradient estimator.
  Just sample $\tau_i \sim P(\tau|\theta)$, and compute $\hat{g}_i = R(\tau_i)\nabla_\theta \log P(\tau|\theta)$

- Need to be able to *compute and differentiate the density* $P(\tau|\theta)$ w.r.t. $\theta$
Proof

Likelihood (with stochastic policies)

\[ P(\tau|\pi, M) = \rho(s_1) \prod_{i=1}^{H} \pi(s_i, a_i)p(s_{i+1}|s_i, a_i) \]

\[ \log P(\tau|\pi, M) = \log \rho(s_1) + \sum_{i=1}^{H} \log \pi(s_i, a_i) + \log p(s_{i+1}|s_i, a_i) \]

\[ \nabla_{\theta} \log P(\tau|\pi, M) = \nabla_{\theta} \log \rho(s_1) + \sum_{i=1}^{H} \left( \nabla_{\theta} \log \pi(s_i, a_i) + \nabla_{\theta} \log p(s_{i+1}|s_i, a_i) \right) \]
1. Let $\pi_{\theta_1}$ be an arbitrary policy

2. At each iteration $k = 1, \ldots, K$
   - Sample $m$ trajectory $\tau_i = (s_1, a_1, r_1, s_2, \ldots, s_T, a_T, r_T, s_{T+1})$ following $\pi_k$
   - Compute unbiased gradient estimate
     \[
     \hat{\nabla}_\theta J(\pi_{\theta}) = \frac{1}{m} \sum_{i=1}^{m} \left( \sum_{t=1}^{H} r_t^i \right) \left( \sum_{t=1}^{H} \nabla_\theta \log \pi_{\theta_k}(s_t, a_t) \right)
     \]
   - Update parameters
     \[
     \theta_{k+1} = \theta_k + \alpha_k \hat{\nabla}_\theta J(\pi_{\theta_k})
     \]

3. Return last policy $\pi_{\theta_K}$
$\hat{g}_i = R(\tau_i) \nabla_\theta \log P(\tau_i | \pi_\theta, M)$

- $R(\tau_i)$ measures how good is sample $\tau_i$
- Moving in the direction of $\hat{g}_i$ pushes up the log probability of the sample, in proportion to how good it is

Interpretation:
- Uses good trajectories as supervised examples
- Like maximum likelihood in supervised learning
- Good stuff are made more likely while bad less

[Schulman, 2016]
REINFORCE: Intuition

\[ \hat{g}_i = R(\tau_i) \nabla_\theta \log P(\tau_i | \pi_\theta, M) \]

- \( R(\tau_i) \) measures how good is sample \( \tau_i \)
- Moving in the direction of \( \hat{g}_i \) pushes up the log probability of the sample, in proportion to how good it is

[Schulman, 2016]
**REINFORCE: Intuition**

\[ \hat{g}_i = R(\tau_i) \nabla_\theta \log P(\tau_i | \pi_\theta, M) \]

- \( R(\tau_i) \) measures how good is sample \( \tau_i \)
- Moving in the direction of \( \hat{g}_i \) pushes up the log probability of the sample, in proportion to how good it is

*Interpretation:* uses good trajectories as supervised examples
  - *Like maximum likelihood* in supervised learning
  - good stuff are made more likely while bad less (TO REMOVE)
  - Trial and Error approach

[Schulman, 2016]
REINFORCE

Pros

- Easy to compute
- *Does not use Markov property!*
- Can be used in partially observable MDPs without modification
Pros
- Easy to compute
- *Does not use* Markov property!
- Can be used in partially observable MDPs without modification

Issues
- Use an MC estimate of $q(s, a)$
- It has possibly a *very large variance*
- Needs many samples to converge
Policy Gradient: temporal structure

$$\nabla_\theta J(\pi_\theta) = \mathbb{E} \left[ \sum_{t=1}^{H} \nabla_\theta \log \pi_\theta(s_t, a_t) \sum_{t' = t}^{H} r_t \right]$$
Policy Gradient: temporal structure

\[ \nabla_\theta J(\pi_\theta) = \mathbb{E} \left[ \sum_{t=1}^{H} \nabla_\theta \log \pi_\theta(s_t, a_t) \sum_{t'=t}^{H} r_{t'} \right] \]

\[ \mathbb{E}_{a \sim \pi_\theta} \left[ \nabla_\theta \log \pi_\theta(s_t, a) \sum_{t'=1}^{t-1} r_{i} \bigg| \tau_{1:t-1} \right] = \left( \sum_{t'=1}^{t-1} r_{i} \right) \int \pi_\theta(s_t, a) \nabla_\theta \log \pi(s_t, a) da \]

= \left( \sum_{t'=1}^{t-1} r_{i} \right) \int \nabla_\theta \pi(s_t, a) da

= \left( \sum_{t'=1}^{t-1} r_{i} \right) \nabla_\theta \int \pi(s_t, a) da = 0 \]

in literature known as G(PO)MDP [Peters and Schaal, 2008b]
Further reduce the variance by introducing a baseline $b(s)$

$$\nabla_{\theta} J(\pi_\theta) = \mathbb{E} \left[ \sum_{t=1}^H \nabla_{\theta} \log \pi_\theta(s_t, a_t) \left( \sum_{t'=t}^H r_{t'} - b(s_t) \right) \right]$$

The gradient estimate is unbiased

"Near optimal choice" that minimize the variance is the expected sum of returns

$$b^*(s_t) = \mathbb{E} \left[ \sum_{t=1}^T r_t | s_1 = s_t, \pi, M \right]$$

*Interpretation:* increase the log probability of an action $a_t$ proportionally to how much returns are better than expected (relative values)

*Intuition:* $b(s_t)$ does not depend on the action thus

$$\mathbb{E}_{a \sim \pi_\theta} [\nabla_{\theta} \log \pi_\theta(s_t, a) b(s_t) | \tau_{1:t-1}] = 0$$
Baseline derivation

Rough idea

\[
\nabla_{\theta_i} J(\pi_\theta) = \mathbb{E}_\tau [\nabla_{\theta_i} \log P(\tau | \pi_\theta)(R(\tau) - b)] \\
:= g(\tau)
\]

\[
\text{Var} = \mathbb{E}_\tau [(g(\tau)(R(\tau) - b))^2] - (\mathbb{E}_\tau [g(\tau)(R(\tau) - b)])^2
\]

\[
\implies \mathbb{E}_\tau [g(\tau)R(\tau)]^2
\]

baseline is unbiased in expectation

\[
\frac{\partial}{\partial b} \text{Var} = \frac{\partial}{\partial b} \mathbb{E}_\tau [g(\tau)^2(R(\tau) - b)^2]
\]

\[
= \frac{\partial}{\partial b} \mathbb{E}_\tau [g(\tau)^2R(\tau)^2] - 2 \frac{\partial}{\partial b} \mathbb{E}_\tau [g(\tau)^2 R(\tau) \ b] + \frac{\partial}{\partial b} \mathbb{E}_\tau [b^2 g(\tau)^2]
\]

\[
\implies b^*(\tau) = \frac{\mathbb{E}_\tau [g(\tau)^2 R(\tau)]}{\mathbb{E}_\tau [g(\tau)^2]}
\]

Expected return weighted by the magnitude of the gradient
Infinite Horizon
Going beyond the finite-horizon case

**Theorem**

For an infinite horizon MDP (average or discounted), the policy gradient is

\[ \nabla_\theta J(\pi_\theta) = \mathbb{E}_{s \sim d^\pi} \mathbb{E}_{a \sim \pi_\theta(s, \cdot)} [\nabla_\theta \log \pi_\theta(s, a) q^\pi(s, a)] \]

- \( d^\pi \) is the stationary distribution
- \( q^\pi \) is the state-action value function
Infinite-horizon discounted

- Define a distribution $\rho$ over $S$
- The $\gamma$-discounted visitation frequency for policy $\pi$ is

$$d^\pi(s) = \lim_{T \to +\infty} \sum_{t=1}^{T} \gamma^{t-1} \mathbb{P}(s_t = s | \pi, M, \rho)$$

- Then

$$q^\pi(s,a) = \lim_{T \to +\infty} \mathbb{E} \left[ \sum_{t=1}^{T} \gamma^{t-1} r(s_t, a_t) | s_1 = s, a_1 = a, \pi, M \right]$$

$$v^\pi(s) = \lim_{T \to +\infty} \mathbb{E} \left[ \sum_{t=1}^{T} \gamma^{t-1} r(s_t, a_t) | s_1 = s, \pi, M \right] = \sum_a \pi(s,a) q^\pi(s,a)$$

$$J(\pi) = \lim_{T \to +\infty} \mathbb{E} \left[ \sum_{t=1}^{T} \gamma^{t-1} r(s_t, a_t) | \pi, M, \rho \right]$$

$$= \sum_s d^\pi(s) \sum_a \pi(s,a) r(s,a) = \sum_s \rho(s) v^\pi(s)$$
Bellman Equation

\[ q^\pi(s, a) = r(s, a) + \sum_y p(y|s, a) v^\pi(y) \]

\[
\nabla_\theta v^\pi(s) = \sum_a q^\pi(s, a) \nabla_\theta \pi(s, a) + \pi(s, a) \nabla_\theta q^\pi(s, a) \\
= \sum_a q^\pi(s, a) \nabla_\theta \pi(s, a) + \gamma \sum_a \pi(s, a) \sum_y p(y|s, a) \nabla_\theta v^\pi(y)
\]

Bellman equation for the gradient!
Policy Gradient: proof

Multiply by $d^\pi(s)$ and sum over states

$$\mathbb{B} = \sum_s d^\pi(s) \gamma \sum_{a,y} \pi(s, a) p(y|s, a) \nabla_\theta v^\pi(y)$$

$$= \sum_s \sum_{k=0}^{+\infty} \gamma^k \mathbb{P}(s_1 \to s, k, \pi) \gamma \sum_{a,y} \pi(s, a) p(y|s, a) \nabla_\theta v^\pi(y)$$
Multiply by $d^\pi(s)$ and sum over states

$$\mathcal{B} = \sum_s d^\pi(s) \gamma \sum_{a,y} \pi(s, a)p(y|s, a) \nabla_\theta v^\pi(y)$$

$$= \sum_s \sum_{k=0}^{+\infty} \gamma^k \mathbb{P}(s_1 \to s, k, \pi) \gamma \sum_{a,y} \pi(s, a)p(y|s, a) \nabla_\theta v^\pi(y)$$
Policy Gradient: proof

Multiply by $d^{\pi}(s)$ and sum over states

$$\mathbb{B} = \sum_s d^{\pi}(s) \gamma \sum_{a,y} \pi(s, a)p(y|s, a) \nabla_{\theta} v^{\pi}(y)$$

$$= \sum_s \sum_{k=0}^{+\infty} \gamma^k P(s_1 \rightarrow s, k, \pi) \gamma \sum_{a,y} \pi(s, a)p(y|s, a) \nabla_{\theta} v^{\pi}(y)$$

$$= \sum_y \left( \sum_{k=0}^{+\infty} \gamma^{k+1} P(s_1 \rightarrow y, k + 1, \pi) \right) \nabla_{\theta} v^{\pi}(y)$$
Policy Gradient: proof

Multiply by \( d^\pi(s) \) and sum over states

\[
\mathbb{E} = \sum_s d^\pi(s) \gamma \sum_{a,y} \pi(s, a) p(y|s, a) \nabla_\theta v^\pi(y)
\]

\[
= \sum_s \sum_{k=0}^{+\infty} \gamma^k \mathbb{P}(s_1 \rightarrow s, k, \pi) \gamma \sum_{a,y} \pi(s, a) p(y|s, a) \nabla_\theta v^\pi(y)
\]

\[
= \sum_y \left( \sum_{k=0}^{+\infty} \gamma^{k+1} \mathbb{P}(s_1 \rightarrow y, k + 1, \pi) \pm \mathbb{P}(s_1 \rightarrow y, 0, \pi) \right) \nabla_\theta v^\pi(y)
\]
Multiply by \( d^\pi(s) \) and sum over states

\[
\mathbb{B} = \sum_s d^\pi(s) \gamma \sum_{a,y} \pi(s, a) p(y | s, a) \nabla_\theta v^\pi(y)
\]

\[
= \sum_s \sum_{k=0}^{+\infty} \gamma^k \mathbb{P}(s_1 \rightarrow s, k, \pi) \gamma \sum_{a,y} \pi(s, a) p(y | s, a) \nabla_\theta v^\pi(y)
\]

\[
= \sum_y \left( \sum_{k=0}^{+\infty} \gamma^{k+1} \mathbb{P}(s_1 \rightarrow y, k+1, \pi) \pm \mathbb{P}(s_1 \rightarrow y, 0, \pi) \right) \nabla_\theta v^\pi(y)
\]

\[
= \sum_y \left( d^\pi(y) - \mathbb{P}(s_1 \rightarrow y, 0, \pi) \right) \nabla_\theta v^\pi(y)
\]

\[
\triangledown \theta J(\pi)
\]
Policy Gradient: proof

Multiply by $d^\pi(s)$ and sum over states

$$\mathbb{B} = \sum_s d^\pi(s) \gamma \sum_{a,y} \pi(s, a)p(y|s, a)\nabla_\theta v^\pi(y)$$

$$= \sum_s \sum_{k=0}^{+\infty} \gamma^k \mathbb{P}(s_1 \rightarrow s, k, \pi) \gamma \sum_{a,y} \pi(s, a)p(y|s, a)\nabla_\theta v^\pi(y)$$

$$= \sum_y \left( \sum_{k=0}^{+\infty} \gamma^{k+1} \mathbb{P}(s_1 \rightarrow y, k+1, \pi) \pm \mathbb{P}(s_1 \rightarrow y, 0, \pi) \right) \nabla_\theta v^\pi(y)$$

$$= \sum_y \left( d^\pi(y) - \mathbb{P}(s_1 \rightarrow y, 0, \pi) \right) \nabla_\theta v^\pi(y)$$

$$:= \rho(y)$$

Summing up everything

$$\sum_s d^\pi(s) \nabla_\theta v^\pi(s) = \sum_s d^\pi(s) \nabla_\theta \pi(s, a) q^\pi(s, a) + \sum_y d^\pi(y) \nabla_\theta v^\pi(y) - \nabla_\theta \sum_y \rho(y) v^\pi(y)$$

$$= \nabla_\theta J(\pi)$$
Collect $m$ trajectories for policy $\pi$ starting from $s_1 \sim \rho$

For each time $t$

$$\hat{q}_t = \sum_{t'=t}^{T} \gamma^{t'-t} r_{t'}$$

(almost) unbiased estimate $\rightarrow \mathbb{E}[\hat{q}|s_t, a_t] = q^\pi(s_t, a_t)$

Then

$$\nabla_\theta J(\pi_\theta) := \frac{1}{m} \sum_{i=1}^{m} \sum_{t=1}^{T} \gamma^{t-1} \nabla_\theta \log \pi_\theta (s_{i,t}, a_{i,t}) \sum_{t'=t}^{T} \gamma^{t'-t} r_{i,t'}$$
REINFORCE for infinite horizon

- Define $F_t := \hat{q}_t \nabla_\theta \log \pi_\theta(s_t, a_t)$

$$
\mathbb{E} \left[ \sum_{t=1}^{+\infty} \gamma^{t-1} F_t \right] = \sum_{t=1}^{+\infty} \gamma^{t-1} \sum_s \mathbb{E}[F_t | s_t = s] \mathbb{P}(s_t = s | s_1 \sim \rho)
$$

$$
= \sum_{s,a} q^\pi(s, a) \nabla_\theta \pi(s, a) \sum_{t=1}^{+\infty} \gamma^{t-1} \mathbb{P}(s_t = s | s_1 \sim \rho)
$$

$$
= \nabla_\theta J(\pi)
$$

- Almost unbiased (T vs. $+\infty$)
- We can introduce a baseline $b(s_t)$ also in this case
Policy Gradient: example

\[
\nabla_\theta J(\pi_\theta) := \frac{1}{m} \sum_{i=1}^{m} \sum_{t=1}^{T} \gamma^{t-1} \nabla_\theta \log \pi_\theta(s_{i,t}, a_{i,t}) \cdot \hat{q}_{i,t}
\]

How do we represent a policy?
Policy Gradient: example

\[ \nabla_{\theta} J(\pi_\theta) := \frac{1}{m} \sum_{i=1}^{m} \sum_{t=1}^{T} \gamma^{t-1} \nabla_{\theta} \log \pi_\theta(s_{i,t}, a_{i,t}) \cdot \hat{q}_{i,t} \]

How do we represent a policy?

Normal Policy

\[ \pi(a|s) = \frac{1}{\sigma_\omega(s) \sqrt{2\pi}} e^{- \frac{(a - \mu_\theta(s))^2}{2\sigma^2_\omega(s)}} \]

then

\[ \nabla_{\theta} \log \pi(a|s) = \frac{(a - \mu_\theta(s))}{\sigma^2_\omega(s)} \nabla_{\theta} \mu_\theta(s) \]

\[ \nabla_{\omega} \log \pi(a|s) = \frac{(a - \mu_\theta(s))^2 - \sigma^2_\omega(s)}{\sigma^3_\omega(s)} \nabla_{\omega} \sigma_\omega(s) \]
Policy Gradient: example

\[ \nabla_\theta \bar{J}(\pi_\theta) := \frac{1}{m} \sum_{i=1}^{m} \sum_{t=1}^{T} \gamma^{t-1} \nabla_\theta \log \pi_\theta(s_{i,t}, a_{i,t}) \cdot \hat{q}_{i,t} \]

**How do we represent a policy?**

**Normal Policy**

\[ \pi(a|s) = \frac{1}{\sigma_\omega(s) \sqrt{2\pi}} e^{-\frac{(a-\mu_\theta(s))^2}{2\sigma_\omega^2(s)}} \]

then

\[ \nabla_\theta \log \pi(a|s) = \frac{(a - \mu_\theta(s))}{\sigma_\omega^2(s)} \nabla_\theta \mu_\theta(s) \]

\[ \nabla_\omega \log \pi(a|s) = \frac{(a - \mu_\theta(s))^2 - \sigma_\omega^2(s)}{\sigma_\omega^3(s)} \nabla_\omega \sigma_\omega(s) \]

**Gibbs (softmax) policy**

\[ \pi(a|s) = \frac{e^{\kappa Q_\theta(s,a)}}{\sum_{a' \in A} e^{\kappa Q_\theta(s,a')}} \]

then

\[ \nabla_\theta \log \pi(a|s) = \kappa \nabla_\theta Q_\theta(s, a) \]

\[ - \kappa \sum_{a' \in A} \pi(a'|s) \nabla_\theta Q_\theta(s, a') \]
Policy Gradient via Automatic Differentiation

\[ \nabla_{\theta} J(\pi_{\theta}) := \frac{1}{m} \sum_{i=1}^{m} \sum_{t=1}^{T} \gamma^{t-1} \nabla_{\theta} \log \pi_{\theta}(s_{i,t}, a_{i,t}) \cdot \hat{q}_{i,t} \]

- Manually code the derivative can be tedious
  \[ \implies \text{use auto diff} \]
- Define a graph such that its gradient is the policy gradient
  “Pseudo loss”: weighted maximum likelihood

\[ \tilde{J} = \frac{1}{m} \sum_{i=1}^{m} \sum_{t=1}^{T} \log \pi_{\theta}(s_{i,t}, a_{i,t}) \hat{q}_{i,t} \]
Gradient in Practice

Finite-Horizon $\gamma$-discounted setting

$$J_\gamma(\pi) = \mathbb{E} \left[ \sum_{t=1}^{H} \gamma^{t-1} r_t \right]$$

$$\nabla_\theta J_\gamma(\pi) = \mathbb{E} \left[ \sum_{t=1}^{H} \gamma^{t-1} \nabla_\theta \log \pi_\theta(s_t, a_t) q^\pi(s_t, a_t) \right]$$
Gradient in Practice

*Finite-Horizon $\gamma$-discounted setting*

$$J_\gamma(\pi) = \mathbb{E} \left[ \sum_{t=1}^{H} \gamma^{t-1} r_t \right]$$

$$\nabla_\theta J_\gamma(\pi) = \mathbb{E} \left[ \sum_{t=1}^{H} \gamma^{t-1} \nabla_\theta \log \pi_\theta(s_t, a_t) q_\pi(s_t, a_t) \right]$$

*In practice*

$$\nabla_\theta J^?_\gamma(\pi) = \mathbb{E} \left[ \sum_{t=1}^{H} \gamma^{t-1} \nabla_\theta \log \pi_\theta(s_t, a_t) q_\pi(s_t, a_t) \right]$$

$\nabla_\theta J^?_\gamma(\pi)$ is a semi-gradient of the undiscounted objective $J(\pi)$
Gradient in practice

\[ J(\pi) = \mathbb{E} \left[ \sum_{t=1}^{H} r_t \right] \leftrightarrow \nabla_{\theta} J(\pi) = \sum_s d_{\gamma}(s) \frac{\partial}{\partial \theta} v_{\gamma}(s) + \sum_s v_{\gamma}(s) \frac{\partial}{\partial \theta} d_{\gamma}(s) \]

\[ := \nabla_{\theta} J^2(\pi) \]

\[ \text{TD(0) step is also a semi-gradient of the mean squared Bellman error [Sutton and Barto, 2018, Chapter 9]} \]

- In \textit{tabular settings}, semi-gradient TD(0) converges to a minimum of the mean squared error [Jaakkola et al., 1994]
- Also \textit{on-policy} TD with linear function approximatio [Sutton and Barto, 2018]
Gradient in practice

\[ J(\pi) = \mathbb{E} \left[ \sum_{t=1}^{H} r_t \right] \quad \Rightarrow \quad \nabla_\theta J(\pi) = \sum_s d_\gamma^\pi(s) \frac{\partial}{\partial \theta} v_\gamma^\pi(s) + \sum_s v_\gamma^\pi(s) \frac{\partial}{\partial \theta} d_\gamma^\pi(s) \]

\[ := \nabla_\theta J^2(\pi) \]

\[ \text{TD}(0) \text{ step is also a semi-gradient of the mean squared Bellman error} \ [\text{Sutton and Barto, 2018, Chapter 9}] \]

- In *tabular settings*, semi-gradient TD(0) converges to a minimum of the mean squared error [Jaakkola et al., 1994]
- Also *on-policy* TD with linear function approximatio [Sutton and Barto, 2018]

👍 Semi-policy gradient may converge to a BAD policy w.r.t. both discounted and undiscounted objectives

*Impossibility result* [Nota and Thomas, 2019]:

\[ \# f(\pi) \in C \text{ such that } \nabla_\theta J^2(\pi) = \frac{\partial}{\partial \theta} f(\pi) \]

(Example?)
Convergence Results
Convergence Results

- Policy gradient is *stochastic gradient*

\[ \theta_{k+1} = \theta_k + \alpha_k (\nabla J(\theta_k) + \text{noise}) \]

- \( J \) is non-convex

\[ \Rightarrow \] converge asymptotically to a stationary point or a local minimum (*under standard technical assumptions*)
Convergence Results

- Policy gradient is *stochastic gradient*

\[ \theta_{k+1} = \theta_k + \alpha_k (\nabla J(\theta_k) + \text{noise}) \]

- \( J \) is non-convex

- \( \implies \) converge asymptotically to a stationary point or a local minimum (under standard technical assumptions)

  what is the *quality* of this point?
Policy gradient is *stochastic gradient*

\[ \theta_{k+1} = \theta_k + \alpha_k (\nabla J(\theta_k) + \text{noise}) \]

- \( J \) is *non-convex*
- \( J \) converge asymptotically to a stationary point or a local minimum *(under standard technical assumptions)*

what is the *quality* of this point?

*Dynamics are linear (LQ systems) \( \implies \) global convergence* [Fazel et al., 2018]

Surprising since \( \min_{\pi} J_{LQ}(\pi) \) may be not convex, quasi-convex, and star-convex
but (far from boundaries) \( J_{LQ} \) is “almost” smooth

*Hints*: use properties of functions that are gradient dominated
Convergence Results

Issues

- **Non-convexity of the loss function**
- **Unnatural policy parameterization**: parameters that are far in Euclidean distance may describe the same policy (*we will talk about this later*)
- **Insufficient exploration**: naive stochastic exploration
- **Large variance of stochastic gradients**: generally increases with the length of the horizon

Solution:

\[ \Rightarrow \]

Similar to LQ, we need structural assumptions [Bhandari and Russo, 2019]

See also [Zhang et al., 2019] for convergence results
Convergence Results

**Issues**

- **Non-convexity of the loss function**
- **Unnatural policy parameterization:** parameters that are far in Euclidean distance may describe the same policy (*we will talk about this later*)
- **Insufficient exploration:** naive stochastic exploration
- **Large variance of stochastic gradients:** generally increases with the length of the horizon

**Solution:**

⇒ similar to LQ, we need structural assumptions [Bhandari and Russo, 2019]

See also [Zhang et al., 2019] for convergence results
Convergence Results: Structural Properties
[Bhandari and Russo, 2019]

Let $\Pi_\theta = \{\pi_\theta | \theta \in \Theta\}$ being the space of parametrized policies

1. Closure under policy improvement

$$\forall \pi \in \Pi_\theta, \ \exists \pi^+ \in \Pi_\theta \quad \text{s.t.} \quad \pi^+ \in \arg \max q^\pi$$

2. Convexity of policy improvement steps

$$q^\pi (s, a) \text{ is convex in } a$$

3. Convexity of the policy class $\Pi_\theta$

soft policy-iteration update $(1 - \alpha)\pi + \alpha\pi^+$ is feasible

4. Regularity conditions

  e.g., compactness of $S$, existence and continuity of derivatives w.r.t. $\theta$, etc.
Global convergence

- Consider the structural properties
- Consider infinite-horizon discounted problems
Global convergence

- Consider the structural properties
- Consider infinite-horizon discounted problems

No suboptimal stationary points by following a specific ascent direction

\[ \Rightarrow \text{global convergence} \] [Bhandari and Russo, 2019]
Consider the structural properties

Consider infinite-horizon discounted problems

No suboptimal stationary points by following a specific ascent direction

\[ \Rightarrow \text{global convergence} \] [Bhandari and Russo, 2019]

Idea:

\[ \pi_{\theta_\alpha} := (1 - \alpha)\pi_\theta + \alpha\pi_{\theta'} \in \Pi_\theta \]

\( \alpha \in [0, 1] \) defines a line in the policy space

What is the direction to follow in the parameter space?
Global convergence

- Consider the structural properties
- Consider infinite-horizon discounted problems

No suboptimal stationary points by following a specific ascent direction

$$\implies \text{global convergence} \ [\text{Bhandari and Russo, 2019}]$$

Idea:

$$\pi_{\theta, \alpha} := (1 - \alpha) \pi_\theta + \alpha \pi_{\theta'} \in \Pi_\theta$$

$$\alpha \in [0, 1]$$ defines a line in the policy space

What is the direction to follow in the parameter space?

find \( u \) such that the directional derivative of \( \pi' \) points in the direction of \( \pi' \) (smooth curve in the parameter space)

Follow the directional derivative between \( \pi_{\theta_k} \) and \( \pi_{k+} \)
Global convergence

- Consider the structural properties
- Consider infinite-horizon discounted problems

**No suboptimal stationary points** by following a specific ascent direction

$$\implies \text{global convergence} [\text{Bhandari and Russo, 2019}]$$

**Idea:**

$$\pi_{\theta_\alpha} := (1 - \alpha)\pi_\theta + \alpha\pi_{\theta'} \in \Pi_\theta$$

$\alpha \in [0, 1]$ defines a line in the policy space

*What is the direction to follow in the parameter space?*

Find $u$ such that the *directional derivative* of $\pi'$ points in the direction of $\pi'$ (smooth curve in the parameter space)

Follow the directional derivative between $\pi_{\theta_k}$ and $\pi_{k+}^+$

*Forward connection: conservative policy iteration and adaptive gradient*
Actor-Critic
REINFORCE

- Monte-Carlo policy gradient is unbiased but still has high variance
Monte-Carlo policy gradient is unbiased but _still_ has high variance. Define an alternative estimate of $q^\pi(s, a) \implies$ actor-critic

- **Critic**: estimate the value function
- **Actor**: update the policy in the direction suggested by the critic
Actor-Critic

- Actor-critic algorithms maintain two sets of parameters: $\theta \mapsto \pi, \omega \mapsto q^\pi$
- **Critic can use TD(0)**

```latex
for t = 1, \ldots, T do
    a_t \sim \pi^\theta(s_t, \cdot) \text{ and observer } r_t \text{ and } s_{t+1}
    \text{Compute temporal difference}
    \delta_t = r_t + \gamma q_\omega(s_{t+1}, a_{t+1}) - q_\omega(s_t, a_t)
    \text{Update } q \text{ estimate}
    \omega = \omega + \beta \delta_t \nabla_\omega q_\omega(x_t, a_t)
    \text{Update policy}
    \theta = \theta + \alpha \nabla_\theta \log \pi_\theta(s_t, a_t) q_\omega(s_t, a_t)
end
```

TD(0) is a semi-gradient approach [Baird, 1995, Sutton, 2015]
**Issues:**

- \( q_\omega(s, a) \) is a biased estimate of \( q^{\pi_\theta}(s, a) \)
- The update of \( \theta \) may not follow the gradient of \( \nabla_\theta J(\pi_\theta) \)

**Solution:**

- Choose the approximation space \( q_\omega(s, a) \) carefully
  \( \Longrightarrow \) **compatible function** approximation between \( q_\omega \) and \( \pi_\theta \)
Compatible Function Approximation

**Theorem**

An action value function space \( q_\omega \) is compatible with a policy space \( \pi_\theta \) if

\[
q_\omega(s, a) = \omega^T \nabla_\theta \log \pi_\theta(s, a)
\]

If \( \omega \) minimizes the squared Bellman residual

\[
\omega = \arg \min_\omega \mathbb{E}_{s \sim d_\pi} \left[ \sum_a \pi_\theta(s, a) (q^{\pi_\theta}(s, a) - q_\omega(s, a))^2 \right]
\]

Then

\[
\nabla_\theta J(\pi_\theta) = \mathbb{E}_{s \sim d_\pi} \mathbb{E}_{a \sim \pi_\theta} [\nabla_\theta \log \pi_\theta(s, a) q_\omega(s, a)]
\]
Actor-Critic with a baseline

\[ \nabla_\theta J(\pi_\theta) = \mathbb{E}_{s \sim d^\pi_\theta} \left[ \sum a \nabla_\theta \pi_\theta(s, a)(q^\pi_\theta(s, a) - b(s)) \right] \]

- \( b(s) \) minimizes the variance
- \( v^\pi(s) \) is a good choice as baseline
  - it minimizes the variance in average reward [Bhatnagar et al., 2009]
- \( A^\pi(s, a) = q^\pi(s, a) - v^\pi(s) \) is the advantage function
Actor-Critic with advantage function

- It is possible to estimate $v^\pi$ and $q^\pi$ \textit{independently} (e.g., by TD(0))
Actor-Critic with advantage function

- It is possible to estimate $v^\pi$ and $q^\pi$ independently (e.g., by TD(0))
- $A^\pi = q_\omega - v_\nu$ is a biased and unstable estimate
Actor-Critic with advantage function

- It is possible to estimate $v^\pi$ and $q^\pi$ independently (e.g., by TD(0))
- $A^\pi = q_\omega - v_\nu$ is a biased and unstable estimate

**Solution:**

- Consider the temporal difference error

$$\delta^{\pi\theta} = r(s, a) + \gamma v^{\pi\theta}(s') - v^{\pi\theta}(s)$$
Actor-Critic with advantage function

- It is possible to estimate $v^\pi$ and $q^\pi$ independently (e.g., by TD(0))
- $A^\pi = q_\omega - v_\nu$ is a biased and unstable estimate

Solution:

- Consider the temporal difference error

$$
\delta^\pi_{\theta} = r(s, a) + \gamma v^\pi_{\theta}(s') - v^\pi_{\theta}(s)
$$

- $\delta^\pi_{\theta}$ is an unbiased estimate of the advantage

$$
\mathbb{E}[\delta^\pi_{\theta} | s, a] = \mathbb{E}[r(s, a) + \gamma v^\pi_{\theta}(s') | s, a] - v^\pi_{\theta}(s) = q^\pi_{\theta}(s, a) - v^\pi_{\theta}(s)
$$
Actor-Critic with advantage function

- Estimate only $v_{\nu} \mapsto \delta_{\nu} = r + \gamma v_{\nu}(s') - v_{\nu}(s)$

**Convergence results** with compatible function approximation [Bhatnagar et al., 2009]

for $t = 1, \ldots, T$ do

- $a_t \sim \pi^\theta(s_t, \cdot)$ and observer $r_t$ and $s_{t+1}$

  Compute temporal difference

  $$\delta_t = r_t + \gamma v_{\nu}(s_{t+1}) - v_{\nu}(s_t)$$

- Update $v$ estimate

  $$v = \omega + \beta \delta_t \nabla\nu v_{\nu}(s_t)$$

- Update policy

  $$\theta = \theta + \alpha \delta_t \nabla_{\theta} \log \pi_{\theta}(s_t, a_t)$$

end
Several recent methods [Gu et al., 2017, Thomas and Brunskill, 2017, Grathwohl et al., 2018, Liu et al., 2018, Wu et al., 2018] have extended to state-action baselines

\[ b(s) \rightarrow b(s, a) \]

👍 unbiased when compatible function approximation is used (proof?)

*Is really working?* See [Tucker et al., 2018] for complete investigation!
So far we have observed fully online actor-critic approaches. In some cases it can be inefficient (e.g., for training approximators) → batching.
From online to batch actor-critic

- So far we have observed fully online actor-critic approaches
- In some case it can be *inefficient* (e.g., for training approximators)
  
  \[ \implies \text{batching} \]

1. Sample trajectories \( \tau_i = \{ s_1, a_1, r_1, \ldots, s_{T+1} \} \) using \( \pi_\theta \)

\[
\hat{v}(s_{i,t}) = \sum_{k=t}^{t+p} \gamma^{k-t} r_k + \gamma^p v_\nu(s_{t+p+1}) \quad \text{bootstrapping}
\]
From online to batch actor-critic

- So far we have observed fully online actor-critic approaches.
- In some cases, it can be inefficient (e.g., for training approximators).

$$\Rightarrow \text{batching}$$

1. Sample trajectories $\tau_i = \{s_1, a_1, r_1, \ldots, s_{T+1}\}$ using $\pi_\theta$

$$\hat{v}(s_{i,t}) = \sum_{k=t}^{t+p} \gamma^{k-t} r_k + \gamma^p v_\nu(s_{t+p+1}) \quad \text{bootstrapping}$$

2. Use supervised regression on $D = \{(s_{i,t}, \hat{v}(s_{i,t}))\}$

$$\arg \min_{\nu} \frac{1}{2} \sum_{(s,\hat{v}) \in D} (v_\nu(s) - \hat{v})^2$$
Sample Efficiency in Actor-Critic

Issues:

- Sample efficiency is pretty poor
- All samples need to be generated by the current policy (on-policy learning)
- Samples are \textit{discarded} after a single update
Sample Efficiency in Actor-Critic

**Issues:**
- Sample efficiency is pretty poor
- All samples need to be generated by the current policy (*on-policy learning*)
- Samples are *discarded* after a single update

**Solutions**
- Use samples from other policies via *importance sampling* (*not very stable*)
- *Conservative approaches*
- Variance reduction techniques
- Newton or Quasi-newton methods
Off-policy Policy Gradient

- Usual approach [Wang et al., 2017]
  - Store observed samples (a.k.a. replay buffer)
  - Off-policy policy evaluation is “easy” (cf. LSTDQ [Lagoudakis and Parr, 2003a])

\[ \pi_k \mapsto v^{\pi_k} \]
Off-policy Policy Gradient

- Usual approach [Wang et al., 2017]
  - Store observed samples (a.k.a. replay buffer)
  - Off-policy policy evaluation is “easy” (cf. LSTDQ [Lagoudakis and Parr, 2003a])

\[ \pi_k \mapsto v^{\pi_k} \]

**Issue:**

- The estimate of the gradient requires samples from \( \pi_\theta \)
- Use *importance ratios* to avoid introducing additional bias
Important Weighting

\[ \mathbb{E}_{x \sim p}[f(x)] = \mathbb{E}_{x \sim q}\left[\frac{p(x)}{q(x)}f(x)\right] \approx \mu_q = \frac{1}{N} \sum_{i=1}^{N} \frac{p(x_i)}{q(x_i)}f(x_i), \quad x_i \sim q \]
Importance Weighting

\[ \mathbb{E}_{x \sim p}[f(x)] = \mathbb{E}_{x \sim q} \left[ \frac{p(x)}{q(x)} f(x) \right] \approx \mu_q = \frac{1}{N} \sum_{i=1}^{N} \frac{p(x_i)}{q(x_i)} f(x_i), \quad x_i \sim q \]

**Variance**

\[
\text{var}(\mu_q) = \frac{1}{N} \text{var} \left( \frac{p(x)}{q(x)} f(x) \right) \\
= \frac{1}{N} \left( \mathbb{E}_{x \sim p} \left[ \frac{p(x)}{q(x)} f(x)^2 \right] - \mathbb{E}_{x \sim p}[f(x)]^2 \right)
\]

The term in red may explode!
Importance Weighting in Policy Gradient
[Jurcícek, 2012, Degris et al., 2012]

\[ \nabla_{\theta} J(\pi_\theta) = \mathbb{E}_{\tau \sim \beta} \left[ \frac{\mathbb{P}(\tau | \pi_\theta)}{\mathbb{P}(\tau | \beta)} \sum_{t=1}^{T} \gamma^{t-1} \nabla_{\theta} \log \pi_\theta(s_t, a_t) q^{\pi_\theta}(s_t, a_t) \right] \]

what’s the issue?

Partial fixes: clipping, normalization, etc.

Off-policy RL is still a relevant open problem
Importance Weighting in Policy Gradient

[Jurcicek, 2012, Degris et al., 2012]

\[ \nabla_{\theta} J(\pi_\theta) = \mathbb{E}_{\tau \sim \beta} \left[ \frac{\mathbb{P}(\tau|\pi_\theta)}{\mathbb{P}(\tau|\beta)} \sum_{t=1}^{T} \gamma^{t-1} \nabla_{\theta} \log \pi_\theta(s_t, a_t) q_\theta^\pi(s_t, a_t) \right] \]

What’s the issue? Exploding or vanishing importance weights

\[ \omega(\beta, \pi_\theta|\tau) := \frac{\mathbb{P}(\tau|\pi_\theta)}{\mathbb{P}(\tau|\beta)} = \frac{\rho(s_1) \prod_{t=1}^{T} p(s_{t+1}|s_t, a_t) \pi_\theta(s_t, a_t)}{\rho(s_1) \prod_{t=1}^{T} p(s_{t+1}|s_t, a_t) \beta(s_t, a_t)} = \prod_{t=1}^{T} \frac{\pi_\theta(s_t, a_t)}{\beta(s_t, a_t)} \]

Partial fixes: clipping, normalization, etc.

Off-policy RL is still a relevant open problem
Sample efficiency through variance-reduced gradient
Variance-reduced gradient estimator

Can we do something better?  

Visualization idea from Bach [2016]
SVRG [Johnson and Zhang, 2013]

Stochastic Variance-Reduced Gradient

A solution from finite-sum optimization:

\[
\max_{\theta} J(\theta) = \sum_{i=1}^{N} f_i(\theta)
\]

- Unbiased
- Linear convergence

SVRG estimator

\[\nabla J(\theta) = \nabla J(\tilde{\theta}) + \nabla f_i(\theta) - \nabla f_i(\tilde{\theta})\]

FG (snapshot)

SG in current parameter

Correction term

More data-efficient than FG

Supervised Learning (SL)
Algorithm 1 SVRG

**Input:** a dataset $\mathcal{D}_N$, number of epochs $S$, epoch size $m$, step size $\alpha$, initial parameter $\theta^0_m := \tilde{\theta}^0$

for $s = 0$ to $S - 1$ do

$\theta^{s+1}_0 := \tilde{\theta}^s = \theta^s_m$

$\tilde{\mu} = \nabla f(\tilde{\theta}^s)$

for $t = 0$ to $m - 1$ do

$x \sim \mathcal{U}(\mathcal{D}_N)$

$v_{t+1}^{s+1} = \tilde{\mu} + \nabla z(x|\theta^{s+1}_t) - \nabla z(x|\tilde{\theta}^s)$

$\theta^{s+1}_{t+1} = \theta^{s+1}_t + \alpha v_{t+1}^{s+1}$

end for

end for

Concave case: return $\theta^S_m$

Non-Concave case: return $\theta^{s+1}_t$ with $(s, t)$ picked uniformly at random from $\{[0, S - 1] \times [0, m - 1]\}$
SVRG for RL: SVRPG

Issues in RL:
- non-concavity
- infinite dataset
- non-stationarity: $\tau \sim \pi_{\theta}$

Solution:

\[
\nabla J(\theta) = \nabla_{N} J(\tilde{\theta}) + \nabla_{B} J(\theta) - \omega(\theta, \tilde{\theta}) \nabla_{B} J(\tilde{\theta})
\]

SVRPG estimator

Large $N$ to approximate FG

Importance weighting for non-stationarity

epoch

iteration
For $s = 1, \ldots$

Sample $N$ trajectories using $\tilde{\theta}$

Compute $\text{FG} = \hat{\nabla}_N J(\tilde{\theta})$

For $t = 1, \ldots, m$

Sample $B$ trajectories using $\theta$

Compute $\text{SG} = \hat{\nabla}_B J(\theta)$

Compute correction $= \omega(\theta, \tilde{\theta}) \hat{\nabla}_B J(\tilde{\theta})$

Update $\theta \leftarrow \theta + \alpha \nabla J(\theta)$

Update $\tilde{\theta} \leftarrow \theta$

iteration

epoch
Importance sampling may reintroduce variance (use all the tricks)

(a) SVRPG vs G(PO)MDP on Cart-pole.

(b) Self-Normalized SVRPG vs SVRPG on Swimmer.

(c) Self-Normalized SVRPG vs G(PO)MDP on Swimmer.
Conservative Approaches
Relative Performance

**Issues:**
- We would like to exploit past samples
- We do not know how much to trust them
- Depends on the distribution over trajectories induced by different policies
Relative Performance

Issues:
- We would like to exploit past samples
- We do not know how much to trust them
- Depends on the distribution over trajectories induced by different policies

Performance-Difference Lemma

[Burnetas and Katehakis, 1997, Prop. 1], [Kakade and Langford, 2002, Lem. 6.1], [Cao, 2007]

For any policies \( \pi, \pi' \in \Pi^{SR} \)

\[
J(\pi') - J(\pi) = \sum_{s,a} d^{\pi'}(s,a) A^\pi(s,a)
\]

\[
= \sum_s d^{\pi'}(s) \sum_a \pi'(s,a) A^\pi(s,a)
\]
Proof

\[ \mathbb{E}_{(s,a) \sim d^\pi'} [A^\pi (s, a)] = \mathbb{E}_{(s,a) \sim d^\pi'} [q^\pi (s, a) - v^\pi (s)] \]

\[ = \mathbb{E}_{(s,a) \sim d^\pi'} [r(s, a)] + \mathbb{E}_{(s,a) \sim d^\pi'} \left[ \gamma \sum_y p(y | s, a) v^\pi (y) - v^\pi (s) \right] \]

\[ = J(\pi') + \mathbb{E}_{(s,a) \sim d^\pi'} \left[ \gamma \sum_y p(y | s, a) v^\pi (y) \right] - \mathbb{E}_{s \sim d^\pi'} [v^\pi (s)] \]
Proof

\[ \mathbb{E}_{(s,a) \sim d^\pi'} [A^\pi (s, a)] = \mathbb{E}_{(s,a) \sim d^\pi'} [q^\pi (s, a) - v^\pi (s)] \]

\[ = \mathbb{E}_{(s,a) \sim d^\pi'} [r (s, a)] + \mathbb{E}_{(s,a) \sim d^\pi'} \left[ \gamma \sum_y p(y|s,a) v^\pi (y) - v^\pi (s) \right] \]

\[ = J(\pi') + \mathbb{E}_{(s,a) \sim d^\pi'} \left[ \gamma \sum_y p(y|s,a) v^\pi (y) \right] - \mathbb{E}_{s \sim d^\pi'} [v^\pi (s)] \]

\[ = \sum_s \left( \sum_{k=0}^{+\infty} \gamma^k \mathbb{P}(s_1 \rightarrow s, k, \pi', \rho) \right) \gamma \sum_{a,y} \pi'(s, a) p(y|s,a) v^\pi (y) \]

\[ = \sum_y \left( d^\pi' (y) - \underbrace{\mathbb{P}(s_1 \rightarrow y, 0, \pi, \rho)}_{:= \rho(y)} \right) v^\pi (y) \]
Proof

\[ \mathbb{E}_{(s,a) \sim d^{\pi^*}} [A^{\pi^*}(s, a)] = \mathbb{E}_{(s,a) \sim d^{\pi^*}} [q^{\pi^*}(s, a) - v^{\pi^*}(s)] \]

\[ = \mathbb{E}_{(s,a) \sim d^{\pi^*}} [r(s, a)] + \mathbb{E}_{(s,a) \sim d^{\pi^*}} \left[ \gamma \sum_y p(y|s, a) v^{\pi^*}(y) - v^{\pi^*}(s) \right] \]

\[ = J(\pi^*) + \mathbb{E}_{(s,a) \sim d^{\pi^*}} \left[ \gamma \sum_y p(y|s, a) v^{\pi^*}(y) \right] - \mathbb{E}_{s \sim d^{\pi^*}} [v^{\pi^*}(s)] \]

\[ = \sum_s \left( \sum_{k=0}^{+\infty} \gamma^k \mathbb{P}(s_1 \rightarrow s, k, \pi', \rho) \right) \gamma \sum_{a,y} \pi'(s, a) p(y|s, a) v^{\pi^*}(y) \]

\[ = \sum_y \left( d^{\pi'}(y) - \mathbb{P}(s_1 \rightarrow y, 0, \pi, \rho) \right) v^{\pi^*}(y) \]

\[ := \rho(y) \]

\[ = J(\pi^*) + \sum_y d^{\pi'}(y) v^{\pi^*}(y) - \sum_y \rho(y) v^{\pi^*}(y) - \mathbb{E}_{s \sim d^{\pi^*}} [v^{\pi^*}(s)] \]
Optimization step

\[
\max_{\pi'} J(\pi')
\]

Issue: as before, cannot be directly estimated using information from \(\pi\).
Optimization step

\[
\max_{\pi'} J(\pi') = \max_{\pi'} J(\pi') - J(\pi)
\]

Issue: as before, cannot be directly estimated using information from \(\pi\)
Optimization step

\[
\max_{\pi'} J(\pi') = \max_{\pi'} J(\pi') - J(\pi)
\]
\[
= \max_{\pi'} \mathbb{E}_{(s,a) \sim d_{\pi'}} [A^{\pi}(s,a)]
\]

*Issue:* as before, cannot be directly estimated using information from \(\pi\)
Optimization step

\[ J(\pi') - J(\pi) = \mathbb{E}_{s \sim d^{\pi}} \left[ \sum_a \pi'(s, a) A^{\pi}(s, a) \right] + \sum_s (d^{\pi'}(s) - d^{\pi}(s)) \sum_a \pi'(s, a) A^{\pi}(s, a) \]
Optimization step

\[ J(\pi') - J(\pi) = \mathbb{E}_{s \sim d_\pi} \left[ \sum_a \pi'(s, a) A^\pi(s, a) \right] + \sum_s \left( \frac{d'(s) - d(\pi)}{} \right) \sum_a \pi'(s, a) A^\pi(s, a) \]

\[ \geq \mathbb{E}_{s \sim d_\pi} \left[ \sum_a \pi'(s, a) A^\pi(s, a) - \frac{\gamma \varepsilon}{(1 - \gamma)^2} D_{TV}(\pi' \parallel \pi)[s] \right] \]

where \( \varepsilon = \max_s \left| \mathbb{E}_{a \sim \pi'}[A^\pi(s, a)] \right| \) and

\[ D_{TV}(\pi' \parallel \pi)[s] = \sum_a |\pi'(s, a) - \pi(s, a)| \]
Surrogate Loss

\[ L_\pi(\pi') = J(\pi) + \sum_s d^\pi(s) \sum_a \pi'(s, a) A^\pi(s, a) \]

- \( L_\pi(\pi) = J(\pi) \)
- If parametric policies \( \pi = \pi_\theta \), \( \nabla_\theta L_{\pi_\theta}(\pi_\theta) = \nabla_\theta J(\pi_\theta) \)

in an interval close to \( \pi \), \( L_\pi \) is a good surrogate for \( J \)

\[ \implies \text{Conservative Policy Iteration} \quad [\text{Kakade and Langford, 2002}] \]
Surrogate Loss

\[ L_{\pi}(\pi') = J(\pi) + \sum_s d^{\pi}(s) \sum_a \pi'(s, a) A^{\pi}(s, a) - \sum_s d^{\pi}(s) \frac{\gamma \varepsilon}{(1 - \gamma)^2} D_{TV}(\pi' || \pi)[s] \]

- \( L_{\pi}(\pi) = J(\pi) \)
- If parametric policies \( \pi = \pi_{\theta} \), \( \nabla_{\theta} L_{\pi_{\theta}}(\pi_{\theta}) = \nabla_{\theta} J(\pi_{\theta}) \)

! in an interval close to \( \pi \), \( L_{\pi} \) is a good surrogate for \( J \)

\[ \implies \text{Conservative Policy Iteration} \] [Kakade and Langford, 2002]
Conservative Policy Iteration

- New policy improvement schema
  - Give current policy $\pi_k$, solve

\[
\max_{\pi'} \left\{ L_{\pi_k}(\pi') - C \mathbb{E}_{s \sim d^\pi} [D_{TV}(\pi' || \pi_k)[s]] \right\}
\]
Conservative Policy Iteration

- **New policy improvement schema**
  - Give current policy $\pi_k$, solve
  
  \[
  \max_{\pi'} \left\{ L_{\pi_k}(\pi') - C \mathbb{E}_{s \sim d^\pi} \left[ D_{TV}(\pi' \| \pi_k)[s] \right] \right\} \geq 0
  \]
Conservative Policy Iteration

- **New policy improvement schema**
  - Give current policy $\pi_k$, solve

\[
J(\pi') - J(\pi_k) \geq \max_{\pi'} \left\{ L_{\pi_k}(\pi') - C \mathbb{E}_{s \sim d^\pi} \left[ D_{TV}(\pi'|\pi_k)[s] \right] \right\} \geq 0
\]
Conservative Policy Iteration

- **New policy improvement schema**
  - Give current policy $\pi_k$; solve

\[
J(\pi') - J(\pi_k) \geq \max_{\pi'} \left\{ L_{\pi_k}(\pi') - C \mathbb{E}_{s \sim d^\pi} [D_{TV}(\pi' || \pi_k)(s)] \right\} \geq 0
\]

\[\Rightarrow \text{Monotonic performance improvement}\]
Conservative Policy Iteration

- **New policy improvement schema**
  - Give current policy $\pi_k$, solve

  \[
  J(\pi') - J(\pi_k) \geq \max_{\pi'} \left\{ L_{\pi_k}(\pi') - C \mathbb{E}_{s \sim d_{\pi}} \left[ D_{TV}(\pi' \parallel \pi_k)[s] \right] \right\} \geq 0
  \]

  **Monotonic performance improvement**

Several approaches have been proposed [e.g., Kakade and Langford, 2002, Perkins and Precup, 2002, Gabillon et al., 2011, Wagner, 2011, 2013, Pirotta et al., 2013b, Scherrer et al., 2015, Schulman et al., 2015]
Approximate Monotone Improvement

- The objective can be estimated using rollouts from the most recent policy.
- Updates respect a notion of distance in the policy space!

This is the basis for many algorithms!
How to solve the optimization problem?

\[
\max_{\pi'} \left\{ L_{\pi_k}(\pi') - C \mathbb{E}_{s \sim d^\pi} \left[ D_{TV}(\pi' \| \pi_k)[s] \right] \right\}
\]
How to solve the optimization problem?

\[
\max_{\pi'} \left\{ L_{\pi_k}(\pi') - C \mathbb{E}_{s \sim d^\pi} \left[ D_{TV}(\pi'\|\pi_k)[s] \right] \right\}
\]

In discrete MDP with convex policy update

\[
\pi_{k+1} = \alpha \bar{\pi} + (1 - \alpha) \pi_k
\]

where \(\bar{\pi}\) is the greedy policy

\(\implies\) closed form solution for \(\alpha\)

\(\implies\) guaranteed improvement
Conservative in Continuous MDPs

- Consider parametrized policies $\theta \mapsto \pi_\theta$
- Construct a *lower bound* to $J(\theta + \Delta\theta) - J(\theta)$
  - e.g., [Pirotta et al., 2013, Papini et al., 2017]
Consider parametrized policies $\theta \mapsto \pi_\theta$

Construct a lower bound to $J(\theta + \Delta \theta) - J(\theta)$
- e.g., [Pirotta et al., 2013, Papini et al., 2017]

If $\Pi_\theta$ is a smoothing policy class [Papini et al., 2019]
(as a consequence of quadratic bound for $L$-smooth functions)

$$\forall \theta, \theta' \quad J(\theta') - J(\theta) \geq (\theta' - \theta)^T \nabla_\theta J(\theta) - \frac{L}{2} \|\theta' - \theta\|^2_2$$
Consider parametrized policies $\theta \mapsto \pi_\theta$

Construct a lower bound to $J(\theta + \Delta \theta) - J(\theta)$

- e.g., [Pirotta et al., 2013, Papini et al., 2017]

If $\Pi_\theta$ is a smoothing policy class [Papini et al., 2019]
(as a consequence of quadratic bound for $L$-smooth functions)

\[ \forall \theta, \theta' \quad J(\theta') - J(\theta) \geq (\theta' - \theta)^T \nabla_\theta J(\theta) - \frac{L}{2} \|\theta' - \theta\|^2 \]

\[ = \alpha \|\nabla_\theta J(\theta)\|^2 - \alpha^2 \frac{L}{2} \|\nabla_\theta J(\theta)\|^2 \]

by using gradient update rule $\theta' = \theta + \alpha \nabla_\theta J(\theta)$
Consider parametrized policies $\theta \mapsto \pi_\theta$

Construct a lower bound to $J(\theta + \Delta \theta) - J(\theta)$
- e.g., [Pirotta et al., 2013, Papini et al., 2017]

If $\Pi_\theta$ is a smoothing policy class [Papini et al., 2019]
(as a consequence of quadratic bound for $L$-smooth functions)

$$
\forall \theta, \theta' \quad J(\theta') - J(\theta) \geq (\theta' - \theta)^T \nabla_\theta J(\theta) - \frac{L}{2} \|\theta' - \theta\|^2_2
$$

$$
= \alpha \|\nabla_\theta J(\theta)\|^2_2 - \alpha^2 \frac{L}{2} \|\nabla_\theta J(\theta)\|^2_2
$$

by using gradient update rule $\theta' = \theta + \alpha \nabla_\theta J(\theta)$

$$
\implies \alpha^* = \frac{1}{L} \quad \implies \text{Monotonic policy performance improvement}
$$
Conservative Approaches: Approximation

- Can be extended to handle *approximate estimate*
  \[ \| A(s, a) - \hat{A}(s, a) \| \leq \epsilon \quad \text{and/or} \quad \| \nabla J(\theta) - \hat{\nabla} J(\theta) \| \leq \epsilon \]

- Need to change the stopping condition to *account for the finite-sample error*
Conservative Approaches: Approximation

- Can be extended to handle *approximate estimate*

\[ \|A(s, a) - \hat{A}(s, a)\| \leq \epsilon \quad \text{and/or} \quad \|\nabla J(\theta) - \hat{\nabla} J(\theta)\| \leq \epsilon \]

- Need to change the stopping condition to *account for the finite-sample error*

**Example:** \( \hat{\nabla}_N J(\theta) \) estimate of the gradient using \( N \) trajectories. Then *whp*

\[ \|\nabla J(\theta) - \hat{\nabla}_N J(\theta)\| \leq \frac{\epsilon \delta}{\sqrt{N}} \]

As a consequence, *whp*

\[ J(\theta') - J(\theta) \geq \alpha \left( \|\nabla_\theta J(\theta)\|_2^2 - \frac{\epsilon^2 \delta}{N} \right) - \alpha^2 \frac{L}{2} \|\nabla_\theta J(\theta)\|_2^2 \]
Conservative Approaches: Approximation

- Can be extended to handle *approximate estimate*

\[ \|A(s, a) - \hat{A}(s, a)\| \leq \epsilon \quad \text{and/or} \quad \|\nabla J(\theta) - \hat{\nabla} J(\theta)\| \leq \epsilon \]

- Need to change the stopping condition to *account for the finite-sample error*

**Example:** \( \hat{\nabla}_N J(\theta) \) estimate of the gradient using \( N \) trajectories. Then *whp*

\[ \|\nabla J(\theta) - \hat{\nabla}_N J(\theta)\| \leq \frac{\epsilon \delta}{\sqrt{N}} \]

As a consequence, *whp*

\[ J(\theta') - J(\theta) \geq \alpha \left( \|\nabla_{\theta} J(\theta)\|_2^2 - \frac{\epsilon^2 \delta}{N} \right) - \alpha^2 \frac{L}{2} \|\nabla_{\theta} J(\theta)\|_2^2 \]

+ possibility to adapt also \( N \)
Toward Practical Algorithm

- Optimizing the total variation $D_{TV}(\pi'\|\pi)$ may be difficult

- Relax the problem using Pinsker’s inequality [Csiszar and Körner, 2011]

$$D_{TV}(\pi'\|\pi) \leq \sqrt{2D_{KL}(\pi'\|\pi)}$$

* implicitly done in the analysis of conservative gradient
Kullback–Leibler divergence

Given two probability distributions $P$ and $Q$

$$D_{KL}(P\|Q) = \sum_x P(x) \log \frac{P(x)}{Q(x)}$$

Properties:

- $D_{KL}(P\|Q) \geq 0$
- $D_{KL}(Q\|Q) = 0$
- $D_{KL}(P\|Q) \neq D_{KL}(Q\|P)$ (non-symmetric)
- No triangle inequality

**Note:** Réni divergences provide generalizations of the KL divergence
Further Steps toward Practical Algorithms

- $C'$ provided by theory is quite high (*too conservative*)
- Replace regularization with constraint (*trust region*) (e.g., REPS [Peters et al., 2010])

$$\pi_{k+1} = \arg \max_{\pi'} L_\pi(\pi')$$

$$\text{s.t. } \mathbb{E}_{s \sim d_\pi} [D_{KL}(\pi' || \pi)] \leq \delta$$
Further Steps toward Practical Algorithms

- $C'$ provided by theory is quite high (too conservative)
- Replace regularization with constraint (trust region) (e.g., REPS [Peters et al., 2010])

$$
\pi_{k+1} = \arg \max_{\pi'} L_\pi(\pi')
$$

$$\text{s.t. } \mathbb{E}_{s \sim d_\pi} [D_{KL}(\pi' \| \pi)] \leq \delta$$

- Importance weighting

$$
\mathbb{E}_{s \sim d_\pi} \mathbb{E}_{a \sim \pi'} [A_\pi(s, a)] = \mathbb{E}_{s \sim d_\pi} \mathbb{E}_{a \sim z} \left[ \frac{\pi'(s, a)}{z(s, a)} A_\pi(s, a) \right]
$$
Further Steps toward Practical Algorithms

- $C$ provided by theory is quite high (too conservative)
- Replace regularization with constraint (trust region) (e.g., REPS [Peters et al., 2010])

$$\pi_{k+1} = \arg \max_{\pi'} L_\pi(\pi')$$
$$\text{s.t. } \mathbb{E}_{s \sim d_\pi} [D_{KL}(\pi'\|\pi)] \leq \delta$$

- Importance weighting

$$\mathbb{E}_{s \sim d_\pi} \mathbb{E}_{a \sim \pi'} [A^\pi(s, a)] = \mathbb{E}_{s \sim d_\pi} \mathbb{E}_{a \sim z} \left[ \frac{\pi'(s, a)}{z(s, a)} A^\pi(s, a) \right]$$

- Replace $A^\pi$ with $q^\pi$ and remove $J(\pi)$

$$\pi_{k+1} = \arg \max_{\pi'} \mathbb{E}_{s \sim d_\pi} \mathbb{E}_{a \sim z} \left[ \frac{\pi'(s, a)}{z(s, a)} q^\pi(s, a) \right]$$
$$\text{s.t. } \mathbb{E}_{s \sim d_\pi} [D_{KL}(\pi'\|\pi)] \leq \delta$$

$$\implies$$ Trust-Region Policy Optimization (TRPO) [Schulman et al., 2015]
Beyond Simple Gradient Descent
Gradient Descent

**Steepest descent direction** of a function $h(\theta) \rightarrow -\nabla h(\theta)$

- It yields the **most reduction** in $h$ per unit of change in $\theta$
- Change is measured using the standard **Euclidean norm** $\| \cdot \|$

\[
\frac{-\nabla h}{\|\nabla h\|} = \lim_{\epsilon \to 0} \frac{1}{\epsilon} \arg \min_{d: \|d\| \leq \epsilon} \{ h(\theta + d) \}
\]

Is the Euclidean norm the best metric? Can we use an alternative definition of (local) distance?\[\Rightarrow\] as suggested by [Amari, 1998] it is better to define a metric based not on the choice of the coordinates but rather on the manifold these coordinates parametrize! (Example: gradient descent is not affine invariant)
Gradient Descent

**Steepest descent direction** of a function $h(\theta) \rightarrow -\nabla h(\theta)$

- It yields the *most reduction* in $h$ per unit of change in $\theta$
- Change is measured using the standard *Euclidean norm* $\| \cdot \|$

$$\frac{-\nabla h}{\|\nabla h\|} = \lim_{\epsilon \to 0} \frac{1}{\epsilon} \arg \min_{d: \|d\| \leq \epsilon} \{ h(\theta + d) \}$$

Is the Euclidean norm the best metric?
Can we use an alternative definition of *(local)* distance?
Gradient Descent

**Steepest descent direction** of a function $h(\theta) \rightarrow -\nabla h(\theta)$

- It yields the *most reduction* in $h$ per unit of change in $\theta$
- Change is measured using the standard *Euclidean norm* $\| \cdot \|$

\[
\frac{-\nabla h}{\| \nabla h \|} = \lim_{\epsilon \rightarrow 0} \frac{1}{\epsilon} \arg \min_{d: \|d\| \leq \epsilon} \{ h(\theta + d) \}
\]

Is the Euclidean norm the best metric?
Can we use an alternative definition of *(local)* distance?

\[ \Rightarrow \quad \text{as suggested by [Amari, 1998] it is better to } \textit{define a metric} \text{ based not on the choice of the coordinates but rather } \textit{on the manifold these coordinates parametrize}! \]

(Example: gradient descent is not affine invariant)
Natural Gradient

In Riemannian space, the distance is defined as

$$d^2(v, v + \delta v) = \delta v^T G(v) \delta v$$

where $G$ is the metric tensor
Natural Gradient

- In Riemannian space, the distance is defined as

\[ d^2(v, v + \delta v) = \delta v^T G(v) \delta v^T \]

where \( G \) is the metric tensor

**Example:** consider the Euclidean space \((\mathbb{R}^2)\)
- Cartesian coordinate, the metric tensor is the identity
- Polar coordinate

\[ x = r \cos \theta \implies \delta x = \delta r \cos \theta - r \delta \theta \sin \theta \]
\[ y = r \sin \theta \implies \delta y = \delta r \sin \theta + r \delta \theta \cos \theta \]
\[ d^2(v, v + \delta v) = \delta x^2 + \delta y^2 = \delta r^2 + r^2 \delta \theta^2 = (\delta r, \delta \theta)^T \text{diag}(1, r^2) (\delta r, \delta \theta) \]
Natural Gradient

The steepest descent in a Riemannian is given by

\[ \tilde{\nabla} h(\theta) = G(\theta)^{-1} \nabla h(\theta) \]
Natural Gradient

The steepest descent in a Riemannian is given by

\[ \tilde{\nabla} h(\theta) = G(\theta)^{-1} \nabla h(\theta) \]

Natural gradient can be applied to any objective function

*Issue:* what is the metric tensor?
Natural Gradient

Natural Gradient [Amari, 1998]

The steepest descent in a Riemannian is given by

\[ \tilde{\nabla} h(\theta) = G(\theta)^{-1} \nabla h(\theta) \]

Natural gradient can be applied to any objective function

*Issue:* what is the metric tensor?

*known for many objectives!*
The steepest descent in a Riemannian is given by

$$\tilde{\nabla} h(\theta) = G(\theta)^{-1} \nabla h(\theta)$$

Natural gradient can be applied to any objective function

**Issue:** what is the metric tensor?

*known for many objectives!*

**Maximum Likelihood:** we have a probabilistic model represented by its likelihood $p(x|\theta)$

We want to maximize this likelihood function to find the most likely parameter
Consider a Gaussian parameterized by only its mean and keep the variance fixed to 2 and 0.5 for the first and second image respectively.

The distance of those Gaussians are the same, i.e., 4, according to Euclidean metric (red line).

https://wiseodd.github.io/techblog/2018/03/14/natural-gradient/
Fisher Information Matrix

\[ F = \mathbb{E}_{x \sim p(\cdot | \theta)} \left[ \nabla \log p(x | \theta) \nabla \log p(x | \theta)^T \right] \]

**Property 1:** Fisher Information Matrix is the Hessian of KL-divergence between two distributions \( p(x | \theta) \) and \( p(x | \theta') \), with respect to \( \theta' \), evaluated at \( \theta = \theta' \)

\[ H_{DKL}(p(x | \theta) \| p(x | \theta')) = F \]
Fisher Information Matrix

\[ F = \mathbb{E}_{x \sim p(\cdot|\theta)} \left[ \nabla \log p(x|\theta) \nabla \log p(x|\theta)^T \right] \]

Property 1: Fisher Information Matrix is the Hessian of KL-divergence between two distributions \( p(x|\theta) \) and \( p(x|\theta') \), with respect to \( \theta' \), evaluated at \( \theta = \theta' \)

\[ H_{DKL}(p(x|\theta) || p(x|\theta')) = F \]

Property 2: Second-order Taylor series expansion

\[ D_{KL}(p(x|\theta) || p(x|\theta + d)) = d^T F d + O(d^3) \]

(proofs)
For a positive definite matrix \( A \), we have [Ollivier et al., 2017] (def. \( \|x\|_B = \sqrt{x^T B x} \))

\[
- A^{-1} \nabla h \| \nabla h \|_{A^{-1}} = \lim_{\epsilon \to 0} \frac{1}{\epsilon} \arg \min_{d: \|d\|_{A^{-1}} \leq \epsilon} \{ h(\theta + d) \}
\]
For a positive definite matrix $A$, we have [Ollivier et al., 2017] (def. $\|x\|_B = \sqrt{x^T B x}$)

$$\frac{-A^{-1} \nabla h}{\|\nabla h\|_{A^{-1}}} = \lim_{\epsilon \to 0} \frac{1}{\epsilon} \arg \min_{d: \|d\|_{A^{-1}} \leq \epsilon} \{h(\theta + d)\}$$

$$A = \frac{1}{2} F \implies -\sqrt{2} \frac{\tilde{\nabla} h}{\|\nabla h\|_{F^{-1}}} = \lim_{\epsilon \to 0} \frac{1}{\epsilon} \arg \min_{d: D_{KL}(p(x|\theta)||p(x|\theta+d)) \leq \epsilon^2} \{h(\theta + d)\}$$

**Negative natural gradient**

- steepest descent direction *in the space of distributions*
- where distance is *(approximately)* measured in local neighborhoods by the KL divergence
Natural Gradient in ML

[Martens, 2014]

For a positive definite matrix $A$, we have [Ollivier et al., 2017] (def. $\|x\|_B = \sqrt{x^T B x}$)

$$\frac{-A^{-1} \nabla h}{\|\nabla h\|_{A^{-1}}} = \lim_{\epsilon \to 0} \frac{1}{\epsilon} \arg \min_{d: \|d\|_{A^{-1}} \leq \epsilon} \{h(\theta + d)\}$$

$$A = \frac{1}{2} F \implies -\sqrt{2} \frac{\tilde{\nabla} h}{\|\nabla h\|_{F^{-1}}} = \lim_{\epsilon \to 0} \frac{1}{\epsilon} \arg \min_{d: D_{KL}(p(x|\theta)||p(x|\theta+d)) \leq \epsilon^2} \{h(\theta + d)\}$$

Negative natural gradient

- steepest descent direction in the space of distributions
- where distance is (approximately) measured in local neighborhoods by the KL divergence
- $D_{KL}(p(x|\theta)||p(x|\theta+d))$ is locally/asymptotically symmetric as $d \to 0$, and so will be (approximately) symmetric in a local neighborhood [Martens, 2014]
- $\tilde{\nabla} h$ is be invariant to the choice of parameterization

Natural Gradient in ML

[Martens, 2014]

For a positive definite matrix $A$, we have [Ollivier et al., 2017] (def. $\|x\|_B = \sqrt{x^T B x}$)

$$\frac{-A^{-1} \nabla h}{\|\nabla h\|_{A^{-1}}} = \lim_{\epsilon \to 0} \frac{1}{\epsilon} \arg \min_{d: \|d\|_{A^{-1}} \leq \epsilon} \{h(\theta + d)\}$$

$$A = \frac{1}{2} F \implies -\sqrt{2} \frac{\tilde{\nabla} h}{\|\nabla h\|_{F^{-1}}} = \lim_{\epsilon \to 0} \frac{1}{\epsilon} \arg \min_{d: D_{KL}(p(x|\theta)||p(x|\theta+d)) \leq \epsilon^2} \{h(\theta + d)\}$$

Negative natural gradient

- steepest descent direction in the space of distributions
- where distance is (approximately) measured in local neighborhoods by the KL divergence
- $D_{KL}(p(x|\theta)||p(x|\theta+d))$ is locally/asymptotically symmetric as $d \to 0$, and so will be (approximately) symmetric in a local neighborhood [Martens, 2014]
- $\tilde{\nabla} h$ is be invariant to the choice of parameterization
Natural Policy Gradient

Trust-region objective

\[ \pi_{k+1} = \arg \max_{\pi'} L_{\pi_k}(\pi') \]

s.t. \( D_{KL}(\pi'\|\pi_k) \leq \delta \)

Approximate objective and KL

\[ L_{\theta_k}(\theta) \approx L_{\theta_k}(\theta_k) + g^T(\theta - \theta_k) \]

\[ D_{KL}(\theta\|\theta_k) \approx \frac{1}{2}(\theta - \theta_k)^T F(\theta - \theta_k) \]

\[ \theta_{k+1} = \theta_k + \sqrt{\frac{2\delta}{g^T F^{-1} g}} \underbrace{F^{-1} g}_{:= \tilde{\nabla} J} \]

Truncated Natural Policy Gradient

Issues:

- $\theta \in \mathbb{R}^d$, $d$ can be very large (e.g., thousands or millions)
- $H$ or $F$ have dimension $d^2$
- matrix inversion is $O(d^3)$
Truncated Natural Policy Gradient

**Issues:**

- $\theta \in \mathbb{R}^d$, $d$ can be very large (e.g., thousands or millions)
- $H$ or $F$ have dimension $d^2$
- matrix inversion is $O(d^3)$

**Solution:**

- Use conjugate gradient to compute $F^{-1}g$ without inverting $F$ [Pascanu and Bengio, 2013]
- With $j$ iterations, CG solves systems of equations $Hx = g$ for $x$ by finding projection onto Krylov subspace (i.e., $\text{span}(g, Hg, \ldots H^{j-1}g)$)

$\Rightarrow$ **Truncated Natural Policy Gradient**
Truncated Natural Policy Gradient

Issues:

- $\theta \in \mathbb{R}^d$, $d$ can be very large (e.g., thousands or millions)
- $H$ or $F$ have dimension $d^2$
- matrix inversion is $O(d^3)$

Solution:

- Use conjugate gradient to compute $F^{-1}g$ without inverting $F$ [Pascanu and Bengio, 2013]
- With $j$ iterations, CG solves systems of equations $Hx = g$ for $x$ by finding projection onto Krylov subspace (i.e., $span(g, Hg, \ldots H^{j-1}g)$)

$\implies$ Truncated Natural Policy Gradient

Other solutions are possible: see ACKTR [Wu et al., 2017], [Ollivier, 2017]
Example: Walker-2d

[Duan et al., 2016]
Discussion

- Natural gradient contains second order informations
- Newton method?
Discussion

- Natural gradient contains second order informations
- Newton method?

The Hessian [Furmston and Barber, 2012, Shen et al., 2019]

\[ \nabla^2 J(\theta) = \mathbb{E}_\tau \left[ \nabla g(\theta, \tau) \nabla \log \mathbb{P}(\tau|\theta)^T + \nabla^2 g(\theta, \tau) \right] \]

with

\[ g(\theta, \tau) = \sum_{h=1}^{H} \sum_{i=h}^{H} \gamma^i r(s_i, a_i) \log \pi_\theta(s_h, a_h) \]
[Furmston and Barber, 2012] noticed a connection between $\mathbb{E}[\nabla^2 g(\theta, \tau)]$ and the FIM!

This hessian can be estimated using first-order information (leading to *quasi Newton approaches*) or *finite difference*.

- see [Shen et al., 2019] also for sample complexity

REINFORCE find an $\epsilon$-approximate first-order stationary point in $O(1/\epsilon^4)$

Hessian aided policy gradient method [Shen et al., 2019] sample complexity of $O(1/\epsilon^3)$
Proximal Policy Optimization
[Schulman et al., 2017b]

- Avoid to compute the natural gradient
- Approximate the KL constraint

Adaptive KL Penalty

Consider regularized optimization problem

$$\theta_{k+1} = \arg \max_{\theta} L_{\theta_k}(\theta) - \lambda_k \mathbb{E}[D_{KL}(\theta \| \theta_k)]$$

Adapt $\lambda_k$ to enforce KL constraint

$$\lambda_{k+1} = \begin{cases} 2 \lambda_k & \text{if } \mathbb{E}[D_{KL}(\theta \| \theta_k)] \geq 1.5 \\ 2 \frac{\lambda_k}{2} & \text{if } \mathbb{E}[D_{KL}(\theta \| \theta_k)] \leq \frac{\delta}{1.5} \\ \lambda_k & \text{otherwise} \end{cases}$$
Proximal Policy Optimization

[Schulman et al., 2017b]

- Avoid to compute the natural gradient
- Approximate the KL constraint

1. **Adaptive KL Penalty**
   - Consider regularized optimization problem
     \[
     \theta_{k+1} = \arg \max_{\theta} L_{\theta_k}(\theta) - \lambda_k \mathbb{E}[D_{KL}(\theta||\theta_k)]
     \]
   - Adapt \( \lambda_k \) to enforce KL constraint
     \[
     \lambda_{k+1} = \begin{cases} 
     2\lambda_k & \text{if } \mathbb{E}[D_{KL}(\theta||\theta_k)] \geq 1.5\delta \\
     \lambda_k/2 & \text{if } \mathbb{E}[D_{KL}(\theta||\theta_k)] \leq \delta/1.5 \\
     \lambda_k & \text{otherwise}
     \end{cases}
     \]
Proximal Policy Optimization
[Schulman et al., 2017b]

2 Clipped Objective

- Recall surrogate objective

\[ L^{IS}_{\pi}(\pi') = \mathbb{E}_{s \sim d^\pi} \mathbb{E}_{a \sim \pi} \left[ \frac{\pi'(s, a)}{\pi(s, a)} A^\pi(s, a) \right] = \mathbb{E}_{s \sim d^\pi} \mathbb{E}_{a \sim \pi} \left[ r_{sa}(\pi') A^\pi(s, a) \right] \]

- Form a lower bound via clipped importance ratios

\[ L^{CLIP}_{\pi}(\pi') = \mathbb{E}_{s \sim d^\pi} \mathbb{E}_{a \sim \pi} \left[ \min \left\{ r_{sa}(\pi') A^\pi(s, a), \text{clip}(r_{sa}(\pi'), 1 - \epsilon, 1 + \epsilon) A^\pi(s, a) \right\} \right] \]

- \( \pi_{k+1} = \arg \max_{\pi} L^{CLIP}_{\pi_k}(\pi) \)
Proximal Policy Optimization
[Schulman et al., 2017b]

- Clipping prevents policy from moving too much away from $\theta_k$
- Seems to work as well as PPO with KL penalty
- Much simpler to implement

*How does it work?*

Various objectives as a function of function of $\alpha$ between $\theta_k$ and $\theta_{k+1}$
Figure 3: Comparison of several algorithms on several MuJoCo environments, training for one million timesteps.
Non-Parametric Policy Update

- Solve a constrained optimization problem in a non-parameterized policy space
- *Fit a parametric policy* on the best non-parametric policy

⇒ Supervised Policy Update [Vuong et al., 2019]
Non-Parametric Policy Update

- Solve a constrained optimization problem in a non-parameterized policy space
- *Fit a parametric policy* on the best non-parametric policy

$$\implies$$ Supervised Policy Update [Vuong et al., 2019]

1. Sample \( N \) trajectories using policy \( \pi_{\theta_k} \)
   - construct dataset \((s_i, a_i, A_i)\) where \( A_i \approx A^{\pi_k}(s_i, a_i) \)
2. For each \( s_i \) solve the constrained optimization problem
   - obtain a non-parametric policy \( \tilde{\pi} \) defined in each sample \( s_i \)
3. Fit a parametric policy \( \pi_{\theta_{k+1}} \) on \( \pi \)

$$\min_{\theta} \left\{ \mathcal{L}(\theta) = \frac{1}{m} \sum_{i=1}^{m} D_{KL}(\pi_\theta || \tilde{\pi})[s_i] \right\}$$
Example: TRPO optimization problem

Almost closed form solution (up to parameters $\lambda = f(\delta, \epsilon)$)

$$\tilde{\pi}(s, a) \propto \pi_{\theta_k}(s, a) \exp \left[ \frac{A^{\pi_{\theta_k}}(s, a)}{\lambda} \right]$$
Non-Parametric Policy Update

Example: TRPO optimization problem
Almost closed form solution (up to parameters $\lambda = f(\delta, \epsilon)$)

$$
\pi(s, a) \propto \pi_{\theta_k}(s, a) \exp \left[ \frac{A^{\pi_{\theta_k}}(s, a)}{\lambda} \right]
$$

Then (approximately)

$$
\mathcal{L}(\theta) \approx \frac{1}{m} \sum_{i=1}^{m} \left( \nabla_{\theta} D_{KL}(\pi_{\theta} \parallel \pi_{\theta_k})[s_i] - \frac{1}{\lambda} \frac{\nabla_{\theta} \pi_{\theta}(s_i, a_i)}{\pi_{\theta_k}(s_i, a_i)} A_i \right) \mathbb{1}(D_{KL}(\pi_{\theta} \parallel \pi_{\theta_k})[s_i] \leq \epsilon)
$$
Non-Parametric Policy Update

Example: TRPO optimization problem
Almost closed form solution (up to parameters $\lambda = f(\delta, \epsilon)$)

$$\tilde{\pi}(s, a) \propto \pi_{\theta_k}(s, a) \exp \left[ \frac{A_{\pi_{\theta_k}}(s, a)}{\lambda} \right]$$

Then (approximately)

$$\mathcal{L}(\theta) \approx \frac{1}{m} \sum_{i=1}^{m} \left( \nabla_{\theta} D_{KL}(\pi_{\theta} \parallel \pi_{\theta_k})[s_i] - \frac{1}{\lambda} \frac{\nabla_{\theta} \pi_{\theta}(s_i, a_i)}{\pi_{\theta_k}(s_i, a_i)} A_i \right) 1(D_{KL}(\pi_{\theta} \parallel \pi_{\theta_k})[s_i] \leq \epsilon)$$

! minimize by gradient descent and consider $\lambda$ to be a parameter!
still an actor-critic approach!
Example: TRPO optimization problem

Almost closed form solution (up to parameters $\lambda = f(\delta, \epsilon)$)

$$\tilde{\pi}(s, a) \propto \pi_{\theta_k}(s, a) \exp \left[ \frac{A^{\pi_{\theta_k}}(s, a)}{\lambda} \right]$$

Then (approximately)

$$\mathcal{L}(\theta) \approx \frac{1}{m} \sum_{i=1}^{m} \left( \nabla_{\theta} D_{\text{KL}}(\pi_{\theta} || \pi_{\theta_k})[s_i] - \frac{1}{\lambda} \frac{\nabla_{\theta} \pi_{\theta}(s_i, a_i)}{\pi_{\theta_k}(s_i, a_i)} A_i \right) 1 \left( D_{\text{KL}}(\pi_{\theta} || \pi_{\theta_k})[s_i] \leq \epsilon \right)$$

minimize by gradient descent and consider $\lambda$ to be a parameter!
still an actor-critic approach!
Not really a novel idea $\implies$ Classification-based PI
Classification-based Policy Iteration (RCPI)

- replaces the policy evaluation step with computing rollout estimates of $q^\pi$

  \[
  \mathcal{D} = \{x_i\}_{i=1}^N \mapsto \hat{q}^\pi
  \]

- casts the policy improvement step as a classification problem
  - find a policy in a given hypothesis space that best predicts the greedy action at every (observed) state

  \[
  \min_{\pi \in \Pi} \frac{1}{N} \sum_{i=1}^N \left( \max_a \hat{q}^{\pi k}(s_i, a) - \hat{q}^{\pi k}(s_i, \pi(s_i)) \right)
  \]

Estimate the return of a state-action pair as

\[ R_{j}^{\pi_{k}}(s_i, a) = R_{j}^{\pi_{k},H}(s_i, a) + \gamma^{H} \hat{v}_{i}^{\pi_{k}}(s_{i}^{H}) \]

with

\[ R_{j}^{\pi_{k},H}(s_i, a) = r(s_i, a) + \sum_{t=1}^{H-1} \gamma^{t} r(x_{ij}^{t}, \pi_{k}(x_{ij}^{t})) \]

Then

\[ \hat{q}^{\pi_{k}}(s_i, a) = \frac{1}{m} \sum_{j=1}^{m} R_{j}^{\pi_{k}}(s_i, a) \]
Discussion

Key components:

1. Stochastic policies
2. Regularized or constrained optimization

What are the motivations

- Exploration
- Controlling the deviation
- Differentiability of Bellman operator

So far regularization was coming from lower bound to the performance. Can we analyse it independently?
Stochastic vs. Deterministic Policies

\[ J_D(\pi) = \mathbb{E}_{s \sim d^\pi} [r(s, \pi(s))] \]

**Deterministic Policy Gradient**

\[ \nabla_\theta J_D(\theta) = \sum_s d^\pi(s) \nabla_\theta \pi_\theta(s) \nabla_a q^\pi(s, a)|_{a=\pi_\theta(s)} \]

\[ = \mathbb{E}_{s \sim d^\pi} [\nabla_\theta \pi_\theta(s) \nabla_a q^\pi(s, a)|_{a=\pi_\theta(s)}] \]

**Issues:**

- We need to be able to differentiate the model
- Explicitly force exploration at the level of actions
Stochastic vs. Deterministic Policies

Plug it into an actor-critic framework

⇒ Use TD(0) to update a parametric representation of $q^\pi$

$$\delta_t = R_t + \gamma Q_w(s_{t+1}, a_{t+1}) - Q_w(s_t, a_t)$$

$w_{t+1} = w_t + \alpha_w \delta_t \nabla_w Q_w(s_t, a_t)$

$\theta_{t+1} = \theta_t + \alpha_\theta \nabla_a Q_w(s_t, a_t) \nabla_\theta \mu_\theta(s)|_{a=\mu_\theta(s)}$

; TD error in SARSA

; Deterministic policy gradient theorem
Softmax Operator

\[
v^*(s) = \max_a \left\{ r(s, a) + \gamma \sum_y p(y|s, a)v^*(y) \right\}
\]

replace \( \max \) with “softmax” operator

\[
v^*(s) = \frac{1}{\eta} \log \left( \sum_a \exp \left[ \eta \left( r(s, a) + \gamma \sum_y p(y|s, a)v^*(y) \right) \right] \right)
\]

The two approaches are connected by Lagrangian duality when

$$\Omega(\pi(s, \cdot)) = \sum_a \pi(s, a) \log \pi(s, a) \quad \text{negative entropy}$$
**Entropy Regularization**

\[
\max_{\pi} \left\{ J(\pi) = \mathbb{E} \left[ \sum_{t=1}^{+\infty} \gamma^{t-1} r_t + \alpha \Omega(\pi(s_t, \cdot)) \right] \right\}
\]

The two approaches are connected by Lagrangian duality when

\[
\Omega(\pi(s, \cdot)) = \sum_a \pi(s, a) \log \pi(s, a)
\]  

**Results:** [Neu et al., 2017]

- Existence and uniqueness
- Well-defined contractive DP operator
- Policy Gradient Theorem
Entropic Regularization

Optimal policy:

$$\pi^*(s, a) \propto \exp \left[ \eta (r(s, a) + \gamma \mathbb{E}'_s[v^*(s')]) \right]$$

Note:

$$q^\pi(s, a) = r(s, a) + \gamma \sum_y p(y|s, a)v^\pi(y)$$

$$v^\pi(s) = \mathbb{E}_{a \sim \pi}[q^\pi(s, a)] - \Omega(\pi(s, \cdot))$$
Soft-Actor Critic

1. Train the value function $v$

$$
\arg \min_{\psi} \mathbb{E}_{s_t \sim H} \left[ \frac{1}{2} \left( v_\psi(s_t) + \mathbb{E}_{a_t \sim \pi} [q_\theta(s_t, a_t) - \log \pi_\phi(s_t, a_t)] \right)^2 \right]
$$

2. Train the action-value function $q^\pi$

$$
\arg \min_{\theta} \mathbb{E}_{(s,a) \in H} \left[ \frac{1}{2} \left( q_\theta(s_t, a_t) - (r(s_t, a_t) + \gamma \mathbb{E}[v_\psi(s')] \right)^2 \right]
$$

! fix the target network (e.g., DQN) → increase stability / break dependences

3. Fit the new policy

$$
\arg \min_{\phi} \mathbb{E}_{s \in H} \left[ D_{KL}(\pi_\psi \| \exp[\eta q_\psi] / Z) [s] \right]
$$
Suppose the MDP is deterministic (otherwise take a conditional expectation w.r.t. to history)

For any $v^*, \pi^*$ optimizing the regularized objective

$$v^*(s) - \gamma v^*(s') = r(s, a) - \eta \log \pi^*(s, a)$$

$$v^*(s_1) - \gamma^{t-1} v^*(s_t) = \sum_{t=1}^{t-1} \gamma^{i-1} (r(s_i, a_i) - \eta \log \pi^*(s_i, a_i))$$

if $(\pi, v)$ satisfies the path consistency for every $(s, a)$, then $\pi = \pi^*$ and $v = v^*$
Path-Consistency Learning

- Maintain two sets of parameters \((\phi, \theta)\): \(\theta \mapsto \pi_\theta, \phi \mapsto v_\phi\)
- Minimize the consistency error

\[
\min_{\phi, \theta} O_{PCL}(\phi, \theta, H) = \sum_{s_{i:i+d} \in E_H} \frac{1}{2} C(s_{i:i+d}, \phi, \theta)^2
\]

where \(E_H\) is the set of (sub)trajectories and

\[
C(s_{i:i+d}, \phi, \theta) = -v_\phi(s_{i}) + \gamma^d v_\phi(s_{i+d}) + \sum_{j=0}^{d-1} \gamma^j (r(s_{i+j}, a_{i+j}) - \eta \log \pi_\theta(s_{a+j}, a_{i+j}))
\]
Path-Consistency Learning

- Maintain two sets of parameters $(\phi, \theta)$: $\theta \mapsto \pi_\theta$, $\phi \mapsto v_\phi$
- Minimize the consistency error

$$\min_{\phi, \theta} O_{PCL}(\phi, \theta, H) = \sum_{s_{i:i+d} \in E_H} \frac{1}{2} C(s_{i:i+d}, \phi, \theta)^2$$

where $E_H$ is the set of (sub)trajectories and

$$C(s_{i:i+d}, \phi, \theta) = -v_\phi(s_i) + \gamma^d v_\phi(s_{i+d}) + \sum_{j=0}^{d-1} \gamma^j (r(s_{i+j}, a_{i+j}) - \eta \log \pi_\theta(s_{a+j}, a_{i+j}))$$

In practice:
- Use replay buffer
- Update incrementally $\Rightarrow$ semi-batch
Path-Consistency Learning

- Maintain two sets of parameters \((\phi, \theta)\): \(\theta \mapsto \pi_\theta, \phi \mapsto v_\phi\)
- Minimize the consistency error

\[
\min_{\phi, \theta} O_{PCL}(\phi, \theta, H) = \sum_{s_{i:i+d} \in E_H} \frac{1}{2} C(s_{i:i+d}, \phi, \theta)^2
\]

where \(E_H\) is the set of (sub)trajectories and

\[
C(s_{i:i+d}, \phi, \theta) = -v_\phi(s_i) + \gamma^d v_\phi(s_{i+d}) + \sum_{j=0}^{d-1} \gamma^j (r(s_{i+j}, a_{i+j}) - \eta \log \pi_\theta(s_{a+j}, a_{i+j}))
\]

**In practice:**
- Use replay buffer
- Update incrementally \(\Rightarrow\) semi-batch

Can be extended to different regularizers (e.g., Shannon entropy, Tsallis entropy [Chow et al., 2018])
Regularized Markov Decision Processes
[Geist et al., 2019]

Bellman operator

\[ L^\pi v(s) = \sum_a \pi(s, a) \left( r(s, a) + \gamma \sum_y p(y|s, a)v^\pi(y) \right) = \sum_a \pi(s, a)q^\pi(s, a) \]

Optimal Bellman operator

\[ L^* v(s) = \max_a \left\{ r(s, a) + \gamma \sum_y p(y|s, a)v^*(y) \right\} \]

Greedy policy

\[ L^* v = L_{\pi'} v \iff \pi' \in \arg\max_{\pi} L^\pi v \]
Regularized Markov Decision Processes

**Regularizer**

\[ \Omega : \mathcal{P}(\mathcal{A}) \to \mathcal{S} \quad \text{strongly convex function} \]

**Legendre-Fenchel transform** (or convex conjugate)

\[ \Omega^* : \mathbb{R}^A \to \mathbb{R} \]

\[ \forall q \in \mathbb{R}^A, \quad \Omega^*(q) = \max_{z \in \mathcal{P}(\mathcal{A})} \left\{ \sum_s z(a)q(a) - \Omega(z) \right\} \]
Regularized Markov Decision Processes

**Regularizer**

\[ \Omega : \mathcal{P}(A) \rightarrow S \]

*strongly convex function*

**Legendre-Fenchel transform** (or convex conjugate)

\[ \Omega^* : \mathbb{R}^A \rightarrow \mathbb{R} \]

\[
\forall q \in \mathbb{R}^A, \quad \Omega^*(q) = \max_{z \in \mathcal{P}(A)} \left\{ \sum_s z(a)q(a) - \Omega(z) \right\}
\]

Property of strongly convex functions: *unique maximizing argument*

\[ \nabla \Omega^* \text{ is Lipschitz and} \quad \nabla \Omega^*(q) = \arg \max_{z \in \mathcal{P}(A)} \left\{ \sum_s z(a)q(a) - \Omega(z) \right\} \]
Examples:

<table>
<thead>
<tr>
<th></th>
<th>$\Omega(\pi(s, \cdot))$</th>
<th>$\Omega^*(q(s, \cdot))$</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Negative entropy</strong></td>
<td>$\sum_a \pi_s(a) \log \pi(s, a)$</td>
<td>$\log \sum_a \exp q(s, a)$</td>
</tr>
<tr>
<td></td>
<td>$\nabla \Omega^*(q(s, \cdot)) = \frac{\exp q(s, a)}{\sum_b \exp q(s, b)}$</td>
<td>i.e., softmax</td>
</tr>
<tr>
<td><strong>KL-divergence</strong></td>
<td>$\sum_a \pi(s, a) \log \pi(s, a) + \log(A)$</td>
<td>$\ln \sum_a \frac{1}{A} \exp[[q(s, a)]$</td>
</tr>
<tr>
<td>between $\pi$ and uniform</td>
<td>$\nabla \Omega^* \text{ is Mellowmax [Asadi and Littman, 2017]}$</td>
<td></td>
</tr>
<tr>
<td><strong>Tsallis entropy</strong></td>
<td>$\frac{1}{2} (|\pi(s, \cdot)|_2^2 - 1)$</td>
<td>$\nabla \Omega^* \text{ is the sparsemax [Chow et al., 2018]}$</td>
</tr>
<tr>
<td>($q = 2, k = 1/2$)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Regularized Markov Decision Processes

**Regularized** Bellman operators w.r.t. $\Omega$

$$L^\pi_\Omega v(s) = L^\pi v(s) - \Omega(\pi(s, \cdot)) = \sum_a \pi(s, a)q^\pi(s, a) - \Omega(\pi(s, \cdot))$$

**Regularized Optimal** Bellman operators w.r.t. $\Omega$

$$L^*_\Omega v(s) = \max_\pi L^\pi_\Omega v[s] = \Omega^*(q(s, \cdot))$$

**Greedy** policy

$$\pi' = G_\Omega(v) = \nabla \Omega^*(q) \iff L^\pi_\Omega v = L^*_\Omega v$$

We have the usual properties for $L^\pi_\Omega$: **affine, monotonicity, distributivity, contraction**
Regularized Markov Decision Processes

Regularized value functions:  \( v_{\pi}^\Omega = L_{\pi}^\Omega v_{\pi}^\Omega \)

\[
q_{\pi}(s, a) = r(s, a) + \gamma \sum_y p(y|s, a)v_{\pi}(y)
\]

\[
v_{\pi}(s) = \mathbb{E}_{a \sim \pi}[q_{\pi}(s, a)] - \Omega(\pi(s, \cdot))
\]

Regularized optimal value functions:  \( v_{\pi}^\star = L_{\pi}^\star v_{\pi}^\star \)

\[
q_{\pi}^\star(s, a) = r(s, a) + \gamma \sum_y p(y|s, a)v_{\pi}^\star(y)
\]

\[
v_{\pi}^\star(s) = \Omega^\star(q_{\pi}^\star(s, \cdot))
\]

Optimality

\( \pi_{\Omega}^\star = \mathcal{G}_\Omega(v_{\pi}^\star) \) is optimal

\( \forall \pi, \quad v_{\Omega}^{\pi^\star} = v_{\pi}^\star \geq v_{\Omega}^\pi \)
This explains many recent algorithms
They can be seen as a particular instance of Modified Policy Iteration

\[ \pi_{k+1} = \mathcal{G}_\Omega(v_k) \]
\[ v_{k+1} = (L^{\pi_{k+1}}_\Omega)^m v_k \]

- Up to modifications for make them practical
  - Soft Q-learning with negative entropy [Fox et al., 2016, Schulman et al., 2017a] or Tsallis entropy [Lee et al., 2018]
  - SAC with entropic regularizer [Haarnoja et al., 2018]
  - Algorithms based on path consistency [Nachum et al., 2017, Chow et al., 2018]
Regularized Markov Decision Processes

Issues:
- Regularization as defined above is changing the objective
- We obtain a different optimal policy
- Should be an algorithm trick and not a change in the objective
  - i.e., estimate the original optimal policy by solving
    a series of regularized problems
Issues:

- Regularization as defined above is changing the objective
- We obtain a different optimal policy
- Should be an algorithm trick and not a change in the objective
  - i.e., estimate the original optimal policy by solving
    a series of regularized problems

Solution:

- Consider a time varying regularized
- Penalize the difference between policy $\pi$ and the one at previous iteration (already seen)
Regularized Markov Decision Processes

*Bregman divergence*

\[
\Omega_{\pi_s'}(\pi_s) = D_{\Omega}(\pi_s \parallel \pi_s') = \Omega(\pi_s) - \Omega(\pi_s') - \nabla \Omega(\pi_s')^T(\pi_s - \pi_s')
\]

*Example:*

negative entropy \(\Rightarrow\) \(\Omega_{\pi_s'}(\pi_s) = D_{KL}(\pi \parallel \pi')[s]\)
Bregman divergence

\[ \Omega_{\pi'_s}(\pi_s) = D_\Omega(\pi_s \| \pi'_s) = \Omega(\pi_s) - \Omega(\pi'_s) - \nabla \Omega(\pi'_s) \top (\pi_s - \pi'_s) \]

**Example:**

negative entropy \( \Rightarrow \) \( \Omega_{\pi'_s}(\pi_s) = D_{KL}(\pi \| \pi')[s] \)

**Policy Iteration improvement**

\[ \pi_{k+1} = \mathcal{G}_{\Omega_{\pi_k}}(v_k) \]
\[ = \arg \max_{\pi} \sum_a \pi(s, a)q_k(s, a) - D_\Omega(\pi \| \pi_k) \]
Regularized Markov Decision Processes

Bregman divergence

\[ \Omega_{\pi'}(\pi_s) = D_\Omega(\pi_s \parallel \pi'_s) = \Omega(\pi_s) - \Omega(\pi'_s) - \nabla \Omega(\pi'_s)^T(\pi_s - \pi'_s) \]

Example:

negative entropy \(\implies\) \[ \Omega_{\pi'}(\pi_s) = D_{KL}(\pi \parallel \pi')[s] \]

Policy Iteration improvement

\[ \pi_{k+1} = G_{\Omega_{\pi_k}}(v_k) \]
\[ = \arg \max_{\pi} \sum_a \pi(s, a)q_k(s, a) - D_\Omega(\pi \parallel \pi_k) \]

\[ \text{similar to Mirror Descent in proximal form with } -q_k \text{ as gradient!} \]

\[ \implies \text{estimates the original optimal policy} \]
Regularized Markov Decision Processes

- Common framework
- Algorithms are either Mirror Descent or Dual Averaging [Neu et al., 2017]

TRPO can be seen as a mirror descent approach \( \implies \) guarantees of convergence
Similar interpretation (as dual averaging algorithm) for DPP [Azar et al., 2012] and MPO [Abdolmaleki et al., 2018].
Regularized Policy Gradient

\[
\nabla J_\Omega(\pi) = \sum_s d^\pi(s) \sum_a \pi(s, a) \left( q_\Omega^\pi(s, a) - \frac{\partial \Omega(\pi(s, \cdot))}{\partial \pi(s, a)} \right) \nabla \log \pi(s, a)
\]

Possible to replace with Bregman divergence \(\Rightarrow\) convergence to original policy
Resources

Reinforcement Learning

Books


Courses


Francis Bach. Stochastic optimization: Beyond stochastic gradients and convexity part i. 2016.


