

CMPT 419/983: Theoretical Foundations of Reinforcement Learning

Lecture 8

Sharan Vaswani

October 27, 2023

Recap

- Approximate policy iteration (API) aims to find an optimal policy without access to \mathcal{P} , r .
- API alternates between policy evaluation and policy improvement: at iteration k ,
 - **Policy Evaluation:** Compute the estimate \hat{q}^{π_k} (for example, using TD, Monte-Carlo).
 - **Policy Improvement:** $\forall s, \pi_{k+1}(s) = \arg \max_a \hat{q}^{\pi_k}(s, a)$.
- If the policy evaluation error at iteration k is controlled s.t. $\hat{q}^{\pi_k} = q^{\pi_k} + \epsilon_k$, then, API has the following convergence, $\|v^{\pi_{K+1}} - v^*\|_\infty \leq \gamma^K \|v^{\pi_0} - v^*\|_\infty + \frac{2 \max_{k \in \{0, \dots, K-1\}} \|\epsilon_k\|_\infty}{(1-\gamma)^2}$
- We have access to $\Phi \in \mathbb{R}^{SA \times d}$ s.t. for every π , there exists a θ^* such that, $\max_{(s,a)} |q^\pi(s, a) - \langle \theta^*, \phi(s, a) \rangle| \leq \epsilon_b$.
- In order to control the policy evaluation error,
 - Choose $\mathcal{C} \subset \mathcal{S} \times \mathcal{A}$, and for each $z := (s, a) \in \mathcal{C}$, rollout m trajectories (truncated to horizon H) and calculate $\hat{R}(z)$. We can ensure that $|\hat{R}(z) - q^\pi(z)| \leq \epsilon_\bullet$ w.p. $1 - \delta$ for all $z \in \mathcal{C}$.
 - Estimate $\hat{\theta} := \arg \min_\theta \frac{1}{2} \sum_{z \in \mathcal{C}} \zeta(z) \left[\langle \theta, \phi(z) \rangle - \hat{R}(z) \right]^2$.

Policy Evaluation for Approximate Policy Iteration

Claim: Assuming $V := \sum_{z \in \mathcal{C}} \zeta(z) \phi(z) \phi(z)^T \in \mathbb{R}^{d \times d}$ is invertible, for any $z \in \mathcal{S} \times \mathcal{A}$,

$$|q^\pi(z) - \langle \hat{\theta}, \phi(z) \rangle| \leq \varepsilon_{\mathbf{b}} + \|\phi(z)\|_{V^{-1}} [\varepsilon_{\mathbf{s}} + \varepsilon_{\mathbf{b}}]$$

Proof: Since $\hat{\theta}$ is computed by minimizing $\frac{1}{2} \sum_{z \in \mathcal{C}} \zeta(z) [\langle \theta, \phi(z) \rangle - \hat{R}(z)]^2$ and V is invertible,

$$\hat{\theta} = V^{-1} \left[\sum_{z' \in \mathcal{C}} \zeta(z') \hat{R}(z') \phi(z') \right]$$

$$\begin{aligned} |q^\pi(z) - \langle \hat{\theta}, \phi(z) \rangle| &= |q^\pi(z) - \langle \theta^*, \phi(z) \rangle + \langle \theta^*, \phi(z) \rangle - \langle \hat{\theta}, \phi(z) \rangle| \\ &\quad \text{(Add/subtract } \langle \theta^*, \phi(z) \rangle \text{)} \\ &\leq |q^\pi(z) - \langle \theta^*, \phi(z) \rangle| + |\langle \theta^*, \phi(z) \rangle - \langle \hat{\theta}, \phi(z) \rangle| \\ &\quad \text{(Triangle inequality)} \end{aligned}$$

$$\implies |q^\pi(z) - \langle \hat{\theta}, \phi(z) \rangle| \leq \varepsilon_{\mathbf{b}} + |\langle \theta^*, \phi(z) \rangle - \langle \hat{\theta}, \phi(z) \rangle|$$

We will now bound $|\langle \theta^*, \phi(z) \rangle - \langle \hat{\theta}, \phi(z) \rangle|$.

Policy Evaluation for Approximate Policy Iteration

For $z' \in \mathcal{C}$, define $\mathcal{E}(z') := \hat{R}(z') - \langle \theta^*, \phi(z') \rangle$. Hence,

$$\begin{aligned}\hat{\theta} &= V^{-1} \left[\sum_{z' \in \mathcal{C}} \zeta(z') [\langle \theta^*, \phi(z') \rangle + \mathcal{E}(z')] \phi(z') \right] \\ &= V^{-1} \left[\sum_{z' \in \mathcal{C}} \zeta(z') \phi(z') \phi(z')^T \right] \theta^* + V^{-1} \left[\sum_{z' \in \mathcal{C}} \zeta(z') \mathcal{E}(z') \phi(z') \right] \\ \implies \hat{\theta} - \theta^* &= V^{-1} \left[\sum_{z' \in \mathcal{C}} \zeta(z') \mathcal{E}(z') \phi(z') \right]\end{aligned}$$

Hence, for an arbitrary $z \in \mathcal{S} \times \mathcal{A}$,

$$\begin{aligned}|\langle \theta^*, \phi(z) \rangle - \langle \hat{\theta}, \phi(z) \rangle| &= \left| \left\langle V^{-1} \left[\sum_{z' \in \mathcal{C}} \zeta(z') \mathcal{E}(z') \phi(z') \right], \phi(z) \right\rangle \right| \\ &= \left| \left\langle \sum_{z' \in \mathcal{C}} \zeta(z') \mathcal{E}(z') V^{-1} \phi(z'), \phi(z) \right\rangle \right| = \left| \sum_{z' \in \mathcal{C}} \zeta(z') \mathcal{E}(z') \langle \phi(z), V^{-1} \phi(z') \rangle \right|\end{aligned}$$

Policy Evaluation for Approximate Policy Iteration

Recall that $|\langle \theta^*, \phi(z) \rangle - \langle \hat{\theta}, \phi(z) \rangle| = |\sum_{z' \in \mathcal{C}} \zeta(z') \mathcal{E}(z') \langle \phi(z), V^{-1} \phi(z') \rangle|$.

$$\begin{aligned} |\langle \theta^*, \phi(z) \rangle - \langle \hat{\theta}, \phi(z) \rangle| &\leq \sum_{z' \in \mathcal{C}} |\mathcal{E}(z')| \zeta(z') |\langle \phi(z), V^{-1} \phi(z') \rangle| \\ &\leq \left(\max_{z' \in \mathcal{C}} |\mathcal{E}(z')| \right) \sum_{z' \in \mathcal{C}} \zeta(z') |\langle \phi(z), V^{-1} \phi(z') \rangle| \end{aligned}$$

$$\sum_{z' \in \mathcal{C}} \zeta(z') |\langle \phi(z), V^{-1} \phi(z') \rangle| = \sqrt{(\mathbb{E}_{z' \sim \zeta} |\langle \phi(z), V^{-1} \phi(z') \rangle|)^2} \stackrel{\text{Jensen}}{\leq} \sqrt{\mathbb{E}_{z'} |\langle \phi(z), V^{-1} \phi(z') \rangle|^2}$$

$$= \sqrt{\mathbb{E}_{z'} [\phi(z)^T V^{-1} \phi(z') \phi(z')^T V^{-1} \phi(z)]} = \sqrt{\phi(z)^T V^{-1} \left[\sum_{z'} \zeta(z') \phi(z') \phi(z')^T \right] V^{-1} \phi(z)}$$

$$\implies \sum_{z' \in \mathcal{C}} \zeta(z') |\langle \phi(z), V^{-1} \phi(z') \rangle| = \sqrt{\phi(z)^T V^{-1} \phi(z)} = \|\phi(z)\|_{V^{-1}}$$

$$\implies |\langle \theta^*, \phi(z) \rangle - \langle \hat{\theta}, \phi(z) \rangle| \leq \max_{z' \in \mathcal{C}} |\mathcal{E}(z')| \|\phi(z)\|_{V^{-1}}$$

Policy Evaluation for Approximate Policy Iteration

Recall that $|\langle \theta^*, \phi(z) \rangle - \langle \hat{\theta}, \phi(z) \rangle| \leq \max_{z' \in \mathcal{C}} |\mathcal{E}(z')| \|\phi(z)\|_{V^{-1}}$. Bounding $\max_{z' \in \mathcal{C}} |\mathcal{E}(z')|$,

$$\begin{aligned} |\mathcal{E}(z')| &= |\hat{R}(z) - \langle \theta^*, \phi(z) \rangle| = |\hat{R}(z) - q^\pi(z) + q^\pi(z) - \langle \theta^*, \phi(z) \rangle| \\ &\quad \text{(Add/subtract } q^\pi(z)) \\ &\leq |\hat{R}(z) - q^\pi(z)| + |q^\pi(z) - \langle \theta^*, \phi(z) \rangle| \quad \text{(Triangle inequality)} \\ &\leq \varepsilon_{\mathbf{s}} + \varepsilon_{\mathbf{b}} \end{aligned}$$

$$\implies |\langle \theta^*, \phi(z) \rangle - \langle \hat{\theta}, \phi(z) \rangle| \leq [\varepsilon_{\mathbf{s}} + \varepsilon_{\mathbf{b}}] \|\phi(z)\|_{V^{-1}}$$

Putting everything together,

$$|q^\pi(z) - \langle \hat{\theta}, \phi(z) \rangle| \leq \varepsilon_{\mathbf{b}} + [\varepsilon_{\mathbf{s}} + \varepsilon_{\mathbf{b}}] \|\phi(z)\|_{V^{-1}}$$

Hence, in order to control the generalization error, we have to control $\|\phi(z)\|_{V^{-1}}$, while controlling the size of \mathcal{C} .

Policy Evaluation for Approximate Policy Iteration

Kiefer-Wolfowitz Theorem: There exists a $\mathcal{C} \subset \mathcal{S} \times \mathcal{A}$ and a distribution $\zeta \in \Delta_{|\mathcal{C}|}$ such that for $V := \sum_{z \in \mathcal{C}} \zeta(z) \phi(z) \phi(z)^T \in \mathbb{R}^{d \times d}$,

$$\sup_{z \in \mathcal{S} \times \mathcal{A}} \|\phi(z)\|_{V^{-1}} \leq \sqrt{d} \quad ; \quad |\mathcal{C}| \leq \frac{d(d+1)}{2}$$

- Intuitively, this means that we can find a *coreset* of feature vectors that captures most of the information in Φ . Finding such a coreset is referred to as *G-optimal design* in statistics.
- \mathcal{C} and ζ can be approximately computed using a greedy algorithm that has access to Φ (Need to do this in Assignment 3!)

Combining the Kiefer-Wolfowitz theorem with our previous result gives,

$$|q^\pi(z) - \hat{q}^\pi(z)| = |q^\pi(z) - \langle \hat{\theta}, \phi(z) \rangle| \leq \varepsilon_b + \sqrt{d} [\varepsilon_s + \varepsilon_b] = \varepsilon_b (1 + \sqrt{d}) + \varepsilon_s \sqrt{d}$$

- Note that the \sqrt{d} amplification in the error is tight.
- Algorithmically, we need to run Monte-Carlo estimation from $O(d^2)$ (s, a) pairs, and we can estimate $q^\pi(s, a)$ upto an $\varepsilon_b (1 + \sqrt{d}) + \varepsilon_s \sqrt{d}$ error for all (s, a) pairs.

Convergence of Approximate Policy Iteration

We have seen the following results:

$$\|v^{\pi_{k+1}} - v^*\|_\infty \leq \gamma^K \|v^{\pi_0} - v^*\|_\infty + \frac{2 \max_{k \in \{0, \dots, K-1\}} \|\epsilon_k\|_\infty}{(1 - \gamma)^2}$$

$$|q^\pi(s, a) - \hat{q}^\pi(s, a)| \leq \varepsilon_b \left(1 + \sqrt{d}\right) + \varepsilon_s \sqrt{d} \quad (\text{for all } \pi \text{ and } (s, a) \text{ pairs})$$

$$\implies \|v^{\pi_{k+1}} - v^*\|_\infty \leq \gamma^K \|v^{\pi_0} - v^*\|_\infty + \frac{2\varepsilon_b \left(1 + \sqrt{d}\right) + 2\varepsilon_s \sqrt{d}}{(1 - \gamma)^2}$$

- If the q functions are exactly in the span of Φ , $\varepsilon_b = 0$. For example, in the *tabular* setting where $d = S$ and the features are one hot vectors, the error depends on $\sqrt{S} \varepsilon_s$.
- The algorithm for constructing \mathcal{C} requires iterating through the states, and this can be inefficient. [YHAY⁺22] considers an online algorithm that does not require global access to the full Φ matrix, but has similar theoretical guarantees.
- Next, we will see an alternative algorithm – Politex that has slower convergence $[O(1/\sqrt{K})]$, but smaller error amplification $[O(1/(1 - \gamma))]$.

Politex

- Like policy iteration, Politex alternates between evaluating the policy and updating it.
- Unlike policy iteration that uses a max over actions, Politex uses a softmax (multiplicative weights) to update the policy. This makes the resulting algorithm less aggressive.

Algorithm Politex

- 1: **Input:** MDP $M = (\mathcal{S}, \mathcal{A}, \rho)$, π_0 , step-size η
 - 2: **for** $k = 0 \rightarrow K - 1$ **do**
 - 3: **Policy Evaluation:** Compute the estimate $\hat{q}_k := \hat{q}^{\pi_k}$ (for example, using TD, Monte-Carlo) and define $\bar{q}_k = \sum_{i=0}^k \hat{q}_i$
 - 4: **Policy Update:** $\forall (s, a), \pi_{k+1}(a|s) = \frac{\exp(\eta \bar{q}_k(s, a))}{\sum_{a'} \exp(\eta \bar{q}_k(s, a'))}$.
 - 5: **end for**
 - 6: Return the *mixture policy* $\bar{\pi}_K := \frac{\sum_{k=0}^{K-1} \pi_k}{K}$
-

- Politex returns the *mixture policy* $\bar{\pi}_K$ which corresponds to choosing a policy in $\{\pi_k\}_{k=0}^{K-1}$ uniformly at random.
- If $\bar{q}_k = \hat{q}_k$, Politex recovers policy iteration as $\eta \rightarrow \infty$ (Prove in Assignment 3!)

Convergence of Politex

Claim: If the policy evaluation error at iteration k is controlled s.t. $\hat{q}^k = q^{\pi_k} + \epsilon_k$, then Politex has the following convergence,

$$\|v^{\bar{\pi}_K} - v^*\|_\infty \leq \frac{\|\text{Regret}(K)\|_\infty}{(1-\gamma)K} + \frac{2 \max_{k \in \{0, \dots, K-1\}} \|\epsilon_k\|_\infty}{(1-\gamma)},$$

where $\text{Regret}(K) = \sum_{k=0}^{K-1} [\mathcal{M}_{\pi^*} \hat{q}_k - \mathcal{M}_{\pi_k} \hat{q}_k] \in \mathbb{R}^S$ is the regret incurred by Politex on an online linear optimization problem for each state $s \in \mathcal{S}$.

- The error amplification only depends on $1/(1-\gamma)$, and thus Politex has a better dependence on ϵ compared to approximate policy iteration.
- Compared to policy iteration that has an γ^K convergence, the convergence for Politex depends on $\frac{\text{Regret}(K)}{K}$. We will show that $\text{Regret}(K) = O(\sqrt{K})$, and hence, the Politex achieves the slower $O(1/\sqrt{K})$ convergence.
- The above claim does not depend on the specific update rule of Politex, and can be used to prove convergence for alternative algorithms that have sublinear regret.

Convergence of Politex

$$\text{Proof: } v^{\pi^*} - v^{\pi_k} = (I - \gamma \mathbf{P}_{\pi^*})^{-1} [\mathcal{T}_{\pi^*} v^{\pi_k} - v^{\pi_k}] \quad (\text{Value difference lemma})$$

Summing up from $k = 0$ to $k = K - 1$ and dividing by K ,

$$v^{\pi^*} - \frac{\sum_{k=0}^{K-1} v^{\pi_k}}{K} = \frac{1}{K} (I - \gamma \mathbf{P}_{\pi^*})^{-1} \sum_{k=0}^{K-1} [\mathcal{T}_{\pi^*} v^{\pi_k} - v^{\pi_k}]$$

$$\implies v^{\pi^*} - v^{\bar{\pi}_K} = (I - \gamma \mathbf{P}_{\pi^*})^{-1} \sum_{k=0}^{K-1} [\mathcal{T}_{\pi^*} v^{\pi_k} - v^{\pi_k}] = \frac{1}{K} (I - \gamma \mathbf{P}_{\pi^*})^{-1} \sum_{k=0}^{K-1} [\mathcal{T}_{\pi^*} v^{\pi_k} - \mathcal{T}_{\pi_k} v^{\pi_k}]$$

(Since $v^{\bar{\pi}_K} = \frac{\sum_{k=0}^{K-1} v^{\pi_k}}{K}$ (Prove in Assignment 3!) and $v^{\pi} = \mathcal{T}_{\pi} v^{\pi}$)

$$= \frac{1}{K} (I - \gamma \mathbf{P}_{\pi^*})^{-1} \sum_{k=0}^{K-1} [\mathcal{M}_{\pi^*} q^{\pi_k} - \mathcal{M}_{\pi_k} q^{\pi_k}] \quad (\text{Since } \mathcal{T}_{\pi} v = \mathcal{M}_{\pi} [r + \gamma \mathbb{P} v] = \mathcal{M}_{\pi} q)$$

$$= \frac{1}{K} (I - \gamma \mathbf{P}_{\pi^*})^{-1} \sum_{k=0}^{K-1} [\mathcal{M}_{\pi^*} \hat{q}_k - \mathcal{M}_{\pi_k} \hat{q}_k] + \frac{1}{K} (I - \gamma \mathbf{P}_{\pi^*})^{-1} \sum_{k=0}^{K-1} \left[(\mathcal{M}_{\pi_k} - \mathcal{M}_{\pi^*}) \underbrace{(\hat{q}_k - q^{\pi_k})}_{=\epsilon_k} \right]$$

Convergence of Politex

$$v^{\pi^*} - v^{\bar{\pi}_K} = \frac{1}{K} (I - \gamma \mathbf{P}_{\pi^*})^{-1} \sum_{k=0}^{K-1} [\mathcal{M}_{\pi^*} \hat{q}_k - \mathcal{M}_{\pi_k} \hat{q}_k] + \frac{1}{K} (I - \gamma \mathbf{P}_{\pi^*})^{-1} \sum_{k=0}^{K-1} [(\mathcal{M}_{\pi_k} - \mathcal{M}_{\pi^*}) \epsilon_k]$$

Using the definition of $\text{Regret}(K)$ and taking norms,

$$\begin{aligned} \|v^{\pi^*} - v^{\bar{\pi}_K}\|_{\infty} &= \left\| \frac{1}{K} (I - \gamma \mathbf{P}_{\pi^*})^{-1} \text{Regret}(K) + \frac{1}{K} (I - \gamma \mathbf{P}_{\pi^*})^{-1} \sum_{k=0}^{K-1} [(\mathcal{M}_{\pi_k} - \mathcal{M}_{\pi^*}) \epsilon_k] \right\|_{\infty} \\ &\leq \frac{1}{K} \|(I - \gamma \mathbf{P}_{\pi^*})^{-1} \text{Regret}(K)\|_{\infty} + \frac{1}{K} \left\| (I - \gamma \mathbf{P}_{\pi^*})^{-1} \sum_{k=0}^{K-1} [(\mathcal{M}_{\pi_k} - \mathcal{M}_{\pi^*}) \epsilon_k] \right\|_{\infty} \end{aligned}$$

(Triangle inequality)

$$\leq \frac{\|\text{Regret}(K)\|_{\infty}}{K(1-\gamma)} + \frac{1}{K(1-\gamma)} \left\| \sum_{k=0}^{K-1} [(\mathcal{M}_{\pi_k} - \mathcal{M}_{\pi^*}) \epsilon_k] \right\|_{\infty}$$

(Neumann series)

$$\leq \frac{\|\text{Regret}(K)\|_{\infty}}{K(1-\gamma)} + \frac{1}{K(1-\gamma)} \sum_{k=0}^{K-1} [\|\mathcal{M}_{\pi_k} \epsilon_k\|_{\infty} + \|\mathcal{M}_{\pi^*} \epsilon_k\|_{\infty}]$$

(Triangle inequality)

$$\leq \frac{\|\text{Regret}(K)\|_{\infty}}{K(1-\gamma)} + \frac{2 \max_{k \in \{0, \dots, K-1\}} \|\epsilon_k\|_{\infty}}{(1-\gamma)} \quad \square$$

(\mathcal{M}_{π} is non-expansive)

Convergence of Politex

Our aim now is to control $\|\text{Regret}(K)\|_\infty$ where $\text{Regret}(K) = \sum_{k=0}^{K-1} [\mathcal{M}_{\pi^*} \hat{q}_k - \mathcal{M}_{\pi_k} \hat{q}_k]$. By definition, for an arbitrary vector $u \in \mathbb{R}^{S \times A}$, $(\mathcal{M}_\pi u)(s) = \sum_a \pi(a|s) u(s, a)$. Hence,

$$\|\text{Regret}(K)\|_\infty = \max_s \left\| \sum_{k=0}^{K-1} \left[\sum_a \pi^*(a|s) \hat{q}_k(s, a) - \sum_a \pi_k(a|s) \hat{q}_k(s, a) \right] \right\|$$

Define $R_K(\pi^*, s) := \sum_{k=0}^{K-1} \langle \pi^*(\cdot|s), \hat{q}_k(s, \cdot) \rangle - \langle \pi_k(\cdot|s), \hat{q}_k(s, \cdot) \rangle$

$$\implies \|\text{Regret}(K)\|_\infty = \max_s |R_K(\pi^*, s)|$$

To bound $R_K(\pi^*, s)$, we will cast Politex as an online linear optimization for each state $s \in \mathcal{S}$:

- In each iteration $k \in [K]$, Politex chooses a distribution $\pi_k(\cdot|s) \in \Delta_A$ for each state s .
- The “environment” chooses and reveals the vector $\hat{q}_k(s, \cdot) \in \mathbb{R}^A$ and Politex receives a reward $\langle \pi_k(\cdot|s), \hat{q}_k(s, \cdot) \rangle$.
- The aim is to do as well as the optimal policy π^* that receives a reward $\langle \pi^*(\cdot|s), \hat{q}_k(s, \cdot) \rangle$

Digression – Online Optimization

Online Optimization

- 1: **Input:** w_0 , Algorithm \mathcal{A} , Convex set \mathcal{W}
 - 2: **for** $k = 0, \dots, K - 1$ **do**
 - 3: Algorithm \mathcal{A} chooses point (decision) $w_k \in \mathcal{W}$
 - 4: Environment chooses and reveals the (potentially adversarial) function $f_k : \mathcal{W} \rightarrow \mathbb{R}$
 - 5: Algorithm receives a reward $f_k(w_k)$
 - 6: **end for**
-

Application: Prediction from Expert Advice – Given n experts,
 $\mathcal{W} = \Delta_n = \{w_i | w_i \geq 0 ; \sum_{i=1}^n w_i = 1\}$ and $f_k(w_k) = \langle c_k, w_k \rangle$ where c_k is the reward vector.

Application: Imitation Learning – Given access to an expert that knows what action $a \in [A]$ to take in each state $s \in \mathcal{S}$, learn a policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$ that imitates the expert, i.e. we want that $\pi(a|s) \approx \pi_{\text{expert}}(a|s)$. Here, $w = \pi$ and $\mathcal{W} = \Delta_A \times \Delta_A \dots \Delta_A$ (simplex for each state) and f_k is a measure of the (negative) discrepancy between π_k and π_{expert} .

Q: What is w , \mathcal{W} , f_k for Politex (for state s)? **Ans:** $\pi(\cdot|s)$, Δ_A , $\langle \pi(\cdot|s), \hat{q}_k(s, \cdot) \rangle$

Digression – Online Optimization

Recall that the sequence of functions $\{f_k\}_{k=0}^{K-1}$ is potentially adversarial and can depend on w_k .

Objective: Do well against the *best fixed decision in hindsight*, i.e. if we knew the entire sequence of functions beforehand, we would choose $w^* := \arg \max_{w \in \mathcal{W}} \sum_{k=0}^{K-1} f_k(w)$.

Regret: $R_K(w^*) := \sum_{k=0}^{K-1} [f_k(w^*) - f_k(w_k)]$

We want to design algorithms that achieve a *sublinear regret* (that grows as $o(T)$). A sublinear regret implies that the performance of our sequence of decisions is approaching that of w^* .

Q: What is “best” decision we want to compare against in PoliteX (for state s)? **Ans:** $\pi^*(\cdot|s)$

Hence, bounding $R_K(\pi^*, s)$ for PoliteX is equivalent to bounding the regret for a sequence of linear functions of the form: $f_k(w) = \langle g_k, w \rangle$.

Digression – Online Optimization

The simplest algorithm that results in sublinear regret is *Online Gradient Ascent*.

Online Gradient Ascent: At iteration k , the algorithm chooses the point w_k . After the function f_k is revealed, the algorithm receives a reward $f_k(w_k)$ and uses it to compute

$$w_{k+1} = \Pi_{\mathcal{W}}[w_k + \eta_k \nabla f_k(w_k)]$$

where $\Pi_{\mathcal{W}}[x] = \arg \min_{y \in \mathcal{W}} \frac{1}{2} \|y - x\|_2^2$ is the Euclidean projection onto \mathcal{W} .

The Online Gradient Ascent update at iteration k can also be written as:

$$w_{k+1} = \arg \max_{w \in \mathcal{W}} \left[\langle \nabla f_k(w_k), w \rangle - \frac{1}{2\eta_k} \|w - w_k\|_2^2 \right]$$

In other words, gradient ascent moves in the direction of the gradient $\nabla f_k(w_k)$, while remaining “close” (in the Euclidean norm) to the previous iterate w_k .

Instead of using the Euclidean norm, we could measure the distance to w_k differently.

Digression – Online Optimization

- Online Mirror Ascent generalizes gradient ascent by choosing a strictly convex, differentiable function $\psi : \mathbb{R}^d \rightarrow \mathbb{R}$ to induce a distance measure. ψ is referred to as the *mirror map*.
- ψ induces the *Bregman divergence* $D_\psi(\cdot, \cdot)$, a distance measure between points x, y ,

$$D_\psi(y, x) := \psi(y) - \psi(x) - \langle \nabla \psi(x), y - x \rangle .$$

Geometrically, $D_\psi(y, x)$ is the distance between the function $\psi(y)$ and the line $\psi(x) + \langle \nabla \psi(x), y - x \rangle$ which is tangent to the function at x .

Using this distance measure results in the mirror ascent update:

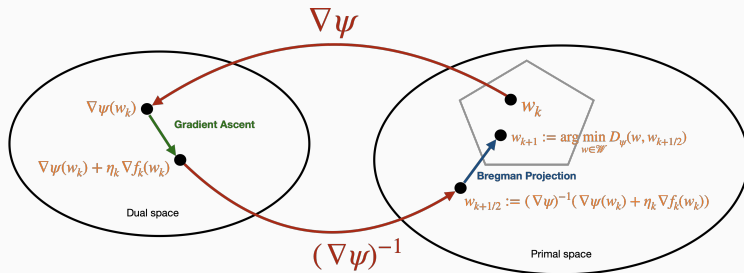
$$w_{k+1} = \arg \max_{w \in \mathcal{W}} \left[\langle \nabla f_k(w_k), w \rangle - \frac{1}{\eta_k} D_\psi(w, w_k) \right]$$

- Setting $\psi(x) = \frac{1}{2} \|x\|^2$ results in $D_\psi(y, x) = \frac{1}{2} \|y - x\|^2$ and recovers gradient ascent.

Digression – Online Optimization

The mirror ascent update can be equivalently written as:

$$w_{k+1/2} = (\nabla\psi)^{-1}(\nabla\psi(w_k) + \eta_k \nabla f_k(w_k)) ; \quad w_{k+1} = \arg \min_{w \in \mathcal{W}} D_\psi(w, w_{k+1/2})$$



Prove in Assignment 3!

Digression – Online Optimization

In order to analyze mirror ascent, we will make some assumptions on f_k and ψ .

- We will assume that $\{f_k\}_{k=0}^{K-1}$ are linear i.e. for some vector g_k , $f_k(w) = \langle g_k, w \rangle$. We will also assume that $\{f_k\}_{k=0}^{K-1}$ are G -Lipschitz continuous.

Lipschitz continuous functions: f is *Lipschitz continuous* iff f can not change arbitrarily fast meaning that its gradient is bounded. Formally, for any $w \in \mathcal{W}$,

$$\|\nabla f(w)\|_{\infty} \leq G$$


where G is the Lipschitz constant.

- We will assume that ψ is ν strongly-convex.

Strongly-convex functions: If f is differentiable, it is ν -strongly convex iff its domain \mathcal{D} is a convex set and for all $x, y \in \mathcal{D}$ and $\nu > 0$,

$$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle + \frac{\nu}{2} \|y - x\|_1^2$$

i.e. for all y , the function is lower-bounded by the quadratic defined in the RHS.

-  Dong Yin, Botao Hao, Yasin Abbasi-Yadkori, Nevena Lazić, and Csaba Szepesvári, *Efficient local planning with linear function approximation*, International Conference on Algorithmic Learning Theory, PMLR, 2022, pp. 1165–1192.