Homework 1

Yu-Chieh Kuo B07611039[†]

[†]Department of Information Management, National Taiwan University

Problem 1: Regret of Multi-round Game

1.(a)

Suppose you are attending an exam. All questions in this exam are multi choices with equal or more than one correct answer. You receive 1 point (-1 loss) if your answer is completely correct, -1 point (1 loss) if your answer is completely incorrect, and a partial point between 1 and -1 (partial loss between -1 and 1) if your answer is partially correct. You are permitted to observe the answer for question t after you submit your answer to question t and suffer a corresponding loss. There exists some possible best answer strategies to all questions; for example, the exam maker has some preference on the number of the correct answer, says 3, for each question. Your goal is to minimize the difference between cumulative loss (score) and the loss with that best answer strategy, which can be formulated as the online learning problem represented in the homework.

1.(b)

Given a sequence of vector $\omega_t(\gamma_t) \in [-1,1]^{\mathcal{K}}$ and the *regret* defined by

$$R_T := \sum_{t=1}^{T} \omega_t(\gamma_t) - \min_{k \in [\mathcal{K}]} \sum_{t=1}^{T} \omega_t(k),$$

we consider a linear monotone mapping $\mathcal{L} \colon [-1,1]^{\mathcal{K}} \to [0,1]^{\mathcal{K}}$ to modify the vector value non-negative, specified, $\mathcal{L}(\omega_t(k)) = \frac{\omega_t(k)+1}{2}$ $\forall k \in [\mathcal{K}]$. We then define a new vector $\omega'(k) := \mathcal{L}(\omega_t(k)) \in [0,1]^{\mathcal{K}}$ $\forall k \in [\mathcal{K}]$. [1]

Given an *average* regret with $\omega'_t(\cdot)$

$$\frac{1}{T} R_T := \frac{1}{T} \sum_{t=1}^{T} \omega'_t(\gamma_t) - \min_{k \in [\mathcal{K}]} \frac{1}{T} \sum_{t=1}^{T} \omega'_t(k),$$

which can be re-written as

$$\max_{k \in [\mathcal{K}]} \left(\frac{1}{T} \sum_{t=1}^{T} \omega'_t(\gamma_t) - \frac{1}{T} \sum_{t=1}^{T} \omega'_t(k) \right),$$

[1]Yu-Chieh thanks to TA Chung-En Tsai's tremendous help!

we define a vector-valued payoff function by

$$\omega'_T := \sum_{t=1}^{T} \left(\omega'_t(\gamma_t) - \omega'_t(a_1), \omega'_t(\gamma_t) - \omega'_t(a_2), \cdots, \omega'_t(\gamma_t) - \omega'_t(a_{\mathcal{K}})\right),$$

which is the difference between loss at round t and loss for each action in Learner's action set $[K]$. As a side note, $\omega'_T \in \mathbb{R}^K$ and $k \in \{a_1, \cdots, a_{\mathcal{K}}\}$. The average vector-valued payoff function is followingly defined by

$$\bar{\omega}'_T := \frac{1}{T}\sum_{t=1}^{T} \left(\omega'_t(\gamma_t) - \omega'_t(a_1), \omega'_t(\gamma_t) - \omega'_t(a_2), \cdots, \omega'_t(\gamma_t) - \omega'_t(a_{\mathcal{K}})\right).$$

Here we notice that the elements of each row of $\bar{\omega}'_T$ should be non-positive as we can replace γ_t by a_i, $a_i \in \mathcal{K}$ to obtain a better action. Hence, we then set a set $\mathcal{S} = \{\mathbf{s} \in \mathbb{R}^{\mathcal{K}} \mid -1 \le s_i \le 0 \ \forall i = 1, \cdots, \mathcal{K}\}$ such that

$$\mathcal{P}\left(\lim_{T\to\infty} \operatorname{dist}(\bar{\omega}_T, \mathcal{S}) = 0\right) = 1,$$

which is formulated as a Blackwell's approachability problem. In addition, $\frac{R_T}{T} \to 0$ if the elements of each row of $\bar{\omega}'_T$ are non-positive, i.e., R_T is no-regret as $T \to \infty$.

The algorithm with input $\omega_t(\gamma_t)$ is also no-regret. Suppose there exists an algorithm $\mathscr{A}$ achieves $o(T)$ when the vectors are in $[0,1]^{\mathcal{K}}$. The algorithm with vectors in $[-1,1]^{\mathcal{K}}$ also achieves $o(T)$ since

$$\sum_{t=1}^{T} \omega_t(\gamma_t) - \min_{k\in[\mathcal{K}]} \sum_{t=1}^{T} \omega_t(k) = 2\overbrace{\left(\sum_{t=1}^{T} \omega'_t(\gamma_t) - \min_{k\in[\mathcal{K}]} \sum_{t=1}^{T} \omega'_t(k)\right)}^{\text{Regret of } \mathscr{A}} = o(T).$$

1.(c)

We now check the approachability of set $\mathcal{S}$. Here I use ω_t instead of ω'_t to denote a sequence of vectors in $[-1,1]^{\mathcal{K}}$ to ease note. Based on the result in Problem 1(b), both results in $(\omega_t)_{t\in\mathbb{N}} \in [-1,1]$ and $(\omega_t)_{t\in\mathbb{N}} \in [0,1]$ achieve $o(T)$.

First, $\mathcal{S}$ is closed and convex. For all $(\omega_t)_{t\in\mathbb{N}} \in [0,1]^{\mathcal{K}}$, we can always find an action $\gamma_t \in [\mathcal{K}]$ such that $\omega_t\gamma_t$ is non-positive by definition. Hence, $\mathcal{S}$ is $response-satisfiable$ (see Definition 4 in Abernethy et al. (2011)) and approachable (see Theorem 6 in Abernethy et al. (2011)). In addition, Blackwell's algorithm specifies Learner randomly announces its action for some distribution over its action set; that is, $\Delta[\mathcal{K}]$. Therefore, Blackwell's condition is satisfied, and Blackwell's theorem states that there exists a randomized strategy such that $\operatorname{dist}(\bar{\omega}_T, \mathcal{S}) \le \frac{2R}{\sqrt{T}}$ with $\|s\| \le R \ \forall s \in \mathcal{S}$. That is,

$$\mathcal{P}\left(\lim_{T\to\infty} \operatorname{dist}(\bar{\omega}_T, \mathcal{S}) = 0\right) = 1 \quad \text{and} \quad R_T = o(T) \ \text{ a.s.}$$

(I found another algorithm for this problem, but I am unsure it is an approachability algorithm(?). I write the algorithm in a more approachability style above, but I still write this algorithm below.)

We demonstrate a randomized algorithm[2] called Randomized Weighted Majority (or Hedge). The algorithm was proposed by Freund and Schapire (1997), and Daniel and Andreas (2010) and Thomas (2018) provided detailed explanations of the algorithm analysis[3].

The Randomized Weighted Majority (MWR) algorithm is described as Algorithm 1.

Algorithm 1 The Randomized Weighted Majority Algorithm.

```
1: Initialize: Set $e_{k,0} = 1$ for each $k \in [\mathcal{K}]$.
2: for each round $t = 1, 2, \cdots, T$ do
3:     Define a distribution $p_t(k) := \frac{e_{i,k}}{\sum_{k\in[\mathcal{K}]} e_{k,t}}$.
4:     LEARNER chooses $\gamma_t$ with the distribution $p_t(k)$ among action set $[\mathcal{K}]$.
5:     LEARNER suffers a loss $\omega_t(\gamma_t)$.
6:     for each $k \in [\mathcal{K}]$ do
7:         Set $e_{k,t+1} = e_{k,t}(1-\eta)^{\omega_t(k)}$
8:     end for
9: end for
```

Line 4 of Algorithm 1 represents the randomness of the algorithm since LEARNER chooses an action from a dynamically updating distribution among all actions in $[\mathcal{K}]$. Based on the result in Problem 1(b), we can prove the MWR algorithm achieves $o(T)$ with the loss in $[0,1]$, which will also guarantee the same result for loss in $[-1,1]$. Below I use $\omega_t(\cdot)$ instead of $\omega_t'(\cdot)$ to ease note.

Theorem 1. For the sequence from $[0,1]$ and the action $k \in [\mathcal{K}]$, $\mathcal{K} \in \mathbb{N}$,

$$\Omega_T^{\text{ALG}} \leq (1+\eta)\sum_{t=1}^{T} \omega_t(\gamma_t) + \frac{\ln \mathcal{K}}{\eta},$$

where

$$\Omega_T^{\text{ALG}} = \sum_{t=1}^{T} \sum_{k\in[\mathcal{K}]} p_t(k)\omega_t(\gamma_t)$$

denotes by the expected sum of loss of the algorithm.

Proof. We first define the sum of weight at round t by $W_t = \sum_{k\in[\mathcal{K}]} e_{k,t}$ for all $k \in [\mathcal{K}]$. W_t decreases over time since

$$W_{t+1} = \sum_{k\in[\mathcal{K}]} e_{k,t+1} = \sum_{k\in[\mathcal{K}]} e_{k,t}(1-\eta)^{\omega_t(k)}.$$

Note that $(1-\eta)^z = 1 - z\eta$ for $z = \{0,1\}$, and $z \mapsto (1-\eta)^z$ is convex w.r.t. z. Hence, for $z \in [0,1]$, $(1-\eta)^z \leq 1 - z\eta$ holds, and it gives

$$W_{t+1} \leq \sum_{k\in[\mathcal{K}]} e_{k,t}(1-\eta\omega_t(k)) = W_t - \eta \sum_{k\in[\mathcal{K}]} e_{k,t}\omega_t(k).$$

[2] I initially wrote down this algorithm but found it seemed not to be a desired approachability algorithm. Please ignore it if it is not. Thank you so much!

[3] I am surprised that these lectures in the reference were taught more than 10 years ago, but there were no related theoretical machine learning courses until Prof. Li returned to NTU. I feel the recruiting committee, or NTU, or academia in Taiwan do not emphasize on the theoretical research in ML, which makes me a little bit upset and depressed, and I also feel Taiwan is behind the world for this case.

Denote by the expected loss of the algorithm in round t $\omega_t^{\mathrm{ALG}} = \frac{1}{W_t}\sum_{k\in[K]} e_{k,t}\omega_t(k)$, the bound for W_{t+1} alters to

$$W_{t+1} \le W_t - \eta\omega_t^{\mathrm{ALG}}W_t = W_t(1-\eta\omega_t^{\mathrm{ALG}}).$$

As a consequence,

$$W_{T+1} \le W_1 \prod_t^T (1-\eta\omega_t^{\mathrm{ALG}}) = \mathcal{K}\prod_t^T(1-\eta\omega_t^{\mathrm{ALG}}).$$

Now we obtain the upper bound of the sum of weight after round t, we would like to obtain a lower bound in the same terms of the expected loss:

$$W_{T+1} \ge e_{k,T+1} = e_{k,1}\prod_{t=1}^T (1-\eta)^{\omega_t(k)} = (1-\eta)^{\sum_{t=1}^T \omega_t(k)}.$$

Combining the upper and lower bounds and taking the logaritm on both sides gives

$$\sum_{t=1}^T \omega_t(k)\ln(1-\eta) \le \ln\mathcal{K} + \sum_{t=1}^T (1-\eta\omega_t^{\mathrm{ALG}})$$

Here we apply an approximation of the logaritm:

$$-z - z^2 \le \ln(1-z) \le -z.$$

Thus, the inequality alters to

$$\sum_{t=1}^T \omega_t(k)(-\eta-\eta^2) \le \ln\mathcal{K} - \eta\Omega_t^{\mathrm{ALG}} \iff \Omega_T^{\mathrm{ALG}} \le (1+\eta)\sum_{t=1}^T \omega_t(\gamma_t) + \frac{\ln\mathcal{K}}{\eta},$$

□

Now, setting $\eta = \sqrt{\frac{\ln\mathcal{K}}{T}}$ yields

$$\Omega_T^{\mathrm{ALG}} \le \min_{k\in[\mathcal{K}]}\sum_{t=1}^T \omega_t(k) + 2\sqrt{T\ln\mathcal{K}} \iff \overbrace{\Omega_T^{\mathrm{ALG}} - \min_{k\in[\mathcal{K}]}\sum_{t=1}^T \omega_t(k)}^{\text{Regret}} \le 2\sqrt{T\ln\mathcal{K}} = o(T).$$

Note that the bound of regret *must* hold, i.e., with probability 1. Therefore, we show that there exists a randomized algorithm that achieves $o(T)$ almost surely.

Problem 2: Online Linear Optimization

2.(a)

Suppose there exists $\gamma \sim \Delta([\mathcal{K}])$ minimizing $\sum_{t=1}^T \langle\omega_t,\gamma\rangle$, we consider a simple case $\gamma = \alpha k_1 + (1-\alpha)k_2$, $0<\alpha<1$, $k_1, k_2 \in [\mathcal{K}]$ W.L.O.G. to represent a mixed strategy. Moreover, we assume

$$\sum_{t=1}^T \langle\omega_t,\gamma\rangle < \sum_{t=1}^T \langle\omega_t,k_1\rangle \le \sum_{t=1}^T \langle\omega_t,k_2\rangle.$$

for convenience. It gives

$$\begin{aligned}
\sum_{t=1}^{T} \langle \omega_t, \gamma \rangle &= \sum_{t=1}^{T} \langle \omega_t, \alpha k_1 + (1-\alpha k_2) \rangle \\
&= \sum_{t=1}^{T} \langle \omega_t, \alpha k_1 \rangle + \sum_{t=1}^{T} \langle \omega_t, (1-\alpha) k_2 \rangle \\
&= \alpha \sum_{t=1}^{T} \langle \omega_t, k_1 \rangle + (1-\alpha) \sum_{t=1}^{T} \langle \omega_t, k_2 \rangle \\
&\geq \alpha \sum_{t=1}^{T} \langle \omega_t, k_1 \rangle + (1-\alpha) \sum_{t=1}^{T} \langle \omega_t, k_1 \rangle .
\end{aligned}$$

If the case of equality holds, then $\alpha k_1 + (1-\alpha)k_2 = k_1 \iff \alpha = 1$, which leads to that γ is a pure strategy (i.e., one-hot vector in $\Delta([\mathcal{K}])$); on the other hand, if the case of equality doesn't hold, then $\alpha k_1 + (1-\alpha)k_2 > k_1$ implies k_1 is the minimizer instead of γ, a contradiction.

2.(b)

Problem 3: Calibration

3.(a)

We first introduce *calibration*. Suppose Learner announces a forecast to predict the weather each day. Here we only care about whethet it rains or not. Denote by p the chance that it rains the next day. Now, from the subsequence of days on which Learner makes a prediction by announcing p, Learner computes the empirical frequency that it actually rained the next day and denote by $q(p)$ the empirical frequency. Learner would like to see $q(p) = p$ (well calibrated, Dawid (1982)) or at least $q(p) \approx p$; that is, the prediction is accurate enough to fit the outcome revealed by Reality. The explanation is modified by Foster (1998) and Raj (2022).

$\varepsilon -$ *calibration* is a method to measure whether the error between Learner's prediction strategy and the outcome from Reality is small enough (less than ε) as the number of forecast processes increases to a large number (infinity). The formal definition, modified by Foster (1998), Kakade and Foster (2008), and Mannor and Stoltz (2009), is described as Definition 1.

Definition 1. (ε-calibrated algorithm) We first introduce a calibration. Given a forcast algorithm $\mathscr{A}$ generating a forecast p_t at each round t, the set of the probability distribution over outcome $\mathcal{X}$ announced by Reality $\Delta(\mathcal{X})$, and the Dirac probability distribution δ_{x_t} on some outcome $x_t \in \mathcal{X}$, the goal of Learner calibration, is that for all strategies of Reality,

$$\forall \varepsilon > 0, \forall p \in \Delta(\mathcal{X}), \quad \lim_{T\to\infty} \left\| \frac{1}{T} \sum_{t=1}^{T} \mathbb{1}_{\{\|p_t - p\| \leq \varepsilon\}} (p_t - \delta_{x_t}) \right\| = 0 \quad \text{a.s.}$$

Next, we extend the calibration to ε-calibration. Given $\varepsilon > 0$ and some finite covering of $\Delta(\mathcal{X})$ by N balls $\mathcal{B}(\varepsilon)$ with radius ε, we denote by $q_1, \cdots, q_N$ the centers of all $\mathcal{B}(\varepsilon)$ in the covering, and Learner will only choose $p_t \in \{q_1, \cdots, q_N\}$. In addition, denote by K_t the index in $\{1, \cdots, N\}$ such that $p_t = q_{K_t}$. Hence, an ε-calibrated algorithm is said that for all strategies of Reality,

$$\limsup_{T\to\infty} \sum_{k=1}^{N} \left\| \frac{1}{T} \sum_{t=1}^{T} \mathbb{1}_{\{K_t = k\}} \left(q_k - \delta_{x_t}\right) \right\| \leq \varepsilon \quad \text{a.s.}$$

3.(b)

Oakes (1985) first proved that a deterministic calibrated forecasting algorithm doesn't exists after Dawid (1982) left a conjecture for the existence of self-calibrating distribution. Foster (1998) gives an easy-to-understand explanation. Suppose Learner uses some *deterministic* forecasting algorithm $\mathscr{A}$. Consider Reality generates the outcome sequence $\mathcal{X}$ by the following procedure:

$$x_t = \begin{cases} 1 & \text{if in round } t \text{ the Learner announces } p_t \leq 0.5 \\ 0 & \text{otherwise.} \end{cases}$$

It shows $\frac{1}{T}\sum_{t=1}^{T} \mathbb{1}_{\{\|p_t - p\| \leq \varepsilon\}} \left(p_t - \delta_{x_t}\right) \geq \frac{1}{4}$ for *all* deterministic forecasting algorithms $\mathscr{A}$. In addition, the case of equality occurs when $\mathscr{A}$ generates $p_t = \frac{1}{2}$ for all t. Hence, $\frac{1}{T}\sum_{t=1}^{T} \mathbb{1}_{\{\|p_t - p\| \leq \varepsilon\}} \left(p_t - \delta_{x_t}\right)$ cannot achieve 0 as $T \to \infty$, i.e., a calibrated algorithm must be randomized.

3.(c)

To formulate the problem of designing an ε-calibrated forecasting algorithm for Learner as an approachability problem, we must find some vector-valued utility function $u(\cdot,\cdot)$ and an *approachable* set to be approached.

We first denote by d the dimension of the utility vector and let $C \subset \mathbb{R}^d$, the problem for Learner to solve is to ensure the average of its vector-valued utility converges to the set C almost surely, i.e.,

$$\mathcal{P}\left(\liminf_{T\to\infty} \inf_{c\in C} \left\| c - \frac{1}{T}\sum_{t=1}^{T} u\left(p_t, x_t\right) \right\| = 0 \right) = 1.$$

Here we specify the vector-valued utility function $u\colon \mathcal{I} \times \mathcal{J} \to \mathbb{R}^d$, $\mathcal{I}$ and $\mathcal{J}$ are action sets for Learner and Reality, as

$$u(k,a) = \left(\underline{0}, \cdots, \underline{0}, q_k - \delta_{x_t}, \underline{0}, \cdots, \underline{0}\right) \quad \forall k \in \{1, \cdots, N\},\ x_t \in \mathcal{X},$$

which is a one-hot vector in $\mathbb{R}^{XN}$ with non-zero element located in the k-th row and given by $q_k - \delta_{x_t}$, where X is the cardinality of $\mathcal{X}$, and $\underline{0}$ is the zero element in R^X.

Moreover, we define the set C to be approached as below. Re-write XN-dim vectors of R^{XN} as N-dim vectors with components in R^X, i.e.,

$$\underline{\underline{y}} = \left(\underline{y}_1, \cdots, \underline{y}_N\right) \quad \forall \underline{\underline{y}} \in \mathbb{R}^{XN},$$

where $\underline{y}_k \in \mathbb{R}^X\ \forall k \in \{1, \cdots, N\}$. Then, the approached set C is defined by

$$C = \left\{ \underline{\underline{y}} \in \mathbb{R}^{XN} \colon \sum_{k=1}^{N} \left\| \underline{y}_k \right\| \leq \varepsilon \right\}.$$

Note that C is convex and closed.

3.(d)

Let $p \in \Delta(\mathcal{X})$, there exists $k \in \{1, \cdots, N\}$ such that $\|q_k - p\| \le \varepsilon$ and thus $u(k,p) \in C$. Hence, we state

$$\forall p \in \Delta(\mathcal{X}),\ \exists \in \{1, \cdots, N\} \quad \text{s.t.} \quad u(k,p) \in C.$$

Therefore, C is proven to be approachable.

3.(e)

I read Lin et al. (2020) to survey several algorithms[4] applied for solving the min-max problem and decide to demonstrate their MINIMAX-APPA (Accelerated Proximal Point Algorithm). This algorithm can be regarded as a combination of INEXACT-APPA and MAXIMIN-AGA-AGD (Accelerated Gradient Ascent; Accelerated Gradient Descent), and I describe the AGD and MAXIMIN-AGA-AGD as Algorithm 2 and Algorithm 3 since we use them in the MINIMAX-APPA. Before illustrating the algorithm, let us clarify additional notations and the scenario of MINIMAX-APPA. We denote by the diameters of set

$$D_p = \max_{p,p' \in \Delta_{m-1}} \|p - p'\| \quad \text{and} \quad D_q = \max_{q,q' \in \Delta_{n-1}} \|q - q'\|,$$

projections onto Δ_{m-1} and Δ_{n-1} P_p and P_q, and denote $f(p,q) := \langle p, Aq \rangle$ for convenience. The function f is assumed to be ℓ-smooth and μ_p-strongly-convex-μ_y-strongly-concave over Δ_{m-1} and Δ_{n-1}. Hence, the MINIMAX-APPA is performed as Algorithm 4.

Algorithm 2 AGD

```
1: Input: $f$, initial point $p_0$, smoothness $\ell$, strongly-convex module $\mu_p$, and tolerance $\varepsilon$.
2: Initialize: $t \leftarrow 0$, $\tilde{p}_0 \leftarrow p_0$, $\eta \leftarrow \frac{1}{\ell}$, and $\theta \leftarrow \frac{\sqrt{\kappa_p}-1}{\sqrt{\kappa_p}+1}$.
3: repeat
4:     $t \leftarrow t + 1$.
5:     $p_t \leftarrow P_p(\tilde{p}_{t-1} - \eta \nabla_p f(\tilde{p}_{t-1}))$.
6:     $\tilde{p}_t \leftarrow p_t + \theta(p_t - p_{t-1})$.
7: until $\|p_t - P_p(p_t - \eta \nabla_q f(p_t))\|^2 \le \frac{\varepsilon}{2\kappa_p^2(\ell - \mu_p)}$.
8: $p^\star \leftarrow P_p(p_t - \eta \nabla_p f(p_t, \cdot))$.
```

The MINIMAX-APPA states *the total number of gradient evaluations* (gradient complexity or iterations complexity in other pieces of literature) as its performance guarantee. The total number of gradient evaluations is bounded by

$$\mathcal{O}\left(\sqrt{\kappa_p \kappa_q} \log^3 \left(\frac{(\kappa_p + \kappa_q)\ell(D_p^2 + D_q^2)}{\varepsilon} \right) \right) \in \tilde{\mathcal{O}}\left(\sqrt{\kappa_p \kappa_q} \right),$$

where $\kappa_p = \frac{\ell}{\mu_p}$ and $\kappa_q = \frac{\ell}{\mu_q}$ are condition numbers. Note that Algorithm 4 finds $p^\star$ in the for loop and $q^\star$ after completing the for loop. However, the performance guarantee doesn't change if we only require $p^\star$.

[4]I learn many learning algorithms and computational complexity measures during the survey. It's quite exciting to explore the world node by node, paper by paper.

Algorithm 3 MAXIMIN-AGA-AGD

```
1: Input: $f$, initial point $p_0$, $q_0$, smoothness $\ell$, strongly-convex module $\mu_p$, $\mu_q$, and tolerance $\varepsilon$.
2: Initialize: $t \leftarrow 0$, $\tilde{p}_0 \leftarrow p_0$, $\eta \leftarrow \frac{1}{2\kappa_p \ell}$, $\kappa_p \leftarrow \frac{\ell}{\mu_p}$, $\kappa_q \leftarrow \frac{\ell}{\mu_q}$, $\theta \leftarrow \frac{4\sqrt{\kappa_p\kappa_q}-1}{4\sqrt{\kappa_p\kappa_q}+1}$, and $\tilde{\varepsilon} \leftarrow \frac{\varepsilon}{(10\kappa_p\kappa_q)^7}$.
3: repeat
4:     $t \leftarrow t+1$.
5:     $\tilde{p}_{t-1} \leftarrow \mathrm{AGD}(f(\cdot, \tilde{q}_{t-1}, p_0, \ell, \mu_p, \tilde{\varepsilon}))$.
6:     $q_t \leftarrow P_q(\tilde{q}_{t-1} + \eta \nabla_q f(\tilde{p}_{t-1}, \tilde{q}_t - 1))$.
7:     $\tilde{q}_t \leftarrow q_t + \theta(q_t - q_{t-1})$.
8:     $p_t \leftarrow \mathrm{AGD}(f(\cdot, q_t, p_0, \ell, \mu_p, \tilde{\varepsilon}))$.
9: until $\left\| q_t - P_q(q_t + \eta \nabla_q f(p_t, q_t)) \right\|^2 \le \frac{\varepsilon}{(10\kappa_p\kappa_q)^4 \ell}$.
10: $p^\star \leftarrow P_p(p_t - \frac{1}{2\kappa_q \ell} \nabla_p f(p_t, q_t))$.
```

Algorithm 4 MINIMAX-APPA

```
1: Input: $f$, initial point $p_0$, $q_0$, proximity $\ell$, strongly-convex parameter $\mu$, tolerance $\varepsilon$, and iteration $T$.
2: Initialize: $\tilde{p}_0 \leftarrow p_0$, $\kappa_x \leftarrow \frac{\ell}{\mu_p}$, $\theta \leftarrow \frac{2\sqrt{\kappa_p}-1}{2\sqrt{\kappa_p}+1}$, $\delta \leftarrow \frac{\varepsilon}{(10\kappa_p\kappa_q)^4}$, and $\tilde{\varepsilon} \leftarrow \frac{\varepsilon}{10^2\kappa_p\kappa_q}$.
3: for each $t = 1, 2, \cdots, T$ do
4:     Denote $g_t(p,q) := f(p,q) + \ell \left\| p - \tilde{p}_{t-1} \right\|^2$.
5:     $p_t \leftarrow$ MAXIMIN-AGA-AGD$(g_t, p_0, q_0, 3\ell, 2\ell, \mu_q, \delta)$.
6:     $\tilde{p}_t \leftarrow p_t + \theta(p_t - p_{t-1})$.
7: end for
8: $\tilde{q} \leftarrow \mathrm{AGD}(-f(p_T, \cdot), q_0, \ell, \mu_q, \tilde{\varepsilon})$.
9: $q_T \leftarrow P_q(\tilde{q} + \frac{1}{2\kappa_p \ell} \nabla_q f(p_T, \tilde{q}))$.
10: $p^\star \leftarrow p_T, q^\star \leftarrow q_T$.
```

References

Abernethy, Jacob, Peter L. Bartlett, and Elad Hazan (2011) "Blackwell Approachability and No-Regret Learning are Equivalent," in *Proceedings of the 24th Annual Conference on Learning Theory*, 19 of Proceedings of Machine Learning Research, 27–46, 09–11 Jun.

Daniel, Golovin and Krause Andreas (2010) "Lecture notes in CS/CNS/EE 253: Advanced Topics in Machine Learning," Janurary.

Dawid, A. P. (1982) "The Well-Calibrated Bayesian," *Journal of the American Statistical Association*, 77 (379), 605–610.

Foster, Dean P. (1998) "Asymptotic calibration," *Biometrika*, 85 (2), 379–390.

Freund, Yoav and Robert E Schapire (1997) "A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting," *Journal of Computer and System Sciences*, 55 (1), 119–139.

Kakade, Sham M. and Dean P. Foster (2008) "Deterministic calibration and Nash equilibrium," *Journal of Computer and System Sciences*, 74 (1), 115–130.

Lin, Tianyi, Chi Jin, and Michael. I. Jordan (2020) "Near-Optimal Algorithms for Minimax Optimization," in *Proceedings of the 33rd Annual Conference on Learning Theory*, 125 of Proceedings of Machine Learning Research, 1–42.

Mannor, Shie and Gilles Stoltz (2009) "A Geometric Proof of Calibration."

Oakes, David (1985) "Self-Calibrating Priors Do Not Exist," *Journal of the American Statistical Association*, 80 (390), 339–339.

Raj, Sangani (2022) "A Comprehensive Guide on Model Calibration: What, When, and How," September.

Thomas, Kesselheim (2018) "Lecture notes in Algorithms and Uncertainty," December.