Homework 0

Yu-Chieh Kuo B07611039†

†Department of Information Management, National Taiwan University

Problem 1

The paper I selected is A Parameter-free Hedging Algorithm, which was authored by Kamalika Chaudhuri, Yoav Freund, and Daniel Hsu from UC San Diego. This paper was accepted by Part of Advances in Neural Information Processing Systems 22 (NIPS 2009) in 2009. I put Chaudhuri et al. (2009a) in the reference.

Problem 2

I found several interesting topics and problems when surveying the accepted papers in ALT and COLT in recent years. However, those papers raised my attention tended to be either too complicated or beyond the problem setting in this homework. They might focus on the more general or complicated form of online learning problems. After randomly searching and reading the references from papers, I found Chaudhuri et al. (2009a), which seemed more readable and understandable to me. In addition, the title "Parameter-free" caught my eyes. I was curious about what and how parameters-free is in this paper and the world of online learning; That's why I finally selected this paper.

Nevertheless, the process of searching papers illustrated the world of online learning and theoretical machine learning to me, which was hugely helpful. I appreciate this kind of assignment design.

Problem 3

Although the description in Chaudhuri et al. (2009a) is different with which in assignment, the setting remains similar. In the setting of Chaudhuri et al. (2009a), it does not specify the sequence $(\omega_t)_{t\in\mathbb{N}}$ revealed by Reality, but it assumes that the loss of the prediction announced by Learner lies in an interval of length 1, such as $[0,1]$ or $[-1/2,1/2]$. I think the paper does not impose the assumption of $(\omega_t)_{t\in\mathbb{N}}$.

Problem 4

Before describing the algorithm, let me specify the setting first. Learner is given access to a set of N actions (the choice of prediction) with $N \geq 2$ in each round t. Learner can choose a mixed strategy with the distribution $p_t = (p_{1,t}, \cdots, p_{N,t})$ over N actions instead of announcing one certain prediction. Each action incurs a loss $\ell_{i,t}$, and LEARNER then suffers the expected loss over this distribution (says A)

$$\ell_{A,t} = \sum_{i=1}^{N} p_{i,t}\ell_{i,t}.$$

The regret of LEARNER to an action i in round t is defined as $r_{i,t} = \ell_{A,t} - \ell_{i,t}$, and LEARNER's cumulative regret to an action i in the first t round is $R_{i,t} = \sum_{\tau=1}^{t} r_{i,\tau}$. Thus, the goal of LEARNER is to minimize $R_{i,t}$ to any action i at any round t.

The algorithm called Normal-Hedge Algorithm is proposed as below:

Algorithm 1 The Normal-Hedge algorithm.

1: Initialize: Set $R_{i,0} \leftarrow 0, p_{i,1} \leftarrow \frac{1}{N}$ for each i.
2: **for** each $t = 1, 2, \cdots, T$ **do**
3: Action i incurs loss $\ell_{i,t}$.
4: LEARNER incurs loss $\ell_{A,t} = \sum_{i=1}^{N} p_{i,t}\ell_{i,t}$.
5: Update culmulative regrets as $R_{i,t} \leftarrow R_{i,t-1} + (\ell_{A,t} - \ell_{i,t})$ for each i
6: Find $c_t > 0$ satisfying $\frac{1}{N}\sum_{i=1}^{N} \exp \frac{([R_{i,t}]_+)^2}{2c_t} = e$.
7: Update the distribution for round $t+1$ as $p_{i,t+1} \propto \frac{([R_{i,t}]_+)}{c_t} \exp \frac{([R_{i,t}]_+)^2}{2c_t}$ for each i.
8: **end for**

Here $[\cdot]_+$ denotes $\max\{0, \cdot\}$. Note that the line 6 of the Algorithm 1 takes $\mathcal{O}(N)$ since the function

$$\phi(x, c) \equiv \exp \frac{([R_{i,t}]_+)^2}{2c_t} \quad \forall\, x \in \mathbb{R},\ c > 0$$

is convex in $c > 0$, we can use a line search to determine c_t such that

$$\frac{1}{N}\sum_{i=1}^{N} \exp \frac{([R_{i,t}]_+)^2}{2c_t} = e.$$

Hence, the time complexity of Algorithm 1 is $\mathcal{O}(T \cdot C)$, where C denotes the time complexity of line search, which various with the different numerical optimization methods.

In my opinion, Algorithm 1 is efficiently implementable. The standard Hedge, iteratively defining the learning rate η as a function of the size of actions N and rounds T, faces the difficulty of setting η as N is large. Some pieces of literature suggesting an identical η or running multiple Hedge simultaneously achieve poor performance or are impractical for real applications. In contrast, the Normal-Hedge does not need to determine the learning rate η, but involves a line search to update the distribution over actions.

Problem 5

The performance measure of the Normal-Hedge Algorithm in Chaudhuri et al. (2009a) does not follow the standard measure to calculate the regret to the best action, since there might be a lot of actions that are close to the action with the lowest loss when N is large. In this case, measuring performance with respect to a small group of actions that perform well is more reasonable than only measuring the best actions. Thus, Chaudhuri et al. (2009a) uses the top ε-quantile of actions as their performance measure. The theoretical guarantee of the Normal-Hedge Algorithm suggests that the regret to the top ε-quantile of actions is at most

$$\mathcal{O}\left(\sqrt{T\ln\frac{1}{\varepsilon}}+\ln^2 N\right),$$

which holds simultaneously for all T and ε. If we set $\varepsilon=\frac{1}{N}$, the upper-bound of the regret to the best action becomes

$$\mathcal{O}\left(\sqrt{T\ln N}+\ln^2 N\right),$$

which is only slightly worse than the bound achieved by Hedge with optimally-tuned parameters.

Problem 6

The most significant improvement of Chaudhuri et al. (2009a) can be seen in the problem when the actions set is large, such as the tracking problem. I quote the explanation of the tracking problem in Chaudhuri et al. (2009b):

> We study the tracking problem, namely, estimating the hidden state of an object over time, from unreliable and noisy measurements. ... We study the tracking problem, which has numerous applications in AI, control and finance.

In finance, portfolio management and optimization are essential questions for researchers. In the online learning setting, actions and strategy distribution can be regarded as the portfolio and the investment strategy over the portfolio, and the portfolio's return is the negative loss value. For example, Zhang et al. (2020) use deep learning models to optimize portfolio allocation. Ban et al. (2018) and Conlon et al. (2021) are other two examples in this field.

Problem 7

The Normal-Hedge Algorithm in Chaudhuri et al. (2009a) is limited in discrete time; however, the theoretical guarantee might not be satisfactory, or can be improved in the continuous-time setting. For example, in control and finance, we might want to set the time stamp as an arbitrary small to announce more accurate predictions. Another incentive to develop the continuous-time setting is to utilize powerful analytical tools from stochastic calculus to allow for simpler analysis. Freund (2009) is the first work to extend Chaudhuri et al. (2009a) to the continuous-time setting, and Portella et al. (2022) is one of the recent pieces of work.

References

Ban, Gah-Yi, Noureddine El Karoui, and Andrew E. B. Lim (2018) "Machine Learning and Portfolio Optimization," *Management Science*, 64 (3), 1136–1154, 10.1287/mnsc.2016.2644.

Chaudhuri, Kamalika, Yoav Freund, and Daniel J Hsu (2009a) "A Parameter-free Hedging Algorithm," in Bengio, Y., D. Schuurmans, J. Lafferty, C. Williams, and A. Culotta eds. *Advances in Neural Information Processing Systems*, 22: Curran

Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2009/file/2cbca44843a864533ec05b321ae1f9d1-Paper.pdf.

Chaudhuri, Kamalika, Yoav Freund, and Daniel J. Hsu (2009b) "Tracking using bexplanation-based modeling," *CoRR*, abs/0903.2862, http://arxiv.org/abs/0903.2862.

Conlon, Thomas, John Cotter, and Iason Kynigakis (2021) "Machine Learning and Factor-Based Portfolio Optimization," Working Papers 202111, Geary Institute, University College Dublin, https://ideas.repec.org/p/ucd/wpaper/202111.html.

Freund, Yoav (2009) "A method for Hedging in continuous time."

Portella, Victor Sanches, Christopher Liaw, and Nicholas J. A. Harvey (2022) "Continuous Prediction with Experts' Advice."

Zhang, Zihao, Stefan Zohren, and Stephen Roberts (2020) "Deep Learning for Portfolio Optimization," *The Journal of Financial Data Science*, 2 (4), 8–20, 10.3905/jfds.2020.1.042.