Implicit Adaptation to Low Rank Structure in Online Learning

Weiqi Yang and Michael Spece

***Abstract*—This paper is about the relationship between regret (in online learning) and the rank of an ensemble's loss matrix *Y*. Recently, several new algorithms have been developed to exploit low rank structure in Y. Unfortunately, each of these is not known to be order minimax optimal outside of specialized settings. This paper explores through simulation whether this apparent difficulty in achieving minimax optimality is because highly specialized algorithms are required. We observe that a horizon-adaptive hedge algorithm appears to exploit low rank structure effectively, suggesting that algorithms do not have to explicitly work to exploit low rank structure.**

***Index Terms*—Experts, online learning, regret, simulation study.**

I. Introduction

Prediction with expert advice is a fundamental problem in online learning. There are T rounds with N experts. In each round $t = 1 \cdots T$, the learner chooses a probability vector $p_t \in \Delta_N$, where Δ_N denotes the N-simplex, namely the set of all distributions over N experts

$$\Delta_N = \left\{ x \in R^N : \forall i, x(i) \geq 0 \wedge \sum_{i=1}^{N} x(i) = 1 \right\}.$$

Then Nature chooses a loss vector $y_t \in [0,1]^N$, and the learner suffers a loss $p_t(y_t) = p_t \cdot y_t$ The regret is defined as follows:

$$\text{Reg}(N,T) = \sum_{t=1}^{T} p_t \cdot y_t - \min_{k\in\{1,\ldots,N\}} \sum_{t=1}^{T} l_t(i).$$

Intuitively if an expert has the greatest overall performance for previous rounds, the player can trust this expert more and follow its advice. This strategy is known as Follow the Leader and can be interpreted as a form empirical risk minimization.

As described in [1], in online learning, there are two types of settings: stochastic and adversarial. In the former, y_t is generated according to a fixed IID distribution and the learner seeks to control the expected regret. In the latter, y_t can be fully dependent on the history of play. It is the latter so-called adversarial setting that is less well understood and the focus of this paper.

A. Algorithms

Many algorithms handle adversarial data, including Follow the Regularized Leader (FTRL), Exponentially Weighted Averaging (EWA, also known as Hedge), and Online Mirror Descent (OMD) (though the latter two can be considered as special cases of the former).

EWA is named and defined in [2]. Its fundamental idea is to weight experts based on their performance. More precisely, if an expert has relatively high loss, it will receive an exponentially decreasing proportion of weight. The update rule is as follow:

$$x_{j,t+1} = \frac{w_{j,t} e^{-\eta l(\hat{y}_{j,t}, y_t)}}{\sum_{i=1}^{N} w_{i,t} e^{-\eta l(\hat{y}_{i,t}, y_t)}}.$$

This algorithm can be modified in many ways. One of these modifications is called the doubling trick, in which the learning rate η becomes adaptive over exponentially increasing time epochs [2]. Consequently, T need not be known in advance. We consider another horizon-adaptive algorithm in Section B.

FTRL modifies Follow the Leader by adding a regularization term $R(x)$ ([3]), which can provide stability to the algorithm. In this case, it can help ensure the algorithm does not overfit. The standard form for FTRL applied to prediction with expert's advice problem is as follow:

$$x_t = argmin_{x_t \in \Delta_N} \sum_{t=1}^{t-1} l(\hat{y}_t, y_t) + R(x)$$

OMD is a transformation of FTRL. And its update rule is naturally interpreted in terms of a gradient operation.

Other algorithm variants include lazy updating (e.g. Hazan, 2016).

B. Online Learning with Low Rank Experts

In more recent studies, researchers have started to focus on some parameters with smaller values in order to make the algorithm more efficient, instead of depending on the number of experts. Particularly, [3] proposes the Low-Rank Experts setting, in which they assume that there is a low dimensional r ($r << N$) embedding of the loss matrix. This idea springs from the popular technique which is called matrix factorization proposed by [4]. In [4], the researchers assume that a large rating matrix can be projected into a lower dimensional embedding.

Low rank assumptions are ubiquitous in data science; one of the most famous techniques is matrix completion proposed by [5] and extended by [6].

[3] proposes a type of OMD: if the embedding is known in advance, then this type can achieve a regret upper bound of $8\sqrt{rT}$. They also show that under the stochastic case, Follow the Leader achieves a bound in terms of the ε-rank of the loss

Manuscript received January 5, 2020; revised December 7, 2020.

Weiqi Yang is with University of Science and Technology of China, China (e-mail: 976237481@qq.com).

Michael Spece is with Carnegie Mellon University, USA.

matrix. The ε-rank is defined as

$$r_\varepsilon(Y) := \min\{rank(Y') : ||Y' - Y||_\infty < \varepsilon\}.$$

They proved a regret bound related to the ε-rank under the stochastic case, but not adversarial case. Then [3] proposes the open problem of whether there is an upper bound of $O(\sqrt{rT})$ under the adversarial setting.

Ref. [7] proposes Online Lazy Newton. This algorithm achieves a regret bound of $\sqrt{rT\log(T)}$; the authors show that their method is invariant to affine transformations.

Ref. [8] treats the problem from a different perspective. Under their setup, they consider an infinite number of experts, and they change EWA. Nevertheless, when they apply their algorithm to the low rank experts problem, they do not get a bound in term of the rank; instead, they address another open problem under the stochastic setting, under a ε-rank for the embedding.

Ref. [9] treats the problem with a wholly new prospective. They assume that the loss matrix structure in hindsight is an additive space composed of low rank spaces and other spaces. Under their setup and noisy low rank experts, they achieve a regret bound of $\sqrt{2(16r + \varepsilon)T}$. But, even under their setup, this bound is suboptimal.

Ref. [10] studies low rank online learning in the supervised setting for real-valued predictions, and achieves a regret of $O(\sqrt{rT})$ for a fixed ambient dimension. However, that paper fails to prove that this bound can apply to the setting in [3].

C. Restart Approaches

These have been studied in various fields in online learning for years. One of the most famous applications is the adaptive regret for fixed shares given by [11]. It uses the restart approach for prediction under mix-loss by implementing the Follow the Leading History presented by [12].

Ref. [13] applies the restart approach to EWA under branching experts.

Ref. [8] also applies the restart approach to EWA so that both number of effective experts and the step parameter can be set dynamically.

Ref. [1] provides the foregoing review of low rank learning and finds various settings, including sub-settings of [3], where $O(\sqrt{rT})$ is achievable.

II. Learning Algorithms Under Study

The first algorithm (Algorithm 1) we consider is a horizon-adaptive version of EWA, which also appears in [4]:

Algorithm 1: Traditional learning algorithm

```
Data: Input $N$,$T$, loss matrix $Y$ (the $t$ th column of $Y$ is called $y^t$,the $k$ th rows and the $t$ th column of $Y$ is called $y_k^t$)
Result: Get the regret
1. $p_1^1 = 1$ $p_k^1 = 0 (k = 2, \dots, N)$
2. $p_k^t = \frac{e^{-\eta \sum_{t'=1}^{t-1} y_k^{t'}}}{\sum_{k'=1}^{N} e^{-\eta \sum_{t'=1}^{t-1} y_{k'}^{t'}}} (\eta = \sqrt{\frac{\log(N)}{t} \times 8}, t = 2, \dots, T, k = 1, \dots, N)$
3. $\mathrm{Reg}(N,T) = \sum_{t=1}^{T} p^t \cdot y^t - \min_{k \in \{1,\dots,N\}} \sum_{t=1}^{T} y_k^t$
```

The second algorithm (Algorithm 2) is new and combines the idea of the previous algorithm with a restart approach of [12]:

Algorithm 2: **Another learning algorithm**

```
Data: Input $N$,$T$, loss matrix $Y$ (the $t$ th column of $Y$ is called $y^t$,the $k$ th rows and the $t$ th column of $Y$ is called $y_k^t$)
Result: Get the regret
1. Cumulativeloss$(k,t) = \sum_{t'=1}^{T} y_k^{t'}$; $S = \{1\}$
2. Time $t$ from 1 to $T$; and for each certain time $t$: if the $p_k^t$'s cumulative loss does not equal to any $p_i^t$'s cumulative loss ($i \in S, k \in \{1 \dots N\}$), then put the $k$ into $S$.
3. $p_k^t = \frac{e^{-\eta \sum_{t'=1}^{t-1} y_k^{t'}}}{\sum_{k' \in S} e^{-\eta \sum_{t'=1}^{t-1} y_{k'}^{t'}}} (\eta = \sqrt{\frac{\log(N)}{t} \times 8}, k \in S)$
   $p_k^t = 0, k \notin S$
4. $\mathrm{Reg}(N,T) = \sum_{t=1}^{T} p^t \cdot y^t - \min_{k \in \{1,\dots,N\}} \sum_{t=1}^{T} y_k^t$
```

III. Simulation

In the simulations of the loss, each vector component follows a distribution D.

In the following experiments, we fix $N = 20$ $T = 20$ if we do not especially mention.

A. How to Generate the Loss Matrix

Algorithm 3: How to generate the loss matrix, which has *N* rows and *T* columns of a not high rank *r* (compared with the min(*N*, *T*))

```
Data: Input a distribution $D$ and a certain rank $r$
Result: Generate the loss matrix of a not high rank $r$ (compared with the min($N$, $T$))
1. Generate three matrices: $A, B, C$. $A$ is a matrix of $N$ rows and $T$ columns, and its each vector component obeys a $D$ distribution.
   $B$ is a matrix of $N$ rows and $N$ columns, and its each vector component obeys a $D$ distribution.
   $C$ is a matrix of $N$ rows and $N$ columns, and its first $r$ diagonal numbers are $1$, other numbers are $0$.
2. $Y = B \times C \times A$. And if $Y$'s element is greater than 1, make it equal to 1; if $Y$'s element is smaller than 0, make it equal to 0.
3. Make a judgment: if the $Y'$ s rank equals to $r$, then use it as the loss matrix; if not, abandon it.
```

But using Algorithm 3 to generate the matrices of higher rank r is not efficient. Because for higher rank r (compared with the $\min(N, T)$): C is close to the identity matrix of $\min(N, T)$ dimensions. So Y is close to the $B \times A$, and in this condition, it is very easy for its elements to be bigger than 1. Then according to Algorithm 3, the Y will be close to a matrix, whose most elements are 1 and rank is lower than expected. Therefore, with a certain number of experiments, only a very small percentage of them meet the requirement that their ranks equel to r. As a result, it is very hard to get close to the real worst regrets of high rank r in Algorithm 3.

So we consider another way to generate the loss matrices (Algorithm 4):

```
Algorithm 4: How to generate the loss matrix, which has $N$ rows and T columns of a certain rank $r$ (high)
Data: Input a distribution $D$ and a certain rank $r$
1. Generate a matrix: $\boldsymbol{Y}$. And $\boldsymbol{Y}$ is a matrix of $\boldsymbol{N}$ rows and $\boldsymbol{T}$ columns,and its each element obeys a $\boldsymbol{D}$ distribution.
2. Make a judgment: if the $\boldsymbol{Y}'$ s rank equals to $\boldsymbol{r}$, then use it as the loss matrix; if not, abandon it.
```

Algorithm 4 also has its disadvantage: when using this way to generate the matrices of low rank r (compared with the $\min(N, T)$), it is very hard to get the matrices that we want. Because, the matrix of rank r must have $(N-r)$ linear dependent rows and $(T-r)$ linear dependent columns. When N, T are much bigger than r, this can be very hard in Algorithm 4.

Therefore, in the following experiments, we use the Algorithm 3 to generate the matrices of relatively low rank r and the Algorithm 4 to generate the matrices of relatively high rank r. (when $N = T = 20$, we think r is relatively high if $r \geq 18$); (when $N = T = 50$, we think r is relatively high if $r \geq 42$)

B. $N = T = 20$ and Distribution= D Binomial Distribution ($p = 0.5$)

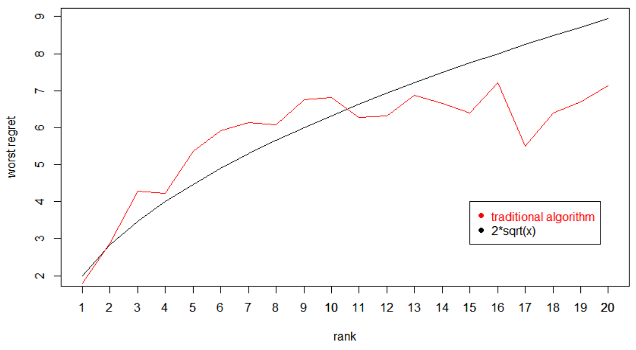

Fig. 1. 100000 trials of binomial distribution (p = 0.5), $N = 20, T = 20$.

The Fig. 1 is the picture describing the relationship between the worst regret and the loss matrices' rank r. And in Experiment B, we will use traditional learning algorithm (Algorithm 1) to compute the regrets.

In Experiment B, we fix the $N = T = 20$, and we use Algorithm 3 with the distribution D = binomial distribution (p = 0.5) to generate the loss matrices of certain rank r, when $r \leq 17$; and use Algorithm 4 with the distribution D = binomial distribution (p = 0.5) to generate the loss matrices of rank r, when $r \geq 18$. For each certain rank r ($r \in \{1 \ldots \min(N, T)\}$), we do 100000 independent trials.

From the Fig. 1, we can find three results:

1. The worst regret can decrease when the rank increases.

2. Although the curve fluctuates at some points, yet in general, the curve is of the same shape as the curve $y = 2\sqrt{x}$.

3. The curve has a rapid decline when the rank begins to be relatively high.

About the Result 1, it was expected that the worst regret will keep increasing when the rank increases. But our finding is inconsistent with that. To find if our finding is just because the number of trial is not big enough to get the accurate worst regrets, we do the Experiment C.

About the Result 2, it was expected that the relationship between the worst regret and the loss matrix's rank r should be as the function $y = C \times \sqrt{x}$ (C is a constant), and our finding is consistent with this previous assumption.

About the Result 3, we think this abnormal decline is because of our way to generate the loss matrices. Since we use Algorithm 3 to generate the loss matrices of rank r, when $r \leq 17$ and use Algorithm 4 to generate the loss matrices of rank r, when $r \geq 18$, it is reasonable that the point, where $r = 17$, is a minimum point. Because when the r ($r \leq 17$) is getting close to 17, it is harder and harder to generate the loss matrices of rank r using the Algorithm 3. So the regret will decrease (getting away from the accurate regret). And when r ($r \geq 17$) is getting close to 20, it is easier and easier to generate the loss matrices of rank r using the Algorithm 4. So the regret will increase (getting close to the accurate regret). As a result, the point, where $r = 17$, is a minimum point.

C. The Accurate Worst Regret of Small $\boldsymbol{N}, \boldsymbol{T}$

In Experiment C, we also use traditional learning algorithm (Algorithm 1) to compute the regrets. And we fix the $N = T = 5$.

But this time, we generate all the possible matrices with 5 rows and 5 columns and these matrices' elements are chosen from set $\{0, 1\}$. In this case, we can get the accurate worst regret. The following table is the result:

TABLE I: THE ACCURATE WORST REGRET

rank	1	2	3	4	5
regret	1.317326	1.861010	2.174237	2.179599	2.178444

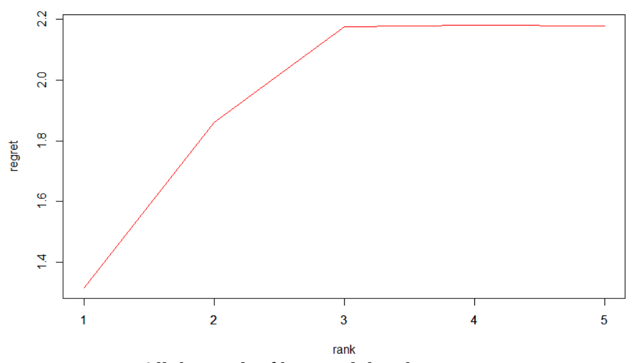

Fig. 2. All the result of binomial distribution, N = 5, T = 5.

The Fig. 2 is the picture describing the relationship between the worst regret and the loss matrices' rank r. And we compute the regret of all the possible loss matrices of rank r, and choose the maximum value of them as the worst regret of the rank r. So what we get is the accurate worst regret.

From the picture and data, we can see that the worst regret of the matrices with the rank r ($r = 4$) is bigger than that of the the matrices with the rank r ($r = 5$). Although this difference is not significant, yet it shows that the worst regret can decrease when the rank of loss matrices increases.

So the Result 1 in Experiment B (the worst regret can decrease when the rank increases) is reasonable.

D. $N = T = 20$ and Distribution D= Uniform Distribution ($maximum = 1, minimum = 0$)

The Fig. 3 is the picture describing the relationship between the worst regret and the loss matrices' rank r. And in Experiment D, we will use traditional learning algorithm (Algorithm 1) to compute the regrets.

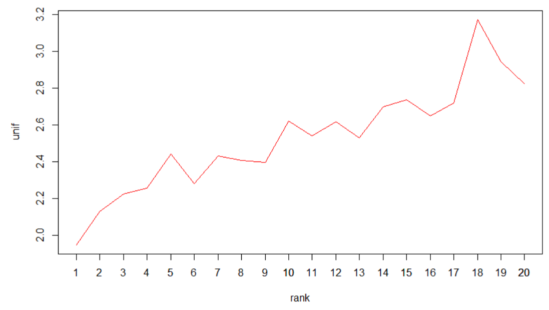

Fig. 3. 100000 trials for each rank of uniform distribution, N = 20, T = 20.

In Experiment D, we fix the $N = T = 20$, and we use Algorithm 3 with the distribution D = uniform distribution ($maximum = 1, minimum = 0$) to generate the loss matrices of rank r, when $r \leq 17$; and use Algorithm 4 with the distribution D = uniform distribution ($maximum = 1, minimum = 0$) to generate the loss matrices of rank r, when $r \geq 18$. For each certain rank r ($r \in \{1 \ldots \min(N,T)\}$), we do 100000 independent trials.

To compare the Fig. 3 with the Fig. 1, we can find that the worst regrets of Fig. 3 are much lower than corresponding worst regrets of Fig. 1. We think this is because of their different distributions.

When the elements of loss matrix follow the binominal distribution, they only have two values to choose from, so it is easier to get the linear dependent rows or columns. Then it is easier to generate the loss matices of certain rank r, which means the results we simulated are close to the accurate worst regret.

But when the elements of loss matrix follow the uniform distribution, these elements have infinite values to choose from. As a result, it is very hard to gender the loss matrices of certain rank r, therefore, it is harder to get close to the accurate worst regret of rank r.

According to this fact, our following research focus on the binomial distribution.

E. For Larger N and T (Also Binomial Distribution)

The Fig. 4 is the picture describing the relationship between the worst regret and the loss matrices' rank r. And in Experiment E, we will use traditional learning algorithm (Algorithm 1) to compute the regrets.

In Experiment E, we fix the $N = T = 50$, and we use Algorithm 3 with the distribution D = binomial distribution (p = 0.5) to generate the loss matrices of rank r, when $r \leq 42$; and use Algorithm 4 with the distribution D = binomial distribution (p = 0.5) to generate the loss matrices of rank r, when $r \geq 43$. For each certain rank r ($r \in \{1 \ldots \min(N,T)\}$), we do 100000 independent trials.

Although, the regrets may be not close to the accurate worst regrets at some points ($40 \geq r \geq 45$), yet as we can see the general shape of the curve.

From the Fig. 4, we find the three results in Experiment B are also true, which moreover proves that the findings in Experiment B can not only be applied to certain N and T but also bigger N and T.

About those abnormal points ($40 \geq r \geq 45$), they are because of our way to generate the loss matrices too. Since we use Algorithm 3 to generate the loss matrices of rank r, when $r \leq 42$; and use Algorithm 4 to generate the loss matrices of rank r, when $r \geq 43$, it is reasonable that the points, where r is near to 42, are minimum points.

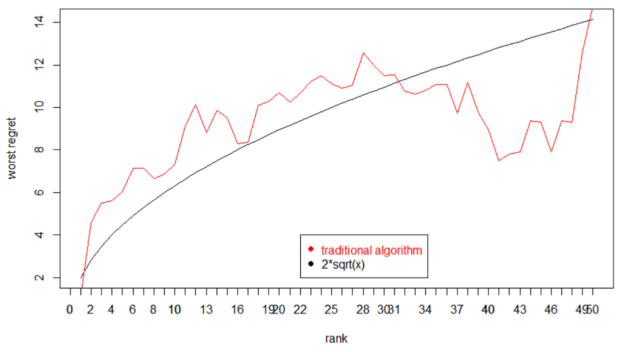

Fig. 4. 100000 trials for each rank of binomial distribution ($N = T = 50$).

Because when the r ($r \leq 42$) is getting close to 42, it is harder and harder to generate the loss matrices of rank r using the Algorithm 3. So the regret will be away from the accurate worst regret. But when r ($r \geq 42$) is getting close to 50, it is easier and easier to generate the loss matrices of rank r using the Algorithm 4. So the regret will getting close to the accurate regrets.

As a result, these points ($40 \geq r \geq 45$) are far below the accurate worst regrets.

F. Use Another Algorithm

In the following experiments, we will change the traditional learning algorithm (Algorithm 1) to another learning algorithm (Algorithm 2), which is designed to exploit low rank structure effectively. In Algorithm 2 we want to only use those experts that have different cumulative loss to compute the regrets. In this way, we expect that this another learning algorithm can compute the regrets more efficiently and we want to see if these two learning algorithms will generate different results.

1) Use another algorithm to repeat the Experiment B

In Experiment a, we will use another learning algorithm (Algorithm 2) to repeat the Experiment B.

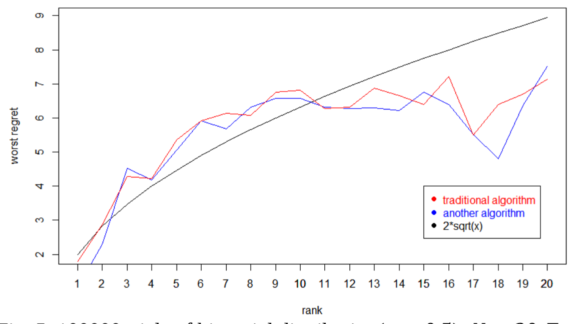

Fig. 5. 100000 trials of binomial distribution(p = 0.5), $N = 20, T = 20$(two algorithms).

So we also fix the $N = T = 20$, and we use Algorithm 3 with the distribution D = binomial distribution (p = 0.5) to generate the loss matrices of rank r, when $r \leq 17$; and use Algorithm 4 with the distribution D = binomial distribution (p = 0.5) to generate the loss matrices of rank r, when $r \geq 18$. For each certain rank r ($r \in \{1 \ldots \min(N,T)\}$), we do 100000 independent trials.

And we show the curve of the traditional algorithm (Algorithm 1) and the curve of another algorithm (Algorithm 2) in the same picture.

From the Fig. 5, we can find that: in Experiment B, these two algorithms have almost the same results.

2) Use another algorithm to repeat the Experiment D

In Experiment b, we will use another learning algorithm (Algorithm 2) to repeat the Experiment D.

So we also fix the $N = T = 20$, and we use Algorithm 3 with the distribution $D =$ uniform distribution ($maximum = 1, minimum = 0$) to generate the loss matrices of rank r, when $r \leq 17$; and use Algorithm 4 with the distribution $D =$ uniform distribution ($maximum = 1, minimum = 0$) to generate the loss matrices of rank r, when $r \geq 18$. For each certain rank r ($r \in \{1 \ldots \min(N, T)\}$), we do 100000 independent trials.

And we show the curves of the traditional algorithm (Algorithm 1) and the curve of another algorithm (Algorithm 2) in the same picture.

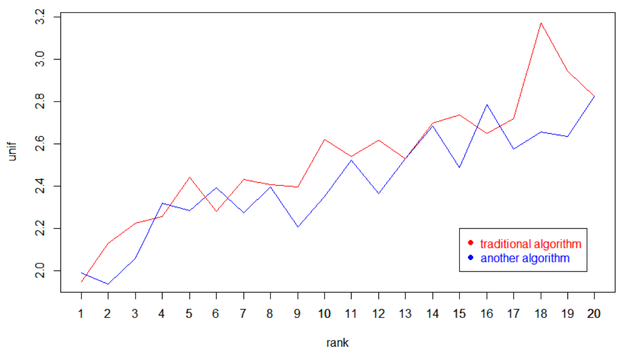

Fig. 6. 100000 trials for each rank of uniform distribution, $N = 20, T = 20$ (two algorithms).

From the Fig 6, we can find that: in Experiment D, these two algorithms have almost the same results.

3) Use another algorithm to repeat the Experiment E

In Experiment c, we will use another learning algorithm (Algorithm 2) to repeat the Experiment E.

So we also fix the $N = T = 50$, and we use Algorithm 3 with the distribution $D =$ binomial distribution ($p = 0.5$) to generate the loss matrices of rank r, when $r \leq 42$; and use Algorithm 4 with the distribution $D =$ binomial distribution ($p = 0.5$) to generate the loss matrices of rank r, when $r \geq 43$. For each certain rank r ($r \in \{1 \ldots \min(N, T)\}$), we do 100000 independent trials.

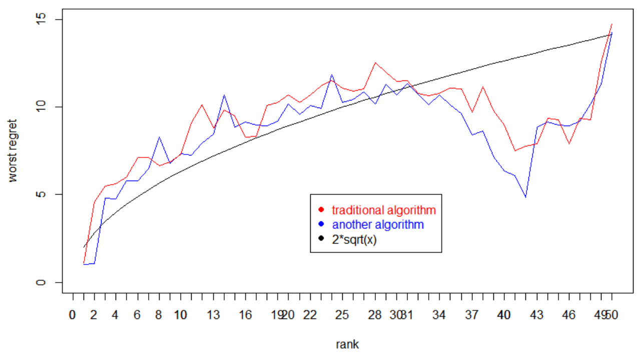

Fig. 7. 100000 trials for each rank of binomial distribution ($N = T = 50$)(two algorithms).

And we show the curves of the traditional algorithm (Algorithm 1) and the curve of another algorithm (Algorithm 2) in the same picture.

From the Fig. 7, we can find that: in Experiment E, these two algorithms have almost the same results.

In conclusion, this new algorithm (Algorithm 2) has almost the same results as the traditional algorithm (Algorithm 1), when it is used to compute the regrets of low rank.

IV. Conclusion

We have shown a traditional algorithm exploits low rank structure effectively and comparably to an algorithm explicitly designed to exploit low rank structure. This suggests the difficulty of proving accurate upper bounds is not as much in algorithm design as it is in uncovering how algorithms take advantage of coincidental structure. Future research might consider in more depth existing algorithms. In particular, given our results are experimental, theoretical justification or other independent verification of them is warranted.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

W. Y. Author conducted the research, analyzed the data; and wrote the simulation and learning algorithms parts of the paper. M.S. Author wrote the introduction and conclusion parts of the paper. All authors had approved the final version.

Acknowledgment

W. Y. Author thanks his adviser. He taught W.Y. much knowledge on relevant fields and provided W.Y. with help when he met some problems. W.Y. Author also thanks his parents. They gave him a lot of encouragement to overcome the difficulty. W.Y. Author also thanks his friends. They helped him check the errors. Without these people, this paper can not be accomplished.

References

[1] W. X. Liu, M. Spece, and S. H. Jia, “Fast learning and low rank experts: Novel adaptive methods,” *SSRN*, October 2019.

[2] C.-B. Nicolo and G. Lugosi, *Prediction, Learning, and Games*, Cambridge University Press, New York, NY, USA, 2006.

[3] E. Hazan, T. Koren, R. Livni, and Y. Mansour, “Online learning with low rank experts,” *CoRR*, 2016.

[4] Y. Koren, R. Bell, and C. Volinsky, “Matrix factorization techniques for recommender systems,” *Computer*, vol. 42, no. 8, pp. 30–37, August 2009.

[5] E. J. Candes and B. Recht, “Exact matrix completion via convex optimization,” *Foundations of Computational Mathematics*, vol. 9, no. 6, p. 717, Apr. 2009.

[6] R. Mazumder, T. Hastie, and R. Tibshirani, “Spectral regularization algorithms for learning large incomplete matrices,” *J. Mach. Learn. Res.*, vol. 11, pp. 2287–2322, August 2010.

[7] T. Koren and Roi Livni, “Affifine-invariant online optimization and the low-rank experts problem,” in *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. pp. 4747–4755, Curran Associates, Inc., 2017.

[8] A. Cohen and S. Mannor, “Online learning with many experts,” *CoRR*, 2017.

[9] S. Barman, A. Gopalan, and A. Saha, “Online learning for structured loss spaces,” *CoRR*, 2017.

[10] D. J. Foster, A. Rakhlin, and K. Sridharan, “Zigzag: A new approach to adaptive online learning,” *CoRR*, 2017.

[11] D. Adamskiy, W. M. Koolen, A. Chernov, and V. Vovk, “A closer look at adaptive regret,” *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 706–726, January 2016.

[12] E. Hazan and C. Seshadhri, *Efficient Learning Algorithms for Changing Environments*, vol. 382, p. 50, 2009.

[13] E. Gofer, N. Cesa-Bianchi, C. Gentile, and Y. Man-sour, "Regret minimization for branching experts," in *Proc. the 26th Annual Conference on Learning Theory*, Princeton, NJ, USA, June 2013, pp. 618–638.

Weiqi Yang is a senior student who majors in probability and statistics at the University of Science and Technology of China, Hefei, China.

In addition to this research, he has studied China's CPI over the last 20 years.

Mr. Yang was awarded the Outstanding Student Scholarship in University of Science and Technology of China.

Michael Spece graduated with the dual BS degrees in applied & computational mathematics and business economics & management from Caltech, Pasadena in 2008 and a joint PhD in machine learning & statistics from Carnegie Mellon University, Pittsburgh, 2018. He is the chief AI / data scientist at AllocateRite, New York City and researches online learning and foundations of data science.

Dr. Spece was offered the NSF fellowship in artificial intelligence, the NDSEG fellowship in mathematics, and was a NSF honorable mention in economics.