Journal of Machine Learning Research 19 (2018) 1-31 Submitted 3/17; Revised 7/18; Published 10/18

Maximum Selection and Sorting with Adversarial Comparators

Jayadev Acharya ACHARYA@CORNELL.EDU
School of ECE
Cornell University
Ithaca, NY 14853, USA

Moein Falahatgar MOEIN@UCSD.EDU
ECE Department
UC San Diego
La Jolla, CA 92093, USA

Ashkan Jafarpour ASHKAN.JAFARPOUR@GMAIL.COM
Google
Sunnyvale, CA 94089, USA

Alon Orlitsky ALON@UCSD.EDU
ECE and CSE Departments
UC San Diego
La Jolla, CA 92093, USA

Ananda Theertha Suresh THEERTHA@GOOGLE.COM
Google Research
New York, NY 10011, USA

Editor: Gabor Lugosi

Abstract

We study maximum selection and sorting of n numbers using imperfect pairwise comparators. The imperfect comparator returns the larger of the two inputs if the inputs are more than a given threshold apart and an adversarially-chosen input otherwise. We consider two adversarial models: a non-adaptive adversary that decides on the outcomes in advance and an adaptive adversary that decides on the outcome of each comparison depending on the previous comparisons and outcomes.

Against the non-adaptive adversary, we derive a maximum-selection algorithm that uses at most $2n$ comparisons in expectation and a sorting algorithm that uses at most $2n \ln n$ comparisons in expectation. In the presence of the adaptive adversary, the proposed maximum-selection algorithm uses $\Theta(n \log(1/\epsilon))$ comparisons to output a correct answer with probability at least $1 - \epsilon$, resolving an open problem in Ajtai et al. (2015).

Our study is motivated by a density-estimation problem. Given samples from an unknown distribution, we would like to find a distribution among a known class of n candidate distributions that is close to the underlying distribution in ℓ_1 distance. Scheffe's algorithm, for example, in Devroye and Lugosi (2001) outputs a distribution at an ℓ_1 distance at most 9 times the minimum and runs in time $\Theta(n^2 \log n)$. Using our algorithm, the runtime reduces to $\Theta(n \log n)$.

Keywords: noisy sorting, adversarial comparators, density estimation, Scheffe estimator

1. Introduction

Maximum selection and sorting are fundamental operations with widespread applications in computing, investment, marketing (Aggarwal et al., 2009), decision making (Thurstone, 1927; David, 1963), and sports. These operations are often accomplished via pairwise comparisons between elements, and the goal is to minimize the number of comparisons.

For example, one may find the largest of n elements by first comparing two elements and then successively comparing the larger one to a new element. This simple algorithm takes $n-1$ comparisons, and it is easy to see that $n-1$ comparisons are necessary. Similarly, *merge sort* sorts n elements using less than $n \log n$ comparisons, close to the information theoretic lower bound of $\log n! = n \log n - o(n)$.

However, in many applications, the pairwise comparisons may be imprecise. For example, in comparing two random numbers, such as stock performances, or team strengths, the output of the comparison may vary due to chance. Consequently, a number of researchers have considered maximum selection and sorting with imperfect, or noisy, comparators. The comparators in these models mostly function correctly but occasionally may produce an inaccurate comparison result, where the form of inaccuracy is dictated by the application.

Based on the form of inaccuracy, models can be divided into two categories: probabilistic and adversarial. Probabilistic models can be parametric or non-parametric. One of the simplest parametric probabilistic models was considered in Feige et al. (1994), where the output of each comparator could be wrong with some known probability p. Algorithms applying this model for maximum selection were proposed in Adler et al. (1994) and for ranking in Karp and Kleinberg (2007); Ben-Or and Hassidim (2008); Braverman and Mossel (2008); Braverman et al. (2016).

Another parametric family of probabilistic models, the Bradley-Terry-Luce model (Bradley and Terry, 1952) assumes that if two values x and y are compared, then x is selected as the larger with probability $x/(x+y)$. Observe that the comparison is correct with probability $\max\{x,y\}/(x+y) \geq 1/2$. Algorithms for ranking and estimating values under this and another related model, the Plackett-Luce (Plackett, 1975; Luce, 2005), are proposed, for example, in Negahban et al. (2012); Szörényi et al. (2015). The Mallows model is yet another example of a parametric probabilistic model and is studied in Busa-Fekete et al. (2014).

Non-parametric probabilistic models assume some natural constraints on comparison probabilities, such as Strong Stochastic Transitivity or Stochastic Triangle Inequality. Algorithms for maximum selection and sorting under these models are studied in Falahatgar et al. (2017b,a, 2018); Yue and Joachims (2011) and algorithms for comparison-probability matrix estimation are considered in Shah et al. (2016). This model is also considered for the top-k sorting problem in Chen et al. (2017b,a).

We consider a model where, unlike the probabilistic models, the comparison outcome can be adversarial. If the numbers compared are more than a threshold Δ apart, the comparison is correct, while if they differ by at most Δ, the comparison outcome is arbitrary, and possibly even adversarial.

This model can be partially motivated by physical observations. Measurements are regularly quantized and often adulterated with some measurement noise. Quantities with the same quantized value may, therefore, be incorrectly compared. In psychophysics, the

Weber-Fechner law (Ekman, 1959) stipulates that humans can distinguish between two physical stimuli only when their difference exceeds some threshold (known as *just noticeable difference*). Additionally, in sports, a judge or a home-team advantage may, even adversarially, sway the outcome of a game between two teams of similar strength but not between teams of significantly different strengths. Our main motivation for the model derives from the important problem of density estimation and distribution learning.

1.1. Density estimation via pairwise comparisons

In a typical PAC-learning setup (Valiant, 1984; Kearns et al., 1994), we are given samples from an unknown distribution p_0 in a known distribution class $\mathcal{P}$ and would like to find, with high probability, a distribution $\hat{p} \in \mathcal{P}$ such that $\|\hat{p} - p_0\|_1 < \delta$.

One standard approach proceeds in two steps (Devroye and Lugosi, 2001):

1. Offline, construct a δ-*cover* of $\mathcal{P}$, a finite collection $\mathcal{P}_\delta \subseteq \mathcal{P}$ of distributions such that for any distribution $p \in \mathcal{P}$, there is a distribution $q \in \mathcal{P}_\delta$ such that $\|p - q\|_1 < \delta$.

2. Using the samples from p_0, find a distribution in $\mathcal{P}_\delta$ whose ℓ_1 distance to p_0 is close to the ℓ_1 distance of the distribution in $\mathcal{P}_\delta$ that is closest to p_0.

These two steps output a distribution whose ℓ_1 distance from p_0 is *close* to δ. Surprisingly, for several common distribution classes, such as Gaussian mixtures, the number of samples required by this generic approach matches the information theoretically optimal sample complexity, up to logarithmic factors (Daskalakis and Kamath, 2014; Suresh et al., 2014; Diakonikolas et al., 2016).

The Scheffe Algorithm (Scheffe, 1947; Devroye and Lugosi, 2001) is a popular method for implementing the second step, namely to find a distribution in $\mathcal{P}_\delta$ with a small ℓ_1 distance from p_0. It takes every pair of distributions in $\mathcal{P}_\delta$ and uses the samples from p_0 to decide which of the two distributions is closer to p_0. It then declares the distribution that "wins" the most pairwise closeness comparisons to be the nearly-closest to p_0. As shown in Devroye and Lugosi (2001), the Scheffe algorithm yields, with high probability, a distribution that is at most nine times further from p_0 than the distribution in $\mathcal{P}_\delta$ with the lowest ℓ_1 distance from p_0, plus a diminishing additive term; hence, a distribution that is roughly 9δ away from p_0 is found. Since this algorithm compares every pair of distributions in $\mathcal{P}_\delta$, it uses quadratic in $|\mathcal{P}_\delta|$ comparisons. In Section 6, we use maximum-selection results to derive an algorithm with the same approximation guarantee but linear in $|\mathcal{P}_\delta|$ comparisons.

1.2. Organization

This paper is organized as follows: in Section 2, we define the problem and introduce the notations; in Section 3, we summarize the results; in Section 4, we derive simple bounds and describe the performance of simple algorithms; and, in Section 5, we present our main maximum-selection algorithms. The relation between density estimation problem and our comparison model is discussed in Section 6, and, in Section 7, we discuss sorting with adversarial comparators.

2. Notations and preliminaries

Practical applications call for sorting or selecting the maximum of not just numbers, but, rather, of items with associated values—for example, finding the person with the highest salary, the product with the lowest price, or a sports team with the most *capability* of winning. Associate with each item i a real value x_i and let $\mathcal{X} \stackrel{\text{def}}{=} \{x_1, \ldots, x_n\}$ be the multiset of values. In maximum selection, we use noisy pairwise comparisons to find an index i such that x_i is close to the largest element $x^* \stackrel{\text{def}}{=} \max\{x_1, \ldots, x_n\}$.

Formally, a faulty comparator $\mathcal{C}$ takes two distinct indices i and j and, if $|x_i - x_j| > \Delta$, outputs the index associated with the higher value, while if $|x_i - x_j| \le \Delta$, outputs either i or j, possibly adversarially. Without loss of generality, we assume that $\Delta = 1$. Then,

$$\mathcal{C}(i,j) = \begin{cases} \arg\max\{x_i, x_j\} & \text{if} \quad |x_i - x_j| > 1, \\ i \text{ or } j \text{ (adversarially)} & \text{if} \quad |x_i - x_j| \le 1. \end{cases}$$

It is easier to think just of the numbers, rather than the indices. Therefore, informally we will simply view the comparators as taking two real inputs x_i and x_j, and outputting

$$\mathcal{C}(x_i,x_j) = \begin{cases} \max\{x_i, x_j\} & \text{if} \quad |x_i - x_j| > 1, \\ x_i \text{ or } x_j \text{ (adversarially)} & \text{if} \quad |x_i - x_j| \le 1. \end{cases} \tag{1}$$

We consider two types of adversarial comparators: *non-adaptive* and *adaptive*.

- A *non-adaptive adversarial comparator* has complete knowledge of $\mathcal{X}$ and the algorithm but must fix its outputs for every pair of inputs before the algorithm starts
- An *adaptive adversarial comparator* not only has access to the algorithm and the inputs but is also allowed to adaptively decide the outcomes of the queries taking into account all the previous comparisons made by the algorithm

A non-adaptive comparator can be naturally represented by a directed graph with n nodes representing the n indices. There is an edge from node i to node j if the comparator declares x_i to be larger than x_j, namely, $\mathcal{C}(x_i, x_j) = x_i$. Figure 1 is an example of such a comparator, where, for simplicity, we show only the values 0, 1, 1, 2, and not the indices. Note that, by definition, $\mathcal{C}(2, 0) = 2$, but for all the other pairs, the outputs can be decided by the comparator. In this example, the comparator declares the node with value 2 as the "winner" against the right node with value 1 but as the "loser" against the left node, also with value 1. Among the two nodes with value 1, it arbitrarily declares the left one as the winner. An adaptive adversary reveals the edges one-by-one as the algorithm proceeds.

We refer to each comparison as a *query*. The number of queries an algorithm $\mathcal{A}$ makes for $\mathcal{X} = \{x_1, \ldots, x_n\}$ is its *query complexity*, denoted by $Q_n^{\mathcal{A}}$.[1] Our algorithms are randomized, and $Q_n^{\mathcal{A}}$ is a random variable. The *expected query complexity* of $\mathcal{A}$ for the input $\mathcal{X}$ is

$$q_n^{\mathcal{A}} \stackrel{\text{def}}{=} \mathbb{E}[Q_n^{\mathcal{A}}],$$

where the expectation is over the randomness of the algorithm. Note that the expected query complexity is defined for all runs of an algorithm, and it is independent of the success probability.

1. This is a slight abuse of notation suppressing $\mathcal{X}$.

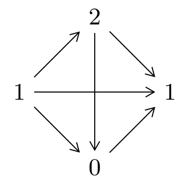


Figure 1: Comparator for four inputs with values $\{0, 1, 1, 2\}$

Let $\mathcal{C}_{\text{non}}(\mathcal{X})$, or simply $\mathcal{C}_{\text{non}}$, be the set of all non-adaptive adversarial comparators, and let $\mathcal{C}_{\text{adpt}}$ be the set of all adaptive adversarial comparators. The *maximum expected query complexity* of $\mathcal{A}$ against non-adaptive adversarial comparators is

$$q_n^{\mathcal{A},\text{non}} \stackrel{\text{def}}{=} \max_{\mathcal{C}\in\mathcal{C}_{\text{non}}} \max_{\mathcal{X}} q_n^{\mathcal{A}}. \tag{2}$$

Similarly, the maximum expected query complexity of $\mathcal{A}$ against adaptive adversarial comparators is

$$q_n^{\mathcal{A},\text{adpt}} \stackrel{\text{def}}{=} \max_{\mathcal{C}\in\mathcal{C}_{\text{adpt}}} \max_{\mathcal{X}} q_n^{\mathcal{A}}.$$

We evaluate an algorithm by how close its output is to x^* (the maximum of $\mathcal{X}$).

Definition 1 *A number x is a t-*approximation *of x^* if $x \geq x^* - t$.*

The *t-approximation error* of an algorithm $\mathcal{A}$ over n inputs is

$$\mathcal{E}_n^{\mathcal{A}}(t) \stackrel{\text{def}}{=} \Pr\left(Y_{\mathcal{A}}(\mathcal{X}) < x^* - t\right),$$

the probability that $\mathcal{A}$'s output $Y_{\mathcal{A}}(\mathcal{X})$ is *not* a t-approximation of x^*. For an algorithm $\mathcal{A}$, the maximum t-approximation error for the worst non-adaptive adversary is

$$\mathcal{E}_n^{\mathcal{A},\text{non}}(t) \stackrel{\text{def}}{=} \max_{\mathcal{C}\in\mathcal{C}_{\text{non}}} \max_{\mathcal{X}} \mathcal{E}_n^{\mathcal{A}}(t),$$

and, similarly, for the adaptive adversary,

$$\mathcal{E}_n^{\mathcal{A},\text{adpt}}(t) \stackrel{\text{def}}{=} \max_{\mathcal{C}\in\mathcal{C}_{\text{adpt}}} \max_{\mathcal{X}} \mathcal{E}_n^{\mathcal{A}}(t).$$

For the non-adaptive adversary, the minimum t-approximation error of any algorithm is

$$\mathcal{E}_n^{\text{non}}(t) \stackrel{\text{def}}{=} \min_{\mathcal{A}} \mathcal{E}_n^{\mathcal{A},\text{non}}(t),$$

and, similarly, for the adaptive adversary,

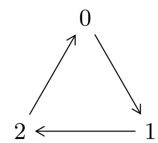
$$\mathcal{E}_n^{\text{adpt}}(t) \stackrel{\text{def}}{=} \min_{\mathcal{A}} \mathcal{E}_n^{\mathcal{A},\text{adpt}}(t).$$

Since adaptive adversarial comparators are stronger than non-adaptive, for all t,

$$\mathcal{E}_n^{\text{adpt}}(t) \geq \mathcal{E}_n^{\text{non}}(t).$$

Example 1 shows that $\mathcal{E}_3^{\text{non}}(t) \geq \frac{1}{3}$ for all $t < 2$.

Example 1 $\mathcal{E}_3^{\mathrm{non}}(t) \geq \frac{1}{3}$ *for all* $t < 2$. *Consider* $\mathcal{X} = \{0, 1, 2\}$ *and the following comparators. .*



By symmetry, no algorithm can differentiate between the three inputs. Hence, any algorithm will output 0 with probability $1/3$.

3. Previous and new results

In Section 4.1 we lower bound $\mathcal{E}_n^{\mathrm{non}}(t)$ as a function of t. In Lemma 2, we show that for all $t < 1$ and odd n, $\mathcal{E}_n^{\mathrm{non}}(t) = 1 - 1/n$, namely for some $\mathcal{X}$, approximating the maximum to within less than one is equivalent to guessing a random x_i as the maximum. In Lemma 3, we modify Example 1 and show that for all $t < 2$ and odd n, any algorithm has t-approximation error close to $1/2$ for some input.

We propose a number of algorithms to approximate the maximum. These algorithms have different guarantees in terms of the probability of error, approximation factor, and query complexity.

We first consider two simple algorithms: the complete tournament, denoted COMPL, and the sequential selection, denoted SEQ. Algorithm COMPL compares all the possible input pairs and declares the input with the most wins as the maximum. We show the simple result that COMPL outputs a 2-approximation of x^*. We then consider the algorithm SEQ that compares a pair of inputs, discards the loser, and compares the winner with a new input. We show that even under random selection of the inputs, there exist inputs such that, with high probability, SEQ cannot provide a constant approximation to x^*.

We then consider more advanced algorithms. The knock-out algorithm, at each stage, pairs the inputs at random and keeps the winners of the comparisons for the next stage. We design a slight modification of this algorithm, denoted KO-MOD that achieves a 3-approximation with error probability at most ϵ, even against adaptive adversarial comparators. We note that Ajtai et al. (2015) proposed a different algorithm with similar performance guarantees.

Motivated by quick-sort, we propose a quick-select algorithm Q-SELECT that outputs a 2-approximation with zero error probability. It has an expected query complexity of at most $2n$ against the non-adaptive adversary. However, in Example 2, we see that this algorithm requires $\binom{n}{2}$ queries against the adaptive adversary.

This leaves the question of whether there is a randomized algorithm for 2-approximation of x^* with $\mathcal{O}(n)$ queries against the adaptive adversary. In fact, Ajtai et al. (2015) pose this as an open question. We resolve this problem by designing an algorithm COMB that combines quick-select and knock-out. We prove that COMB outputs a 2-approximation with probability of error, at most, ϵ, using $\mathcal{O}(n \log \frac{1}{\epsilon})$ queries. We summarize the results in Table 1.

We note that while we focus on randomized algorithms, Ajtai et al. (2015) also studied the best possible trade-offs for deterministic algorithms. They designed a deterministic

<table>
<tr><th>Algorithm</th><th>Notation</th><th>Approximation</th><th>$q_n^{\mathcal{A},\text{non}}$</th><th>$q_n^{\mathcal{A},\text{adpt}}$</th></tr>
<tr><td>complete tournament</td><td>COMPL</td><td>$\mathcal{E}_n^{\text{COMPL,adpt}}(2) = 0$</td><td colspan="2">$\binom{n}{2}$</td></tr>
<tr><td>deterministic upper bound (Ajtai et al., 2015)</td><td>-</td><td>$\mathcal{E}_n^{\mathcal{A},\text{adpt}}(2) = 0$</td><td colspan="2">$\Theta(n^{\frac{3}{2}})$</td></tr>
<tr><td>deterministic lower bound (Ajtai et al., 2015)</td><td>-</td><td>$\mathcal{E}_n^{\mathcal{A},\text{adpt}}(2) = 0$</td><td>-</td><td>$\Omega(n^{\frac{4}{3}})$</td></tr>
<tr><td>sequential</td><td>SEQ</td><td>$\mathcal{E}_n^{\text{SEQ,non}}\left(\frac{\log n}{\log\log n} - 1\right) \to 1$</td><td colspan="2">$n-1$</td></tr>
<tr><td>modified knock-out</td><td>KO-MOD</td><td>$\mathcal{E}_n^{\text{KO-MOD,adpt}}(3) < \epsilon$</td><td colspan="2">$< n + \frac{1}{2}\log^4 n \left\lceil \frac{1}{\epsilon} \ln \frac{1}{\epsilon} \right\rceil^2$</td></tr>
<tr><td>quick-select</td><td>Q-SELECT</td><td>$\mathcal{E}_n^{\text{Q-SELECT,adpt}}(2) = 0$</td><td>$< 2n$</td><td>$\binom{n}{2}$</td></tr>
<tr><td>knock-out and quick-select combination</td><td>COMB</td><td>$\mathcal{E}_n^{\text{COMB,adpt}}(2) < \epsilon$</td><td colspan="2">$\mathcal{O}\left(n \log \frac{1}{\epsilon}\right)$</td></tr>
</table>

Table 1: Maximum selection algorithms

algorithm for 2-approximation of the maximum using only $\mathcal{O}(n^{3/2})$ queries. Moreover, they prove that no deterministic algorithm with fewer than $\Omega(n^{4/3})$ queries can output a 2-approximation of x^* for the adaptive adversarial model.

4. Simple results

In Lemmas 2 and 3, we prove lower bounds on the error probability of any algorithm that provides a t-approximation of x^* for $t < 1$ and $t < 2$, respectively. We then consider two straightforward algorithms for finding the maximum. One is the complete tournament, where all pairs of inputs are compared, and the other is sequential, where inputs are compared sequentially, and the loser is discarded at each comparison.

4.1. Lower bounds

We show the following two results:

- $\mathcal{E}_n^{\text{non}}(t) = 1 - \frac{1}{n}$ for all $0 \le t < 1$ and odd n
- $\mathcal{E}_n^{\text{non}}(t) \ge \frac{1}{2} - \frac{1}{2n}$ for all $1 \le t < 2$ and odd n

These lower bounds can be applied to n, which is even, by adding an extra input that is smaller than all the other inputs and loses to them.

Lemma 2 *For all $0 \le t < 1$ and odd n,*

$$\mathcal{E}_n^{\text{non}}(t) = 1 - \frac{1}{n}.$$

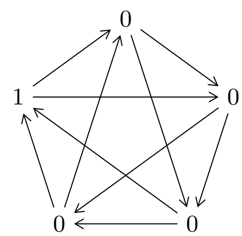


Figure 2: Tournament for Lemma 2 when $n = 5$

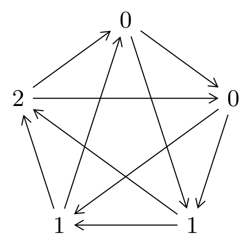


Figure 3: Tournament for Lemma 3 when $n = 5$

Proof Let $(x_1, x_2, \ldots, x_n)$ be an unknown permutation of $(1, \underbrace{0, \ldots, 0}_{n-1})$. Suppose we consider an adversary that ensures each input wins exactly $(n-1)/2$ times. An example is shown in Figure 2 for $n = 5$.

To get a lower bound on the performance of any randomized algorithm, we use Yao's principle. We consider only deterministic algorithms over a uniformly chosen permutation of the inputs, namely only one of the coordinates is 1, and the remaining are less than $1 - t$. In this case, if we fix any comparison graph (as in Figure 2), and permute the inputs, the algorithm cannot distinguish between 1 and 0's, and outputs 0 with probability $1 - 1/n$; therefore, $\mathcal{E}_n^{\text{non}}(t) \geq 1 - \frac{1}{n}$. Also, an algorithm that randomly picks an element as the maximum achieves the error $1 - 1/n$; hence, the lemma. ■

Lemma 3 *For all* $1 \leq t < 2$ *and odd* n*,*

$$\mathcal{E}_n^{\text{non}}(t) \geq \frac{1}{2} - \frac{1}{2n}.$$

Proof Let m be $(n-1)/2$. Let $(x_1, x_2, \ldots, x_n)$ be an unknown permutation of $(2, \underbrace{1, \ldots, 1}_{m}, \underbrace{0, \ldots, 0}_{m})$. Suppose the adversary ensures that 2 loses against all the 1's and, indeed, all inputs have exactly $(n-1)/2$ wins. An example is shown in Figure 3.

Similar to Lemma 2, the inputs are all identical to the algorithm, and, therefore, the algorithm outputs one of the 0's with probability $\frac{m}{n} = \frac{1}{2} - \frac{1}{2n}$. ■

4.2. Two elementary algorithms

In this section, we analyze two well-known maximum selection algorithms, the complete tournament and the sequential selection. We discuss their strengths and weaknesses and show that there is a trade-off between the query complexity and the approximation guarantees of these two algorithms. Another well-known algorithm for maximum selection is the knock-out algorithm, and we discuss a variant of it in Section 5.1.

4.2.1. Complete tournament (round-robin)

As its name evinces, a complete tournament involves a match between every pair of teams. Using this metaphor to competitions, we compare all the $\binom{n}{2}$ input pairs, and the input with maximum wins is declared as the output. If two or more inputs end up with the highest wins, any of them can be declared as the output. This algorithm is formally stated in COMPL.

> **input:** $\mathcal{X}$
> compare all input pairs in $\mathcal{X}$, count the number of times each input wins
> **output:** an input with the maximum number of wins

Algorithm COMPL - Complete tournament

Lemma 4 shows that COMPL gives a 2-approximation against both adversaries. The result, although weaker than the deterministic guarantees of Ajtai et al. (2015), is illustrative and useful in the algorithms proposed later.

Lemma 4 $q_n^{\text{COMPL,adpt}} = \binom{n}{2}$ *and* $\mathcal{E}_n^{\text{COMPL,adpt}}(2) = 0$.

Proof The number of queries is clearly $\binom{n}{2}$. To show $\mathcal{E}_n^{\text{COMPL,adpt}}(2) = 0$, note that if $y < x^* - 2$, then for all z that y wins over, $z \le y + 1 < x^* - 1$, and therefore x^* also beats them. Since x^* wins over y, it wins over more inputs than y, and y cannot be the output of the algorithm. It follows that the input with maximum wins is a 2-approximation of x^*. ∎

COMPL is deterministic, and, after $\binom{n}{2}$ queries, it outputs a 2-approximation of x^*. If the comparators are noiseless, we can simply compare the inputs sequentially, discarding the loser at each step, and, thus, requiring only $n - 1$ comparisons. This evokes the hope of finding a deterministic algorithm that requires a linear number of comparisons and outputs a 2-approximation of x^*. As mentioned earlier, however, Ajtai et al. (2015) showed it is not achievable, as they proved that any deterministic 2-approximation algorithm requires $\Omega(n^{4/3})$ queries. They also showed a strictly superlinear lower bound on any deterministic constant-approximation algorithm. They designed a deterministic 2-approximation algorithm using $\mathcal{O}(n^{3/2})$ queries.

4.2.2. Sequential selection

Sequential selection first compares a random pair of inputs and, at each successive step, compares the winner of the last comparison with a randomly chosen *new* input. It outputs the final remaining input. This algorithm uses $n - 1$ queries.

```
input: 𝒳
    choose a random y ∈ 𝒳 and remove it from 𝒳
    while 𝒳 is not empty
        choose a random x ∈ 𝒳 and remove it from 𝒳
        y ← 𝒞(x, y)
    end while
output: y
```

Algorithm SEQ - Sequential selection

Lemma 5 shows that even against the non-adaptive adversary, the algorithm cannot output a constant-approximation of x^*.

Lemma 5 *Let* $s = \frac{\log n}{\log \log n}$. *For all* $t < s$,

$$\mathcal{E}_n^{\text{SEQ,non}}(t) \geq 1 - \frac{1}{\log \log n}.$$

Proof Assume that s, $\log n$, and $\log \log n$ are integers and

$$x_i = \begin{cases} s & \text{for } i = 1, \\ s-1 & \text{for } i = 2, \ldots, r, \\ s-2 & \text{for } i = r+1, \ldots, r^2, \\ \vdots & \\ m & \text{for } i = r^{s-m-1}+1, \ldots, r^{s-m}, \\ \vdots & \\ 0 & \text{for } i = r^{s-1}+1, \ldots, r^s, \end{cases}$$

where $r = \log n$. Consider the following non-adaptive adversarial comparator:

$$\mathcal{C}(x_i, x_j) = \begin{cases} \max\{x_i, x_j\} & \text{if } |x_i - x_j| > 1, \\ \min\{x_i, x_j\} & \text{if } |x_i - x_j| \leq 1. \end{cases} \tag{3}$$

The sequential algorithm takes a random permutation of the inputs. It then starts by comparing the first two elements and then sequentially compares the winner with the next element, and so on. Let L_j be the location in the permutation where input j appears for the last time. The next two observations follow from the construction of inputs and comparators respectively.

Observation 1 *Input* j *appears at least* $(\log n - 1)$ *times that of input* $j+1$.

Observation 2 *For the adversarial comparator defined in* (3), *if* $L_0 > L_1 > \ldots > L_s$, *then no input* j *can survive beyond location* L_{j-1}, *and, therefore,* SEQ *outputs* 0.

As a consequence of Observation 1, in the random permutation of inputs, $L_j > L_{j+1}$ with probability at least $1 - \frac{1}{\log n}$. By the union bound, $L_0 > L_1 > \ldots > L_s$ with probability at least,

$$1 - \frac{s}{\log n} = 1 - \frac{1}{\log \log n}.$$

By applying Observation 2, SEQ outputs 0 with probability at least $1 - \frac{1}{\log \log n}$. ■

5. Algorithms

In the previous section, we saw that the complete tournament, COMPL, always outputs a 2-approximation but has quadratic query complexity, while the sequential selection, SEQ, has linear query complexity but a poor approximation guarantee. A natural question to ask is whether there exist algorithms with bounded error and linear query complexity. In this section, we propose algorithms with linear query complexity and approximation guarantees that compete with the best possible, namely, 2-approximation of x^*.

We propose three algorithms with different performance guarantees:

- **Modified knock-out**, described in Section 5.1, has linear query complexity, and, with high probability, outputs a 3-approximation of x^* against both adaptive and non-adaptive adversaries
- **Quick-select**, described in Section 5.2, outputs a 2-approximation to x^* (against both adversaries). It also has a linear expected query complexity against non-adaptive adversarial comparators
- **Knock-out and quick-select combination**, described in Section 5.3, has linear query complexity, and, with high probability, outputs a 2-approximation of x^* even against adaptive adversarial comparators

We now go over these algorithms in detail.

5.1. Modified knock-out

For simplification, in this section, we assume that $\log n$ is an integer. The knock-out algorithm derives its name from knock-out competitions where the tournament is divided into $\log n$ successive rounds. In each round, the inputs are paired at random, and the winners advance to the next round. Therefore, in round i, there are $\frac{n}{2^{i-1}}$ inputs. The winner at the end of $\log n$ rounds is declared as the maximum.

Under our adversarial model, at each round of the knock-out algorithm, the largest remaining input decreases by at most one. Therefore, the knock-out algorithm finds at least $\log n$-approximation of x^*. Analyzing the precise approximation error of knock-out algorithm appears to be difficult. However, simulations suggest that for any large n, for the set consisting of $0.2 \cdot n$ 0's, $\alpha \cdot n$ 1's, $(0.7 - \alpha) \cdot n$ 2's, $0.1 \cdot n$ 3's, and a single 4, where $0 < \alpha < 0.7$ is an appropriately chosen parameter, the knock-out algorithm is not able to find a 3-approximation of x^* with positive constant probability. The problem with knock-out algorithm is that if at any of the $\log n$ rounds, many inputs are within 1 from the largest input at that round, there is a fair chance that the largest input will be eliminated. If this elimination happens in several rounds, we will end up with a number significantly smaller than x^*.

To circumvent the problem of discarding large inputs, we select a specified number of inputs at each round and save them for the very end, thereby ensuring that at every round, if the largest input is eliminated, then an input within 1 from it has been saved. We then perform a complete tournament on these saved inputs. The algorithm is explained in KO-MOD.

> **input:** $\mathcal{X}$
> pair the inputs of $\mathcal{X}$ randomly, let $\mathcal{X}'$ be the winners
> **output:** $\mathcal{X}'$

Algorithm KO-SUB - Subroutine for KO-MOD and COMB

> **input:** $\mathcal{X}, \epsilon$
> $\mathcal{Y} = \emptyset$, $n_1 = \lceil \frac{1}{\epsilon} \ln \frac{1}{\epsilon} \cdot \log n \rceil$
> **while** $|\mathcal{X}| > n_1$
> randomly choose n_1 inputs from $\mathcal{X}$ and *copy* them to $\mathcal{Y}$
> $\mathcal{X} \leftarrow$ KO-SUB$(\mathcal{X})$
> **end while**
> **output:** COMPL$(\mathcal{X} \cup \mathcal{Y})$

Algorithm KO-MOD - Modified knock-out algorithm

In Theorem 6, we show that KO-MOD has a 3-approximation error less than ϵ.

We first explain the algorithm and then state the result. Let $n_1 \stackrel{\text{def}}{=} \lceil \frac{1}{\epsilon} \ln \frac{1}{\epsilon} \cdot \log n \rceil$. At each round, we add n_1 of the remaining inputs at random to the multiset $\mathcal{Y}$ and run the knock-out subroutine KO-SUB on the multiset $\mathcal{X}$. When $|\mathcal{X}| \le n_1$, we perform a complete tournament on $\mathcal{X} \cup \mathcal{Y}$ and declare the output as the winner. We show that, with probability at least $1 - \epsilon$, the final set $\mathcal{Y}$ contains at least one input which is a 1-approximation of x^*. Since the complete tournament outputs a 2-approximation of its maximum input, KO-MOD outputs a 3-approximation of x^* with probability greater than $1 - \epsilon$.

Theorem 6 *For $n_1 \ge 2$, we have $q_n^{\text{KO-MOD,adpt}} < n + \frac{1}{2}(\log^4 n) \cdot \lceil \frac{1}{\epsilon} \ln \frac{1}{\epsilon} \rceil^2$ and $\mathcal{E}_n^{\text{KO-MOD,adpt}}(3) < \epsilon$.*

Proof The number of comparisons made by KO-SUB is at most $\frac{n}{2} + \frac{n}{4} + \frac{n}{8} + \ldots < n$. Observe that KO-SUB is called $m \stackrel{\text{def}}{=} \left\lceil \log \frac{n}{n_1} \right\rceil$ times. Let $\mathcal{X}_i$ be the multiset $\mathcal{X}$ at the start of the ith call to KO-SUB. Let $\mathcal{X}_{m+1}$ and $\mathcal{Y}_{m+1}$ be the multisets $\mathcal{X}$ and $\mathcal{Y}$ right before calling compl. Then,

$$\begin{aligned}
|\mathcal{X}_{m+1} \cup \mathcal{Y}_{m+1}| &\le |\mathcal{X}_{m+1}| + |\mathcal{Y}_{m+1}| \\
&\le n_1 + \sum_{i=1}^{m} (|\mathcal{Y}_{i+1}| - |\mathcal{Y}_i|) \\
&\le n_1 + mn_1 \\
&= \left(\left\lceil \log \frac{n}{n_1} \right\rceil + 1 \right) \cdot \left\lceil \frac{1}{\epsilon} \ln \frac{1}{\epsilon} \cdot \log n \right\rceil \\
&\le \left(\left\lceil \log \frac{n}{n_1} \right\rceil + 1 \right) \cdot \left\lceil \frac{1}{\epsilon} \ln \frac{1}{\epsilon} \right\rceil \lceil \log n \rceil \\
&\le \log^2 n \cdot \left\lceil \frac{1}{\epsilon} \ln \frac{1}{\epsilon} \right\rceil,
\end{aligned}$$

where the last inequality follows as $n_1 \ge 2$ and $\log n$ is an integer. Since the complete tournament is quadratic in the input size, the total number of queries is at most $n + \frac{1}{2} \log^4 n \left\lceil \frac{1}{\epsilon} \ln \frac{1}{\epsilon} \right\rceil^2$.

Next, we bound the error of ko-mod. Let

$$\mathcal{X}^* \stackrel{\text{def}}{=} \{x \in \mathcal{X} : x \ge x^* - 1\}$$

be the multiset of all inputs that are at least $x^* - 1$. For $i \le m+1$, let $\mathcal{X}_i^* = \mathcal{X}_i \cap \mathcal{X}^*$ and $\mathcal{Y}_{m+1}^* = \mathcal{Y}_{m+1} \cap \mathcal{X}^*$. Let $\alpha_i \stackrel{\text{def}}{=} \frac{|\mathcal{X}_i^*|}{|\mathcal{X}_i|}$ and $\alpha = \max\{\alpha_1, \alpha_2, \ldots, \alpha_m\}$. We show that, with high probability, $|\mathcal{X}_{m+1}^* \cup \mathcal{Y}_{m+1}^*| \ge 1$, namely, some input in $\mathcal{X}_{m+1} \cup \mathcal{Y}_{m+1}$ belongs to $\mathcal{X}^*$. In particular, we show that, with probability $1 - \epsilon$, for large α, $|\mathcal{Y}_{m+1}^*| > 0$, and for small α, $x^* \in \mathcal{X}_{m+1}$. Observe that

$$\begin{aligned}
\Pr(x^* \notin \mathcal{X}_{m+1}^*) &= \sum_{i=1}^{m} \Pr(x^* \notin \mathcal{X}_{i+1}^* | x^* \in \mathcal{X}_i) \cdot \Pr(x^* \in \mathcal{X}_i) \\
&\le \sum_{i=1}^{m} \Pr(x^* \notin \mathcal{X}_{i+1}^* | x^* \in \mathcal{X}_i) \\
&\stackrel{(a)}{\le} \sum_{i=1}^{m} \frac{|\mathcal{X}_i^*| - 1}{|\mathcal{X}_i| - 1} \\
&\le \sum_{i=1}^{m} \alpha_i \\
&\le \alpha m,
\end{aligned}$$

where (a) follows since at round i, ko-sub randomly pairs the inputs and only inputs in $\mathcal{X}_i^* \backslash \{x^*\}$ are able to eliminate x^*. Next we discuss $\Pr(|\mathcal{Y}_{m+1}^*| = 0)$. At round i, the probability that an input in $\mathcal{X}^*$ is not picked up in $\mathcal{Y}$ is

$$\frac{\binom{|\mathcal{X}_i| - |\mathcal{X}_i^*|}{n_1}}{\binom{|\mathcal{X}_i|}{n_1}} \le \left(1 - \frac{|\mathcal{X}_i^*|}{|\mathcal{X}_i|}\right)^{n_1} = (1 - \alpha_i)^{n_1}.$$

Therefore,

$$\begin{aligned}\Pr(|\mathcal{Y}^*_{m+1}| = 0) &\le \prod_{i=1}^{m}(1-\alpha_i)^{n_1} \\ &\le \min_i (1-\alpha_i)^{n_1} \\ &= (1-\alpha)^{n_1}.\end{aligned}$$

As a result,

$$\begin{aligned}\Pr(|\mathcal{X}^*_{m+1} \cup \mathcal{Y}^*_{m+1}| = 0) &= \Pr(|\mathcal{X}^*_{m+1}| = 0 \ \wedge \ |\mathcal{Y}^*_{m+1}| = 0) \\ &\le \Pr(x^* \notin \mathcal{X}^*_{m+1} \ \wedge \ |\mathcal{Y}^*_{m+1}| = 0) \\ &\le \max_\alpha \min\{\Pr(x^* \notin \mathcal{X}^*_{m+1}), \Pr(|\mathcal{Y}^*_{m+1}| = 0)\} \\ &\le \max_\alpha \min\left\{\alpha m, (1-\alpha)^{n_1}\right\} \\ &\overset{(a)}{\le} \max\left\{\alpha m, (1-\alpha)^{n_1}\right\}\Big|_{\alpha = \frac{\epsilon}{\log n}} \\ &= \max\left\{\frac{\epsilon m}{\log n}, \left(1 - \frac{\epsilon}{\log n}\right)^{n_1}\right\} \\ &\overset{(b)}{<} \epsilon,\end{aligned}$$

where (a) follows since the first argument of the min increases and the second argument decreases with α. Also, (b) follows since $m \le \log n$ and $n_1 = \left\lceil \frac{1}{\epsilon} \ln \frac{1}{\epsilon} \log n \right\rceil$.

So far, we have shown that with probability $1-\epsilon$, there exists a 1-approximation of x^* in $\mathcal{X}_{m+1} \cup \mathcal{Y}_{m+1}$. From Lemma 4, COMPL gives a 2-approximation of the maximum input. Consequently, with probability $1-\epsilon$, KO-MOD outputs a 3-approximation of x^*. ■

In Appendix A, we show that KO-MOD cannot output better than 3-approximation of x^* with constant probability.

5.2. Quick-select

Motivated by quick-sort, we propose a quick-select algorithm Q-SELECT that at each round compares all the inputs with a random pivot to provide stronger performance guarantees against the non-adaptive adversary.

input: $\mathcal{X}$
pick a pivot $x_p \in \mathcal{X}$ at random
compare x_p with all other inputs in $\mathcal{X}$
let $\mathcal{Y} \subset \mathcal{X} \backslash \{x_p\}$ be the multiset of inputs that beat x_p
output: if $\mathcal{Y} \ne \emptyset$ output $\mathcal{Y}$ otherwise output $\{x_p\}$

Algorithm QS-SUB - Subroutine for Q-SELECT and COMB

We show that Q-SELECT provides a 2-approximation with no error against both the adaptive and non-adaptive adversaries. To show this result, observe that x^* will only be

input: $\mathcal{X}$
    while $|\mathcal{X}| > 1$
        $\mathcal{X} \leftarrow$ QS-SUB$(\mathcal{X})$
    end while
output: the unique input in $\mathcal{X}$

Algorithm Q-SELECT - Quick-select

eliminated if a 1-approximation of x^* is chosen as pivot, and, therefore, only inputs that are 2-approximation of x^* will survive.

Lemma 7 $\mathcal{E}_n^{\text{Q-SELECT,adpt}}(2) = 0$.

Proof If the output is x^*, the lemma holds. Otherwise, x^* is discarded when it was chosen as a pivot or compared with a pivot. Let x_p be the pivot when x^* is discarded; hence, $x_p \geq x^* - 1$. By the algorithm's definition, all the surviving inputs are at least $x_p - 1 \geq x^* - 2$. ■

We now show that the expected query complexity of Q-SELECT against a non-adaptive adversary is at most $2n$. This result follows from the observation that the non-adaptive adversary fixes the comparison graph at the beginning, and hence a random pivot wins against half of the inputs in expectation. This idea is made rigorous in the proof of Lemma 8.

In Example 2 we show an instance for which Q-SELECT requires $\binom{n}{2}$ queries against the adaptive adversary.

Lemma 8 $q_n^{\text{Q-SELECT,non}} < 2n$.

Proof Recall that the non-adaptive adversary can be modeled as a complete directed graph where each node is an input and there is an edge from x to y if $\mathcal{C}(x, y) = x$. Let $\text{in}(x)$ be the in-degree of x in such a graph.

At round i, the algorithm chooses a pivot x_p at random and compares it to all the remaining inputs. By keeping the winners, $\max\{\text{in}(x_p), 1\}$ inputs will remain for the next round. As a result, we have the following recursion for non-adaptive adversaries:

$$\begin{aligned} q_n^{\text{Q-SELECT}} &= \mathbb{E}\left[Q_n^{\text{Q-SELECT}}\right] \\ &= n - 1 + \frac{1}{n}\sum_{i=1}^{n} \mathbb{E}\left[Q_{\text{in}(x_i)}^{\text{Q-SELECT}}\right] \\ &= n - 1 + \frac{1}{n}\sum_{i=1}^{n} q_{\text{in}(x_i)}^{\text{Q-SELECT}}. \end{aligned}$$

By (2),

$$\begin{aligned}
q_n^{\text{Q-SELECT,non}} &= \max_{\mathcal{C}\in\mathcal{C}_{\text{non}}} \max_{\mathcal{X}} q_n^{\text{Q-SELECT}} \\
&= \max_{\mathcal{C}\in\mathcal{C}_{\text{non}}} \max_{\mathcal{X}} \left[n-1+\frac{1}{n}\sum_{i=1}^{n} q_{\text{in}(x_i)}^{\text{Q-SELECT}} \right] \\
&\le n-1+\frac{1}{n}\sum_{i=1}^{n} \max_{\mathcal{C}\in\mathcal{C}_{\text{non}}} \max_{\mathcal{X}} q_{\text{in}(x_i)}^{\text{Q-SELECT}} \\
&= n-1+\frac{1}{n}\sum_{i=1}^{n} q_{\text{in}(x_i)}^{\text{Q-SELECT,non}},
\end{aligned} \tag{4}$$

where the inequality follows as the maximum of sums is at most the sum of maxima. We prove by strong induction that $q_n^{\text{Q-SELECT,non}} \le 2(n-1)$, which holds for $n=1$. Suppose it holds for all $n' < n$, then,

$$\begin{aligned}
q_n^{\text{Q-SELECT,non}} &\le n-1+\frac{1}{n}\sum_{i=1}^{n} q_{\text{in}(x_i)}^{\text{Q-SELECT,non}} \\
&\le n-1+\frac{1}{n}\sum_{i=1}^{n} 2\cdot \text{in}(x_i) \\
&= n-1+\frac{n(n-1)}{n} \\
&\le 2(n-1),
\end{aligned}$$

where the equality follows since the in-degrees sum to $\frac{n(n-1)}{2}$. ■

Lemma 8 shows that $q_n^{\text{Q-SELECT,non}} < 2n$. Next, we show a naive concentration bound for the query complexity of Q-SELECT. By Markov's inequality, for a non-adaptive adversary,

$$\Pr(Q_n^{\text{Q-SELECT}} > 4n) \le \frac{1}{2}.$$

Let k be an integer multiple of 4. Now suppose we run Q-SELECT, allowing kn queries. At each $4n$ queries, the Q-SELECT ends with probability $\ge \frac{1}{2}$. Therefore,

$$\Pr(Q_n^{\text{Q-SELECT}} > kn) \le 2^{-\frac{k}{4}}.$$

This naive bound is exponential in k. The next lemma shows a tighter super-exponential concentration bound on the query complexity of the algorithm beyond its expectation. We defer the proof to appendix B.

Lemma 9 *Let $k' = \max\{e, k/2\}$. For a non-adaptive adversary, Pr$(Q_n^{\text{Q-SELECT}} > kn) \le e^{-(k-k')\ln k'}$.*

While Q-SELECT has linear expected query complexity under the non-adaptive adversarial model, the following example suggested to us by Nelson (2015) shows that it has a quadratic query complexity against an adaptive adversary.

Example 2 *Let $\mathcal{X} = \{0, 0, \ldots, 0\}$. At each round, the adversary declares the pivot to be smaller than all the other inputs. Consequently, only the pivot is eliminated, and the query complexity is $\binom{n}{2}$.*

5.3. Knock-out and quick-select combination

KO-MOD has the benefit of reducing the number of inputs exponentially at each round and therefore maintaining a linear query-complexity while having only a 3-approximation guarantee. On the other side, Q-SELECT has a 2-approximation guarantee while it may require $\mathcal{O}(n^2)$ queries for some instances of inputs. In COMB we combine the benefits of these algorithms and avoid their shortcomings. By carefully repeating QS-SUB, we try to reduce the number of inputs by a fraction at each round and keep the largest element in the remaining set. If the number of inputs is not reduced by a fraction, most of them must be close to each other. Therefore, repeating the KO-SUB for a sufficient number of times and keeping the inputs with the higher number of wins will guarantee the reduction of the input size without making the approximation error worse. Our final algorithm COMB provides a 2-approximation of x^*, even against the adaptive-adversarial comparator and has linear query complexity. Therefore, an open question of Ajtai et al. (2015) is resolved.

input: $\mathcal{X}, \epsilon$
  $\beta_1 = 9$, $\beta_2 = 25$, $i = 0$
  while $|\mathcal{X}| > 1$
    $i = i + 1$ (i is the round)
    $n_i = |\mathcal{X}|$
    run $\mathcal{X} \leftarrow$ QS-SUB($\mathcal{X}$) for $\lfloor \beta_1 \log \frac{1}{\epsilon} \rfloor$ times
    $\mathcal{X}_i = \mathcal{X}$
    if $|\mathcal{X}_i| > \frac{2}{3} n_i$
      run KO-SUB on *fixed* $\mathcal{X}$ for $\left\lfloor \beta_2 \left(\frac{4}{3}\right)^i \log \frac{1}{\epsilon} \right\rfloor$ times
      if there exists an input with $> \frac{3}{4} \left\lfloor \beta_2 \left(\frac{4}{3}\right)^i \log \frac{1}{\epsilon} \right\rfloor$ wins
        let $\mathcal{X}$ be a multiset of inputs with $> \frac{3}{4} \left\lfloor \beta_2 \left(\frac{4}{3}\right)^i \log \frac{1}{\epsilon} \right\rfloor$ wins
      else
        let $\mathcal{X}$ be an input with highest number of wins
  end while
output: $\mathcal{X}$

Algorithm COMB - Knock-out and quick-select combination

We begin the algorithm's analysis with a few lemmas.

Lemma 10 *At each round $|\mathcal{X}|$ reduces by at least a third, namely, $n_{i+1} \leq \frac{2}{3} n_i$.*

Proof If at any round $|\mathcal{X}_i| \leq \frac{2}{3} n_i$, then the lemma holds, and the algorithm does not call KO-SUB. On the other hand, if KO-SUB is called, then by Markov's inequality at most two-thirds of the inputs win more than three-fourth of the queries. As a result, at round i, at least one-third of the inputs in $\mathcal{X}$ will be eliminated. ■

Recall that $\mathcal{X}^* = \{x \in \mathcal{X} : x \geq x^* - 1\}$. Lemma 11 shows that choosing inputs inside $\mathcal{X}^*$

as a pivot guarantees a 2-approximation of x^*. The proof is similar to Lemma 7 and is omitted.

Lemma 11 *If $x^* \in \mathcal{X}$, at a call to* QS-SUB *either x^* survives or a pivot from $\mathcal{X}^*$ is chosen where in the latter case, only inputs that are 2-approximation of x^* will survive.*

We showed that at each round, COMB reduces $|\mathcal{X}|$ by at least a third. As a result, the number of inputs decreases exponentially, and the total number of queries is linear in n. We also show that if x^* is eliminated at some round, then, with high probability, the pivot at that round is an input from $\mathcal{X}^*$. Using Lemma 11, this implies that COMB outputs a 2-approximation of x^* with high probability.

Theorem 12 $q_n^{\text{COMB,adpt}} = \mathcal{O}\left(n \log \frac{1}{\epsilon}\right)$ *and* $\mathcal{E}_n^{\text{COMB,adpt}}(2) < \epsilon$.

Proof We start by analyzing the query complexity of COMB. By Lemma 10,

$$n_i \le n \cdot \left(\tfrac{2}{3}\right)^{i-1}.$$

Therefore, the total number of queries at round i is at most

$$n\left(\tfrac{2}{3}\right)^{i-1} \beta_1 \log \tfrac{1}{\epsilon} + \tfrac{n}{2}\left(\tfrac{2}{3}\right)^{i-1} \beta_2 \left(\tfrac{4}{3}\right)^{i} \log \tfrac{1}{\epsilon},$$

where the first term is for calls to QS-SUB, and the second term is for calls to KO-SUB. Adding the query complexity of all the rounds,

$$\begin{aligned} q_n^{\text{COMB,adpt}} &\le n \log \tfrac{1}{\epsilon} \sum_{i=1}^{\infty} \left(\beta_1 \left(\tfrac{2}{3}\right)^{i-1} + \tfrac{2}{3}\beta_2 \left(\tfrac{8}{9}\right)^{i-1}\right) \\ &\le n(3\beta_1 + 6\beta_2) \log \tfrac{1}{\epsilon} \\ &= \mathcal{O}\left(n \log \tfrac{1}{\epsilon}\right). \end{aligned}$$

We now analyze the approximation guarantee of COMB. We show that at least one of the following events happens with probability greater than $1-\epsilon$.

- COMB outputs x^*.
- An input inside $\mathcal{X}^*$ is chosen as a pivot at some round.

Let $\mathcal{X}_i^* \stackrel{\text{def}}{=} \mathcal{X}_i \cap \mathcal{X}^*$ and $\alpha_i \stackrel{\text{def}}{=} \frac{|\mathcal{X}_i^*|}{|\mathcal{X}_i|}$. We consider the following two cases separately.

- **Case 1** There exists an i such that $|\mathcal{X}_i| > \frac{2}{3} n_i$ and $\alpha_i > \frac{1}{8}$.
- **Case 2** For all i, either $|\mathcal{X}_i| \le \frac{2}{3} n_i$ or $\alpha_i \le \frac{1}{8}$.

First, we consider Case 1. We show that in this case a pivot from $\mathcal{X}^*$ is chosen with probability $> 1-\epsilon$. Observe that at round i, $|\mathcal{X}|$ starts at $n_i < \frac{3}{2}|\mathcal{X}_i|$ and gradually decreases. On the other hand, in all the $\left\lfloor \beta_1 \log \frac{1}{\epsilon} \right\rfloor$ calls to QS-SUB, $|\mathcal{X} \cap \mathcal{X}^*|$ is at least $|\mathcal{X}_i^*| = \alpha_i |\mathcal{X}_i|$. Therefore, in all the calls to QS-SUB at round i,

$$\frac{|\mathcal{X} \cap \mathcal{X}^*|}{|\mathcal{X}|} \ge \frac{\alpha_i |\mathcal{X}_i|}{\frac{3}{2}|\mathcal{X}_i|} = \frac{2}{3}\alpha_i.$$

Let E be the event of not choosing a pivot from $\mathcal{X}^*$ at round i. As a result,

$$\begin{aligned}\Pr(E) &\leq \left(1 - \tfrac{2}{3}\alpha_i\right)^{\left\lfloor \beta_1 \log \frac{1}{\epsilon} \right\rfloor} \\ &\leq \left(\tfrac{11}{12}\right)^{8 \log \frac{1}{\epsilon}} \\ &< \epsilon.\end{aligned}$$

Therefore, in Case 1, with probability at least $1-\epsilon$, a pivot from $\mathcal{X}^*$ is chosen.

We now consider Case 2. By Lemma 11, during the calls to QS-SUB, either x^* survives or an input from $\mathcal{X}^*$ is chosen as a pivot. Therefore, we may only lose x^* without choosing a pivot from $\mathcal{X}^*$, if at some round i, $|\mathcal{X}_i| > \frac{2}{3}n_i$ and x^* wins less than three-fourth of its queries during the calls to KO-SUB.

Recall that in Case 2, if $|\mathcal{X}_i| > \frac{2}{3}n_i$ then $\alpha_i \leq \frac{1}{8}$. Observe that x^* wins against a random input in $\mathcal{X}_i$ with probability greater than $> 1-\alpha_i$, which is at least seven-eighths. Let E'_i be the event that x^* wins fewer than three-quarters of its queries at round i. By the Chernoff bound,

$$\begin{aligned}\Pr(E'_i) &\leq \exp\left(-\left\lfloor \beta_2 \left(\tfrac{4}{3}\right)^i \log \tfrac{1}{\epsilon}\right\rfloor \cdot D\left(\tfrac{3}{4}||\tfrac{7}{8}\right)\right) \\ &\leq \epsilon^{2\left(\frac{4}{3}\right)^i},\end{aligned}$$

where $D(p||q) \stackrel{\text{def}}{=} p \ln \frac{p}{q} + (1-p)\ln\frac{1-p}{1-q}$ is the Kullback-Leibler distance between Bernoulli distributed random variables with parameters p and q, respectively. Assuming $\epsilon < \frac{1}{2}$, the total probability of missing x^* without choosing a pivot form $\mathcal{X}^*$ is at most

$$\begin{aligned}\sum_{i=1}^{\infty} \Pr(E'_i) &\leq \sum_{i=1}^{\infty} \epsilon^{2\left(\frac{4}{3}\right)^i} \\ &< \epsilon.\end{aligned}$$

This shows that with probability $> 1-\epsilon$, either x^* survives or an input inside $\mathcal{X}^*$ is chosen as a pivot. The theorem follows from Lemma 11. ■

6. Application to density estimation

Our study of maximum selection with adversarial comparators was motivated by the following density estimation problem:

Given a *known* set $\mathcal{P}_\delta = \{p_1, \ldots, p_n\}$ of n distributions and k samples from an *unknown* distribution p_0, output a distribution $\hat{p} \in \mathcal{P}_\delta$ such that for a small constant $C > 1$ and with high probability,

$$\|\hat{p} - p_0\|_1 \leq C \cdot \min_{p \in \mathcal{P}_\delta} \|p - p_0\|_1 + o_k(1).$$

This problem was studied in Devroye and Lugosi (2001) who showed that for $n = 2$, the SCHEFFE-TEST, described below in pseudocode, takes k samples and, with probability $1-\varepsilon$, outputs a distribution $\hat{p} \in \mathcal{P}_\delta$ such that

$$||\hat{p} - p_0||_1 \leq 3 \cdot \min_{p \in \mathcal{P}_\delta} ||p - p_0||_1 + \sqrt{\frac{10 \log \frac{1}{\varepsilon}}{k}}. \tag{5}$$

input: distributions p_1 and p_2, k *i.i.d.* samples of unknown distribution p_0
let $\mathcal{S} = \{x : p_1(x) > p_2(x)\}$
let $p_1(\mathcal{S})$ and $p_2(\mathcal{S})$ be the probability mass that p_1 and p_2 assign to $\mathcal{S}$
let $\mu_{\mathcal{S}}$ be the frequency of samples in $\mathcal{S}$
output: if $|p_1(\mathcal{S}) - \mu_{\mathcal{S}}| \leq |p_2(\mathcal{S}) - \mu_{\mathcal{S}}|$ output p_1, otherwise output p_2

Algorithm SCHEFFE-TEST- Scheffe test for two distributions

SCHEFFE-TEST provides a factor-3 approximation with high probability. The algorithm, as stated in its pseudocode, requires computing $p_i(\mathcal{S})$ which can be hard since the distributions are not restricted. However, as noted in Suresh et al. (2014), the algorithm can be made to run in time linear in k. Devroye and Lugosi (2001) also extended SCHEFFE-TEST for $n > 2$. Their proposed algorithm for $n > 2$ runs SCHEFFE-TEST for each pair of distributions in $\mathcal{P}_\delta$ and outputs the distribution with the maximum wins, where a distribution is a winner if it is the output of SCHEFFE-TEST. This algorithm is referred to as the Scheffe tournament. They showed that this algorithm finds a distribution $\hat{p} \in \mathcal{P}_\delta$ such that

$$||\hat{p} - p_0||_1 \leq 9 \min_{p \in \mathcal{P}_\delta} ||p - p_0||_1 + o_k(1),$$

and the running time is clearly $\Theta(n^2 k)$—quadratic in the number of distributions.

Mahalanabis and Stefankovic (2008) showed that the optimal coefficients for the Scheffe algorithms are indeed 3 and 9 for $n = 2$ and $n > 2$, respectively. They proposed an algorithm with an improved factor-3 approximation for $n > 2$—still running in time $\Theta(n^2)$, however. They also proposed a linear-time algorithm, but it requires a preprocessing step that runs in time exponential in n.

Scheffe's method has been used recently to obtain nearly sample optimal algorithms for learning Poisson Binomial distributions (Daskalakis et al., 2012), and Gaussian mixtures (Daskalakis and Kamath, 2014; Suresh et al., 2014).

We now describe how our noisy comparison model can be applied to this problem to yield a linear-time algorithm with the same estimation guarantee as the Scheffe tournament. Our algorithm uses the Scheffe test as a subroutine. Given a sufficient number of samples, $k = \Theta(\log n)$, the small term in the RHS of (5) vanishes, and SCHEFFE-TEST outputs

$$\begin{cases} p_i & \text{if } ||p_i - p_0||_1 < \frac{1}{3} ||p_j - p_0||_1, \\ p_j & \text{if } ||p_j - p_0||_1 < \frac{1}{3} ||p_i - p_0||_1, \\ \text{unknown} & \text{otherwise.} \end{cases}$$

Let $x_i = -\log_3 ||p_i - p_0||_1$, then analogously to the maximum selection with adversarial noise in (1), SCHEFFE-TEST outputs

$$\begin{cases} \max\{x_i, x_j\} & \text{if } |x_i - x_j| > 1, \\ \text{unknown} & \text{otherwise.} \end{cases}$$

Given a fixed multiset of samples the tournament results are fixed; hence, this setup is identical to the non-adaptive adversarial comparators. In particular, with probability $1-\varepsilon$, our quick-select algorithm can find $\hat{p} \in \mathcal{P}_\delta$ such that

$$||\hat{p} - p_0||_1 \le 9 \cdot \min_{p \in \mathcal{P}_\delta} ||p - p_0||_1 ,$$

with running time $\Theta(nk)$. Next, we consider the combination of SCHEFFE-TEST and Q-SELECT in greater detail.

Theorem 13 *Combination of* SCHEFFE-TEST *and* Q-SELECT *algorithms, with probability* $1-\varepsilon$*, results in* $\hat{p}$ *such that*

$$||\hat{p} - p_0||_1 \le 9 \cdot \min_{p \in \mathcal{P}_\delta} ||p - p_0||_1 + 4\sqrt{\frac{10 \log \frac{\binom{n}{2}}{\varepsilon}}{k}}.$$

Proof Let

$$p^* \stackrel{\text{def}}{=} \operatorname*{argmin}_{p \in \mathcal{P}_\delta} ||p - p_0||_1 .$$

Using (5), for each p_i and p_j in $\mathcal{P}_\delta$, with probability $1 - \varepsilon/\binom{n}{2}$, SCHEFFE-TEST outputs $\hat{p}$ such that

$$||\hat{p} - p_0||_1 \le 3 \cdot \min_{p \in \{p_i, p_j\}} ||p - p_0||_1 + \sqrt{\frac{10 \log \frac{\binom{n}{2}}{\varepsilon}}{k}}. \tag{6}$$

By the union bound (6) holds for all p_i and p_j with probability at least $1-\varepsilon$. Similar to Lemma 7, if p^* is eliminated, then at some round, Q-SELECT has chosen p' as a pivot such that

$$\left|\left|p' - p_0\right|\right|_1 \le 3 \cdot ||p^* - p_0||_1 + \sqrt{\frac{10 \log \frac{\binom{n}{2}}{\varepsilon}}{k}}.$$

Now after choosing p' as a pivot, for any distribution p'' that survives,

$$\begin{aligned}\left|\left|p'' - p_0\right|\right|_1 &\le 3 \cdot \left|\left|p' - p_0\right|\right|_1 + \sqrt{\frac{10 \log \frac{\binom{n}{2}}{\varepsilon}}{k}} \\ &\le 9 \cdot ||p^* - p_0||_1 + 4\sqrt{\frac{10 \log \frac{\binom{n}{2}}{\varepsilon}}{k}}.\end{aligned}$$

■

7. Noisy sorting

7.1. Problem statement

In this section, we consider sorting with noisy comparators. The comparator model is the same as before, and the goal is to approximately sort the inputs in decreasing order.

Consider an Algorithm $\mathcal{A}$ for sorting the inputs. The output of $\mathcal{A}$ is denoted by $\mathbf{Y}_{\mathcal{A}}(\mathcal{X}) \stackrel{\text{def}}{=} (Y_1, Y_2, \ldots, Y_n)$, a particular ordering of the inputs. Similar to the maximum-selection problem, a t-approximation error is

$$\mathcal{E}_n^{\mathcal{A}}(t) \stackrel{\text{def}}{=} \Pr\left(\max_{i,j:i>j}(Y_i - Y_j) > t\right),$$

namely, the probability of Y_i appearing after Y_j in $\mathbf{Y}_{\mathcal{A}}$ while $Y_i - Y_j > t$. Note that our definitions for $\mathcal{E}_n^{\mathcal{A},\text{non}}(t)$, $\mathcal{E}_n^{\mathcal{A},\text{adpt}}(t)$, $q_n^{\mathcal{A},\text{adpt}}$, and $q_n^{\mathcal{A},\text{non}}$ hold the same as before.

In the following, we first revisit the complete tournament with a small modification for the sake of the sorting problem, and we show that, under the adaptive adversarial model, it has zero 2-approximation error and query complexity of $\binom{n}{2}$. We then discuss the quick-sort algorithm Q-SORT and show that it has zero 2-approximation error but with improved query complexity for the non-adaptive adversary. We apply the known bounds for the running time of the general quick-sort algorithm with n distinct inputs to find the query complexity of Q-SORT.

7.2. Complete tournament

The algorithm is similar to COMPL in Section 4.2.1, and we refer to it as COMPL-SORT. The only difference is in the output of the algorithm.

input: $\mathcal{X}$
compare all input pairs in $\mathcal{X}$, count the number of times each input wins
output: output the inputs in the order of their number of wins, breaking the ties randomly

Algorithm COMPL-SORT - Complete tournament

The following lemma—and its proof—is similar to Lemma 4, and, therefore, we skip the proof.

Lemma 14 $q_n^{\text{COMPL-SORT,adpt}} = \binom{n}{2}$ *and* $\mathcal{E}_n^{\text{COMPL-SORT,adpt}}(2) = 0$.

Next, we discuss an algorithm with improved query complexity.

7.3. Quick-sort

Quick-sort is a well-known algorithm and, here, is denoted by Q-SORT. The expected query complexity of quick-sort with noiseless comparisons and distinct inputs is

$$f(n) \stackrel{\text{def}}{=} 2n\ln n - (4 - 2\gamma)n + 2\ln n + \mathcal{O}(1), \tag{7}$$

where γ is Euler's constant (McDiarmid and Hayward, 1996). Note that $f(n)$ is a convex function of n.

In the rest of this section, we study the error guarantee of quick-sort and its query complexity in the presence of noise. In Lemma 15, we show that the error guarantee of quick-sort for our noise model is the same as the complete tournament, namely, it can sort the inputs with zero 2-approximation error. Next, in Lemma 16, we show that the expected query complexity of quick-sort with non-adaptive adversarial noise is at most its expected query complexity in the noiseless model.

Lemma 15 $\mathcal{E}_n^{\text{Q-SORT,adpt}}(2) = 0.$

Proof The proof is by contradiction. Suppose $x_i > x_j + 2$, but x_j appears before x_i in the output of quick-sort algorithm. Then there must have been a pivot x_p such that $\mathcal{C}(x_i, x_p) = x_p$ while $\mathcal{C}(x_j, x_p) = x_j$. Since $x_i > x_j + 2$ no such a pivot exists. ■

The quick-sort algorithm chooses a pivot randomly to divide the set of inputs into smaller-size sets. The optimal pivot for noiseless quick-sort is known to be the median of the inputs to balance the size of the remained sets. In fact, it is easy to show that if we choose the median of the inputs as the pivot, the query complexity of quick-sort reduces to less than $n \log n$. Observe that in a non-adaptive adversarial model, the probability of having balanced sets after choosing pivot increases. As a result, in Lemma 16, we show that the expected query complexity of quick-sort in the presence of noise is upper bounded by $f(n)$.

Lemma 16 $q_n^{\text{Q-SORT,non}} = f(n)$ *and is achieved when the queries are noiseless and the inputs are distinct.*

Proof Let $\text{in}(x)$ and $\text{out}(x)$ be the in-degree and out-degree of node x in the complete tournament respectively. For the noiseless comparator with distinct inputs, the in-degrees and out-degrees of inputs are permutation of $(0, 1, \ldots, n-1)$. We show that

$$\operatorname*{argmax}_{\mathcal{C} \in \mathcal{C}_{\text{non}}} \max_{\mathcal{X}} q_n^{\text{Q-SORT}},$$

is a comparator whose complete tournament in-degrees and out-degrees are permutations of $(0, 1, \ldots, n-1)$. For notational simplicity let $q_n = q_n^{\text{Q-SORT,non}}$. We have the following recursion for quick-sort similar to (4):

$$q_n \leq n - 1 + \frac{1}{n} \sum_{i=1}^{n} q_{\text{out}(x_i)} + q_{\text{in}(x_i)}. \tag{8}$$

By induction, we show that the solution to (8) is bounded above by $f(n)$, a convex function of n. The induction holds for $n = 0, 1$, and 2. Now suppose the induction holds for all $i < n$. Since $f(n)$ is a convex function of n and $\sum_i \text{in}(x_i) = \sum_i \text{out}(x_i) = \frac{n(n-1)}{2}$, the right hand side of (8) is maximized when the in-degrees and out-degrees take their extreme values, namely, when they are permutation of $(0, 1, \ldots, n-1)$. Plugging in these values, (8) is equivalent to,

$$\begin{aligned} q_n &\leq n - 1 + \frac{1}{n} \sum_{i=1}^{n} f(\text{in}(x_i)) + f(\text{out}(x_i)) \\ &\leq n - 1 + \frac{1}{n} \sum_{i=1}^{n} f(i-1) + f(n-i), \end{aligned}$$

where the solution to this recursion is $f(n)$, given in (7). Hence q_n is bounded above by $f(n)$, and the equality happens when the in-degrees and out-degrees are permutations of $(0, 1, \ldots, n-1)$. ■

Knuth (1998); Hennequin (1989); McDiarmid and Hayward (1992) show different concentration bounds for quick-sort. In particular, McDiarmid and Hayward (1992) show that the probability of the quick-sort algorithm requiring more comparisons than $(1+\epsilon)$ times its expected query complexity is $n^{-2\epsilon \ln\ln n + \mathcal{O}(\ln\ln\ln n)}$. Observe that for the non-adaptive adversarial model, the chance of a random pivot cutting the set of inputs into balanced sets increases. As a result, one can show that the analysis in McDiarmid and Hayward (1992) follows automatically. In particular, Lemmas 2.1 and 2.2 in McDiarmid and Hayward (1992), which are the basis of their analysis, are valid for our non-adaptive adversarial model. Therefore, their tight concentration bound for quick-sort algorithm can be applied to our non-adaptive adversarial model.

Acknowledgments

The authors would like to thank the editor and anonymous reviewers for their constructive comments, and Jelani Nelson for introducing the authors to the adaptive adversarial model. This work was supported in part by NSF grants CIF-1564355 and CIF-1619448. Part of this work was done while J.Acharya was at MIT, supported by MIT-Shell initiative.

Appendix A. For all $t < 3$, ko-mod cannot output a t-approximation

Example 3 shows that the modified knock-out algorithm cannot achieve better than 3-approximation of x^*.

Example 3 *Suppose $n-2$ is multiple of 3 and n is a large number. Let $\mathcal{X}$ be a random permutation of*

$$\{3, \underbrace{2, 2, \ldots, 2}_{\frac{n-2}{3}}, \underbrace{1, 1, \ldots, 1}_{\frac{n-2}{3}}, \underbrace{0, 0, \ldots, 0}_{\frac{n-2}{3}}, 0^*\}.$$

This multiset consists of an input with value zero but specified with 0^* since this input is going to behave differently from other 0s. Let the adversarial comparator be such that all 0s, except 0^*, and all 2s lose to all 1s, and 3 loses to all 2s. If two inputs of the same value get paired, one of them wins randomly (except in the case of 0^*). By the properties of comparator, it is obvious that any 2 will defeat all zeros, including 0^*. In order to prove our main claim, we make the following arguments and show that each of them happens with high probability:

- *Pr(input with value 3 is not present in the final multiset)> $\frac{3}{10}$*
- *Pr(input 0^* is present in the final multiset)> $\frac{1}{3}$*
- *With high probability, the fraction of 1s in the final multiset is close to 1*

Before proving each argument, we show why satisfying all the above statements are sufficient to prove our claim. Consider the final multiset; with high probability, it mainly consists of 1*s, and there are a small number of* 0*s and* 2*s. Moreover, with probability greater than* $\frac{1}{3} \times \frac{3}{10}$*, input with value* 3 *has been removed before reaching the final multiset, and* 0^* *has survived to reach the final multiset. Therefore, if we run algorithm* COMPL *on the final multiset, the input* 0^* *will have the most wins and be declared as the output. Hence for all* $t < 3$*, we have* $\mathcal{E}_n^{\text{KO-MOD,non}}(t) >$ *constant. Note that we did not try to optimize this constant.*

Now we show why each of the arguments above is true. Note that the reasoning made here is in expectation and assuming n *is sufficiently large. However, the concentration bounds for all these claims are straightforward and thus omitted.*

Lemma 17 *With high probability, the fraction of* 1*s in the final multiset is close to 1, and the fraction of* 0*s and* 2*s are very small.*

Proof *We calculate the expected number of* 0*s,* 1*s, and* 2*s at each step. Let* $f_i(j)$ *be the fraction of* j*'s at the end of step* i*. After each step, we lose an input with value* 1 *if and only if they are paired with each other. As a result, we have the following recursion:*

$$f_{i+1}(1) = 2 \cdot f_i(1) \left(\tfrac{f_i(1)}{2} + 1 - f_i(1) \right),$$

where the factor 2 *on the RHS of the recursion above is due to the fact that at each step we are reducing the number of inputs to half. Starting with* $f_0(1) = 1/3$*, we get the set of values* $\{1/3, 5/9, 65/81, 6305/6561 \sim 0.96, ...\}$ *for* $f_i(1)$*s. We can see that the ratio is approaching 1 very fast. More precisely, the fraction of* 0*s is decreasing quadratically since their only chance of survival is to get paired among themselves. As a result, after a couple of steps, the fraction of zeros is extremely small, and, henceforth, the only chance of survival for* 2*s becomes getting paired among themselves. Additionally their fraction is going to decrease quadratically afterward. As a result, more samples of* 1*s will be in the final* $\mathcal{Y}$ *with high probability.* ■

Lemma 18 *Pr(input with value* 3 *is not present in the final multiset)*$> \frac{3}{10}$.

Proof *The input with value* 3 *is going to be removed when it is compared against one of the* 2*s. There is a slight chance of it surviving if it is chosen randomly for being in the output. Thus, the probability of input* 3 *being removed from the multiset in the first round is*

$$\text{Pr(input 3 is being removed in the first round)} = \frac{n-2}{3n}\left(1 - \frac{n_1}{n}\right) > \frac{3}{10},$$

where $n_1 = \left\lceil \frac{1}{\epsilon} \ln \frac{1}{\epsilon} \log n \right\rceil$. ■

Lemma 19 *Pr(input* 0^* *is present in the final multiset)*$> \frac{1}{3}$.

Proof *Similar to the argument made in the proof of Lemma 17, we have the following recursion for* $f_i(2)$*.*

$$f_{i+1}(2) = 2 \cdot f_i(2) \left(\tfrac{f_i(2)}{2} + 1 - f_i(2) - f_i(1) \right)$$

Thus, we have $f_0(2) = 1/3$, $f_1(2) = 1/3$, $f_2(2) = 5/27$, $f_3(2) = 85/2187$. As we stated in the proof of Lemma 17, the expected fraction of 2s is decreasing quadratically and

$$Pr(0^* surviving) = (1 - \tfrac{1}{3})(1 - \tfrac{1}{3})(1 - \tfrac{5}{27})(1 - \tfrac{85}{2187}) \cdots > \frac{1}{3},$$

proving the lemma. ■

Appendix B. Proof of Lemma 9

Abbreviate $Q_n^{\text{Q-SELECT}}$ by Q_n. As in the Chernoff bound proof, for all $\lambda > 0$,

$$\Pr(Q_n > kn) \le \frac{\mathbb{E}[e^{\lambda Q_n}]}{e^{k\lambda n}}. \tag{9}$$

Let $\lambda = \frac{1}{n}\ln k'$ and $\Phi(i) \stackrel{\text{def}}{=} \mathbb{E}[e^{\lambda Q_i}]$. We prove by induction that $\Phi(i) \le e^{k'\lambda i}$. The induction holds for $i = 0$. Similar to (4), we have the following recursion for $\Phi(n)$:

$$\begin{aligned}\Phi(n) &\le \frac{e^{\lambda(n-1)}}{n}\sum_{j=1}^{n}\Phi(\text{in}(x_j))\\ &\le \frac{e^{\lambda n}}{n}\sum_{j=1}^{n}\Phi(\text{in}(x_j)).\end{aligned}$$

Since $\text{in}(x_j) < n$, using induction,

$$\frac{e^{\lambda n}}{n}\sum_{j=1}^{n}\Phi(\text{in}(x_j)) \le \frac{e^{\lambda n}}{n}\sum_{j=1}^{n} e^{k'\lambda \text{in}(x_j)}. \tag{10}$$

Observe that $e^{k'\lambda\text{in}(x_j)}$ is a convex function of $\text{in}(x_j)$, and $\sum_{j=1}^{n}\text{in}(x_j) = \frac{n(n-1)}{2}$. As a result, the RHS of (10) is maximized when the in-degrees take their extreme values, namely, any permutation of $(0, 1, \ldots, n-1)$. Therefore,

$$\begin{aligned}\frac{e^{\lambda n}}{n}\sum_{j=1}^{n} e^{k'\lambda\text{in}(x_j)} &\le \frac{e^{\lambda n}}{n}\sum_{j=0}^{n-1} e^{k'\lambda j}\\ &= \frac{e^{\lambda n}}{n}\frac{e^{k'\lambda n}-1}{e^{k'\lambda}-1}.\end{aligned}$$

Combining the above equations,

$$\Phi(n) \le \frac{e^{\lambda n}}{n}\frac{e^{k'\lambda n}-1}{e^{k'\lambda}-1}.$$

Similarly, by induction on $1 \le i < n$,

$$\Phi(i) \le \frac{e^{\lambda i}}{i}\frac{e^{k'\lambda i}-1}{e^{k'\lambda}-1}.$$

In Lemma 20 we show that for $1 \le i \le n$,

$$\frac{e^{\lambda i}}{i} \frac{e^{k'\lambda i} - 1}{e^{k'\lambda} - 1} \le e^{k'\lambda i}. \tag{11}$$

Therefore, $\Phi(i) \le e^{k'\lambda i}$ for $1 \le i \le n$, and, in particular, $\Phi(n) \le e^{k'\lambda n}$. Substituting $\mathbb{E}[e^{\lambda Q_n}] = \Phi(n)$ in (9),

$$\begin{aligned}\Pr(Q_n > kn) &\le \frac{e^{k'\lambda n}}{e^{k\lambda n}} \\ &= \frac{e^{k' \ln k'}}{e^{k \ln k'}} \\ &= e^{-(k-k')\ln k'}.\end{aligned}$$

This proves the lemma.
We now prove (11). Let $k' = \max\{e, \frac{k}{2}\}$ and $\lambda = \frac{1}{n} \ln k'$.

Lemma 20 *For all* $1 \le i \le n$, $\frac{e^{\lambda i}}{i} \frac{e^{k'\lambda i}-1}{e^{k'\lambda}-1} \le e^{k'\lambda i}$.

Proof It suffices to show that for all $0 < t \le n$,

$$f(t) \stackrel{\text{def}}{=} \frac{e^{\lambda t}}{t} \frac{1 - e^{-k'\lambda t}}{e^{k'\lambda} - 1} < 1.$$

Observe that

$$\lim_{t \to 0} f(t) = \frac{k'\lambda}{e^{k'\lambda} - 1} \le 1.$$

On the other hand,

$$\begin{aligned}f(n) &= \frac{e^{\lambda n}}{n} \frac{1 - e^{-k'\lambda n}}{e^{k'\lambda} - 1} \\ &\le \frac{k'}{n} \frac{1}{e^{k' \ln k'/n} - 1} \\ &\le \frac{k'}{n} \frac{n}{k' \ln k'} \\ &\le 1.\end{aligned}$$

Next, we show that $f(t)$ is convex. One can show that,

$$\ln \frac{1 - e^{-u}}{u},$$

is a convex function of u. As a result,

$$\ln \frac{1 - e^{-k'\lambda t}}{t},$$

is a convex function of t. Observe that $\ln e^{\lambda t}$ is also convex. Therefore,

$$\ln \frac{1 - e^{-k'\lambda t}}{t} + \ln e^{\lambda t},$$

is convex. As a result, logarithm of $f(t)$ is convex, and, therefore, $f(t)$ is convex.

We showed that $f(t)$ is convex, $f(t \to 0) \leq 1$, and $f(n) \leq 1$. Therefore, for all $0 < t \leq n$, $f(t) \leq 1$. ■

References

Jayadev Acharya, Ashkan Jafarpour, Alon Orlitksy, and Ananda Theertha Suresh. Sorting with adversarial comparators and application to density estimation. In *Proceedings of the 2014 IEEE International Symposium on Information Theory (ISIT)*, 2014.

Micah Adler, Peter Gemmell, Mor Harchol-Balter, Richard M. Karp, and Claire Kenyon. Selection in the presence of noise: The design of playoff systems. In *Proceedings of the Fifth Annual ACM-SIAM Symposium on Discrete Algorithms. 23-25 January 1994, Arlington, Virginia*, pages 564–572, 1994.

Gagan Aggarwal, S Muthukrishnan, Dávid Pál, and Martin Pál. General auction mechanism for search advertising. In *Proceedings of the 18th international conference on World wide web*, pages 241–250. ACM, 2009.

Miklós Ajtai, Vitaly Feldman, Avinatan Hassidim, and Jelani Nelson. Sorting and selection with imprecise comparisons. *ACM Trans. Algorithms*, 12(2):19:1–19:19, November 2015. ISSN 1549-6325.

Michael Ben-Or and Avinatan Hassidim. The bayesian learner is optimal for noisy binary search (and pretty good for quantum as well). In *Foundations of Computer Science, 2008. FOCS'08. IEEE 49th Annual IEEE Symposium on*, pages 221–230. IEEE, 2008.

Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs the method of paired comparisons. *Biometrika*, 39(3-4):324–345, 1952.

Mark Braverman and Elchanan Mossel. Noisy sorting without resampling. In *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms. January 20-22, 2008, San Francisco, California, USA*, pages 268–276, 2008.

Mark Braverman, Jieming Mao, and S Matthew Weinberg. Parallel algorithms for select and partition with noisy comparisons. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 851–862. ACM, 2016.

Róbert Busa-Fekete, Eyke Hüllermeier, and Balázs Szörényi. Preference-based rank elicitation using statistical models: The case of mallows. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1071–1079, 2014.

Xi Chen, Sivakanth Gopi, Jieming Mao, and Jon Schneider. Competitive analysis of the top-k ranking problem. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1245–1264. SIAM, 2017a.

Xi Chen, Yuanzhi Li, and Jieming Mao. An instance optimal algorithm for top-k ranking under the multinomial logit model. *arXiv preprint arXiv:1707.08238*, 2017b.

Constantinos Daskalakis and Gautam Kamath. Faster and sample near-optimal algorithms for proper learning mixtures of gaussians. In *Proceedings of the 27th Annual Conference on Learning Theory (COLT)*, 2014.

Constantinos Daskalakis, Ilias Diakonikolas, and Rocco A Servedio. Learning poisson binomial distributions. In *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 - 22, 2012*, pages 709–728, 2012.

Herbert Aron David. *The method of paired comparisons*, volume 12. Defence Technical Information Center Document, 1963.

Luc Devroye and Gabor Lugosi. *Combinatorial Methods in Density Estimation.* Springer - verlag, New York, 2001.

Ilias Diakonikolas, Gautam Kamath, Daniel Kane, Jerry Li, Ankur Moitra, and Alistair Stewart. Robust estimators in high dimensions without the computational intractability. *arXiv preprint arXiv:1604.06443*, 2016.

GÖSta Ekman. Weber's law and related functions. *The Journal of Psychology*, 47(2): 343–352, 1959.

Moein Falahatgar, Yi Hao, Alon Orlitsky, Venkatadheeraj Pichapati, and Vaishakh Ravindrakumar. Maxing and ranking with few assumptions. In *Advances in Neural Information Processing Systems*, pages 7060–7070, 2017a.

Moein Falahatgar, Alon Orlitsky, Venkatadheeraj Pichapati, and Ananda Theertha Suresh. Maximum selection and ranking under noisy comparisons. In *International Conference on Machine Learning*, pages 1088–1096, 2017b.

Moein Falahatgar, Ayush Jain, Alon Orlitsky, Venkatadheeraj Pichapati, and Vaishakh Ravindrakumar. The limits of maxing, ranking, and preference learning. In *International Conference on Machine Learning*, pages 1426–1435, 2018.

Uriel Feige, Prabhakar Raghavan, David Peleg, and Eli Upfal. Computing with noisy information. *SIAM Journal on Computing*, 23(5):1001–1018, 1994.

Pascal Hennequin. Combinatorial analysis of quicksort algorithm. *Informatique théorique et applications*, 23(3):317–333, 1989.

Richard M. Karp and Robert Kleinberg. Noisy binary search and its applications. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2007, New Orleans, Louisiana, USA, January 7-9, 2007*, pages 881–890, 2007.

Michael Kearns, Yishay Mansour, Dana Ron, Ronitt Rubinfeld, Robert E Schapire, and Linda Sellie. On the learnability of discrete distributions. In *Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing, 23-25 May 1994, Montréal, Québec, Canada*, pages 273–282, 1994.

Donald Ervin Knuth. *The art of computer programming: sorting and searching*, volume 3. Pearson Education, 1998.

R Duncan Luce. *Individual choice behavior: A theoretical analysis.* Courier Corporation, 2005.

Satyaki Mahalanabis and Daniel Stefankovic. Density estimation in linear time. In *Proceedings of the 21st Annual Conference on Learning Theory (COLT)*, pages 503–512, 2008.

Colin McDiarmid and Ryan Hayward. Strong concentration for quicksort. In *Proceedings of the Third Annual ACM/SIGACT-SIAM Symposium on Discrete Algorithms, 27-29 January 1992, Orlando, Florida*, pages 414–421, 1992.

Colin McDiarmid and Ryan B Hayward. Large deviations for quicksort. *journal of algorithms*, 21(3):476–507, 1996.

Sahand Negahban, Sewoong Oh, and Devavrat Shah. Iterative ranking from pair-wise comparisons. In *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*, pages 2483–2491, 2012.

Jelani Nelson. Personal communication. 2015.

Robin L Plackett. The analysis of permutations. *Applied Statistics*, pages 193–202, 1975.

Henry Scheffe. A useful convergence theorem for probability distributions. In *The Annals of Mathematical Statistics*, volume 18, pages 434–438, 1947.

Nihar Shah, Sivaraman Balakrishnan, Aditya Guntuboyina, and Martin Wainwright. Stochastically transitive models for pairwise comparisons: Statistical and computational issues. In *International Conference on Machine Learning*, pages 11–20, 2016.

Ananda Theertha Suresh, Alon Orlitsky, Jayadev Acharya, and Ashkan Jafarpour. Near-optimal-sample estimators for spherical gaussian mixtures. In *Advances in Neural Information Processing Systems*, pages 1395–1403, 2014.

Balázs Szörényi, Róbert Busa-Fekete, Adil Paul, and Eyke Hüllermeier. Online rank elicitation for plackett-luce: A dueling bandits approach. In *Advances in Neural Information Processing Systems*, pages 604–612, 2015.

Louis L Thurstone. A law of comparative judgment. *Psychological review*, 34(4):273, 1927.

Leslie G. Valiant. A theory of the learnable. In *Proceedings of the 16th Annual ACM Symposium on Theory of Computing, April 30 - May 2, 1984, Washington, DC, USA*, pages 436–445, 1984.

Yisong Yue and Thorsten Joachims. Beat the mean bandit. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 241–248, 2011.