Private Rank Aggregation in Central and Local Models

Daniel Alabi[*1], Badih Ghazi[2], Ravi Kumar[2], and Pasin Manurangsi[2]

[1]Harvard School of Engineering and Applied Sciences
[2]Google Research, Mountain View, CA
alabid@g.harvard.edu, {badihghazi, ravi.k53}@gmail.com, pasin@google.com

Abstract

In social choice theory, (Kemeny) rank aggregation is a well-studied problem where the goal is to combine rankings from multiple voters into a single ranking on the same set of items. Since rankings can reveal preferences of voters (which a voter might like to keep private), it is important to aggregate preferences in such a way to preserve privacy. In this work, we present differentially private algorithms for rank aggregation in the pure and approximate settings along with distribution-independent utility upper and lower bounds. In addition to bounds in the central model, we also present utility bounds for the local model of differential privacy.

Introduction

The goal of rank aggregation is to find a central ranking based on rankings of two or more voters. Rank aggregation is a basic formulation that is studied in diverse disciplines ranging from social choice (Arrow 1963), voting (Young and Levenglick 1978; Young 1995), and behavioral economics to machine learning (Wistuba and Pedapati 2020), data mining (Dwork et al. 2001), recommendation systems (Pennock, Horvitz, and Giles 2000), and information retrieval (Mansouri, Zanibbi, and Oard 2021). The problem has a long and rich algorithmic history (Conitzer, Davenport, and Kalagnanam 2006; Meila et al. 2007; Kenyon-Mathieu and Schudy 2007; Mandhani and Meila 2009; Schalekamp and van Zuylen 2009; Ailon, Charikar, and Newman 2008).

In many rank aggregation applications such as voting, it is imperative to find a central ranking such that the privacy of the voters who contributed the rankings is preserved. A principled way to achieve this is to view the problem through the lens of the rigorous mathematical definition of differential privacy (DP) (Dwork et al. 2006b,a). As in other privacy settings, it is then important to understand the trade-off between privacy and utility, i.e., the quality of the aggregation.

DP rank aggregation has been studied in a few recent papers (Hay, Elagina, and Miklau 2017; Yan, Li, and Liu 2020; Liu et al. 2020). While the algorithms in these works are guaranteed to be DP, their utility is provably good only under generative settings such as the Mallows model (Mallows 1957; Fligner and Verducci 1986) or with other distributional assumptions. The question we ask in this work is: can one circumvent such stylized settings and obtain a DP rank aggregation algorithm with provably good utility in the worst case? We answer this affirmatively by providing a spectrum of algorithms in the central and local DP models.

In the central model of DP, a trusted curator holds the *non-private* exact rankings from all users and aggregates them into a single ranking while ensuring privacy. In the local model of DP, this process would be done without collating the *exact* rankings of the users. As an example, consider a distributed recommendation system that can be used to determine a central ranking based on individual user rankings.

Background

Let $[m] = \{1, \ldots, m\}$ be the universe of items to be ranked (we assume this is public information) and let $\mathbb{S}_m$ be the group of rankings (i.e., permutations) on $[m]$. For example, the universe of items could be the set of all restaurants in New York. Let $\Pi = \{\pi_1, \ldots, \pi_n\}$ be a given set of rankings, where n is the number of users and each $\pi_k \in \mathbb{S}_m$ gives an ordering on the m items. We assume that the lower the position $\pi(j)$ of an item $j \in [m]$ in a ranking π, the higher our preference for that item j. For $i, j \in [m]$ such that $i \neq j$, define $w_{ij}^{\Pi} = \Pr_{\pi \sim \Pi}[\pi(i) < \pi(j)]$, i.e., w_{ij}^{Π} is the fraction of rankings that rank item i before j. As noted by Ailon, Charikar, and Newman (2008), for a graph on m nodes, when the weight of each edge (i, j) is w_{ij}^{Π}, rank aggregation reduces to the weighted version of the minimum Feedback Arc Set in Tournaments (FAST) problem.

Rank aggregation is based on the *Kendall tau* metric:

$$K(\pi_1, \pi_2) = |\{(i,j) : \pi_1(i) < \pi_1(j) \text{ but } \pi_2(i) > \pi_2(j)\}|,$$

for any two rankings $\pi_1, \pi_2 \in \mathbb{S}_m$. In other words, $K(\pi_1, \pi_2)$ is the number of pairwise disagreements between the two permutations π_1 and π_2. Note that $K(\pi, \pi) = 0$ for any permutation π and the maximum value of $K(\cdot, \cdot)$ is $\binom{m}{2}$. We also define the *average Kendall tau* distance (or the *Kemeny Score*) to a set $\Pi = \{\pi_1, \ldots, \pi_n\}$ of rankings as $K(\sigma, \Pi) = \frac{1}{n}\sum_{i=1}^{n} K(\sigma, \pi_i)$. We use $\mathrm{OPT}(\Pi)$ to denote $\min_\sigma K(\sigma, \Pi)$. We say that a randomized algorithm $\mathcal{A}$ obtains an (α, β)-*approximation* for the (Kemeny) rank aggregation problem if, given Π, it outputs σ such that

[*]Work done while an intern at Google Research.

$\mathbb{E}_\sigma[K(\sigma, \Pi)] \leq \alpha \cdot \mathrm{OPT}(\Pi) + \beta$, where α is the approximation ratio and β is the additive error. When $\beta = 0$, we call $\mathcal{A}$ an *α-approximation* algorithm.

Differential Privacy. We consider two sets $\Pi = \{\pi_1, \ldots, \pi_n\}, \Pi' = \{\pi'_1, \ldots, \pi'_n\}$ of rankings to be *neighboring* if they differ on a single ranking. In this work, we consider both the *central* and *local* models of DP. For $\epsilon > 0, \delta \geq 0$, a randomized algorithm $\mathcal{A} : (\mathbb{S}_m)^n \to \mathbb{S}_m$ is *(ϵ, δ)-differentially private (DP) in the central model* if for all neighboring sets Π, Π' of rankings and for all $S \subseteq \mathbb{S}_m$,

$$\Pr[\mathcal{A}(\Pi) \in S] \leq e^\epsilon \Pr[\mathcal{A}(\Pi') \in S] + \delta.$$

In other words, in the central model the algorithm $\mathcal{A}$ can access all the input rankings and only the output aggregated ranking is required to be private.

In the (interactive) local model of DP (Kasiviswanathan et al. 2011), the users retain their rankings and the algorithm $\mathcal{A}$ is a randomized protocol between the users; let $\mathcal{T}_\mathcal{A}$ denote the transcript of the protocol[1]. An algorithm $\mathcal{A}$ is (ϵ, δ)-*differentially private (DP) in the local model* if for all neighboring sets Π, Π' of rankings and for all sets S of transcripts,

$$\Pr[\mathcal{T}_\mathcal{A}(\Pi) \in S] \leq e^\epsilon \Pr[\mathcal{T}_\mathcal{A}(\Pi') \in S] + \delta.$$

We say that $\mathcal{A}$ is a pure-DP algorithm (denoted ϵ-DP) when $\delta = 0$ and is an approximate-DP algorithm otherwise.

Our Results

We obtain polynomial-time pure- and approximate-DP approximation algorithms for rank aggregation in the central and local models. Note that by using the exponential mechanism of (McSherry and Talwar 2007), one can obtain an algorithm with approximation ratio of 1 and additive error of $\tilde{O}(m^3/n)$; however this algorithm is computationally inefficient (Hay, Elagina, and Miklau 2017).

Our algorithms are based on two generic reductions. In our first reduction, we show that using standard DP mechanisms for aggregation (e.g., Laplace or Gaussian in the central model) to estimate the values of w_{ij}'s and then running an off-the-shelf approximation algorithm for rank aggregation (that is not necessarily private), preserves the approximation ratio and achieves additive errors of $O(m^4/n)$ for pure-DP in the central model, $O(m^3/n)$ for approximate-DP in the central model, and $O(m^3/\sqrt{n})$ in the local model. In the second reduction, we show how to improve the additive errors to $\tilde{O}(m^3/n)$, $\tilde{O}(m^{2.5}/n)$, and $\tilde{O}(m^{2.5}/\sqrt{n})$ respectively, while obtaining an approximation ratio of almost 5. This reduction utilizes the query pattern properties of the KwikSort algorithm of Ailon, Charikar, and Newman (2008), which only needs to look at $O(m \log m)$ entries w_{ij}'s. Roughly speaking, this means that we can add a smaller amount of noise per query, leading to an improved error bound. Our results are summarized in Tables 1 and 2.

We remark that the idea of adding a smaller amount of noise when the algorithm uses fewer queries of w_{ij}'s was also suggested and empirically evaluated for the KwikSort algorithm by Hay, Elagina, and Miklau (2017). However, they did not prove any theoretical guarantee of the algorithm. Furthermore, our algorithm differs from theirs when KwikSort exceeds the prescribed number of queries: ours outputs the DP ranking computed using the higher-error algorithm that noises all m^2 entries whereas theirs just outputs the sorted ranking so far where the unsorted part is randomly ordered. The latter is insufficient to get the additive error that we achieve because leaving even a single pair unsorted can lead to error as large as $\Omega(1)$; even if this event happens with a small probability of $1/m^{O(1)}$, the error remains at $\Omega(1/m^{O(1)})$, which does not converge to zero as $n \to \infty$.

In addition to our algorithmic contributions, we also prove lower bounds on the additive error that roughly match the upper bounds for $n \geq \tilde{O}(m)$ in the case of pure-DP and for $n \geq \tilde{O}(\sqrt{m})$ for approximate-DP, even when the approximation ratio is allowed to be large. Our lower bounds proceed by reducing from the 1-way marginal problem and utilizing existing lower bounds for pure- and approximate-DP for the 1-way marginals problem (Hardt and Talwar 2010; Bun, Ullman, and Vadhan 2018; Steinke and Ullman 2015; Dwork et al. 2015; Steinke and Ullman 2017).

	α	β	
ϵ-DP	$5 + \xi$	$\frac{m^3 \log m}{\epsilon n}$	(Corollary 6)
(ϵ, δ)-DP		$\frac{m^{2.5}\sqrt{\log m}}{\epsilon n}\sqrt{\log \frac{1}{\delta}}$	(Corollary 7)
ϵ-DP	$1 + \xi$	$\frac{m^4}{\epsilon n}$	(Corollary 3)
(ϵ, δ)-DP		$\frac{m^3}{\epsilon n}\sqrt{\log \frac{1}{\delta}}$	(Corollary 4)
ϵ-DP	1	$\frac{m^3}{\epsilon n} \log m$	(Hay et. al., 2017)

Table 1: Guarantees of different (α, β)-approximation algorithms in the *central* model; here ξ can be any positive constant. We drop the big-O notation in β for brevity. All of our algorithms run in $\mathrm{poly}(nm)$ time, whereas the exponential mechanism (Hay et. al., 2017) runs in $O(m! \cdot n)$ time.

	α	β	
ϵ-DP	$1 + \xi$	$\frac{m^3}{\epsilon\sqrt{n}}$	(Corollary 9)
ϵ-DP	$5 + \xi$	$\frac{m^{2.5} \log m}{\epsilon\sqrt{n}}$	(Corollary 12)

Table 2: Guarantees of (α, β)-approximation algorithms in the *local* model. As before, we drop the big-O notation in β.

Notation

Let $\mathbf{w}^\Pi$ denote the $[m] \times [m]$ matrix whose (i, j)th entry is w^Π_{ij}. Note that the average Kendall tau distance can also be written as

$$K(\sigma, \Pi) = \sum_{i,j \in [m]} \mathbb{1}[\sigma(i) \leq \sigma(j)] \cdot w^\Pi_{ji},$$

where $\mathbb{1}[\cdot]$ is the binary indicator function. Hence, we may write $K(\sigma, \mathbf{w}^\Pi)$ to denote $K(\sigma, \Pi)$ and $\mathrm{OPT}(\mathbf{w}^\Pi)$ to denote

[1]We refer the readers to (Duchi and Rogers 2019) for a more detailed formalization and discussion on interactivity in the local model.

$\mathrm{OPT}(\Pi)$. When σ or Π are clear from the context, we may drop them for brevity.

To the best of our knowledge, all known rank aggregation algorithms only use $\mathbf{w}$ and do not directly require the input rankings Π. The only assumption that we will make throughout is that the w_{ij}'s satisfy the *probability constraint* (Ailon, Charikar, and Newman 2008), which means that $w_{ij} \geq 0$ and $w_{ij} + w_{ji} = 1$ for all $i, j \in [m]$ where $i \neq j$. Again, the non-private algorithms (Ailon, Charikar, and Newman 2008; Kenyon-Mathieu and Schudy 2007) that we will use in this paper obtained approximation ratios under this assumption.

In fact, many algorithms do not even need to look at all of $\mathbf{w}$. Due to this, we focus on algorithms that allow *query* access to $\mathbf{w}$. Besides these queries, the algorithms do not access the input otherwise. Our generic reductions below will be stated for such algorithms.

Algorithms in the Central Model

In this section we present pure- and approximate-DP algorithms for rank aggregation in the central model of DP. Our algorithms follow from two generic reductions: for noising queries and for reducing the additive error.

Noising Queries. To achieve DP, we will have to add noise to the query answers. In this regard, we say that a query answering algorithm $\mathcal{Q}$ incurs *expected error* e if for every i, j the answer $\tilde{w}_{ij}$ returned by $\mathcal{Q}$ satisfies $\mathbb{E}[|\tilde{w}_{ij} - w_{ij}|] \leq e$. Furthermore, we assume that $\mathcal{Q}$ computes the estimate $\tilde{w}_{ij}$ for all $i, j \in [m]$ (non-adaptively) but only a subset of $\tilde{w}_{ij}$ is revealed when queried by the ranking algorithm; let $\tilde{\mathbf{w}}$ denote the resulting matrix of $\tilde{w}_{ij}$'s. In this context, we say that $\mathcal{Q}$ satisfies (ϵ, δ)-DP for q (adaptive) queries if $\mathcal{Q}$ can answer q such $\tilde{w}_{ij}$'s while respecting (ϵ, δ)-DP. In our algorithms, we only use $\mathcal{Q}$ that adds independent Laplace or Gaussian noise to each w_{ij} to get $\tilde{w}_{ij}$ but we state our reductions in full generality as they might be useful in the future.

Reduction I: Noising All Entries

We first give a generic reduction from a *not necessarily private* algorithm to DP algorithms. A simple form is:

Theorem 1. *Let $\alpha > 1, e, \epsilon > 0, q \in \mathbb{N}$, and $\delta \in [0, 1]$. Suppose that there exists a polynomial-time (not necessarily private) α-approximation algorithm $\mathcal{A}$ for the rank aggregation problem that always makes at most q queries. Furthermore, suppose that there exists a polynomial-time query answering algorithm $\mathcal{Q}$ with expected error e that is (ϵ, δ)-DP for answering at most q queries.*

Then, there exists a polynomial-time (ϵ, δ)-DP $(\alpha, (\alpha + 1)m^2 e)$-approximation algorithm $\mathcal{B}$ for rank aggregation.

This follows from an easy fact below that the error in the cost is at most the total error from querying all pairs of i, j.

Fact 2. *For any $\tilde{\mathbf{w}}, \mathbf{w}$ and $\sigma \in \mathbb{S}_m$, we have $|K(\sigma, \mathbf{w}) - K(\sigma, \tilde{\mathbf{w}})| \leq \sum_{i,j \in [m]} |w_{ji} - \tilde{w}_{ji}|$.*

Proof of Theorem 1. Our algorithm $\mathcal{B}$ simply works by running $\mathcal{A}$, and every time $\mathcal{A}$ queries for a pair i, j, it returns the answer using the algorithm $\mathcal{Q}$. The output ranking σ is simply the output from $\mathcal{A}$. Since only $\mathcal{Q}$ is accessing the input directly and from our assumption, it immediately follows that $\mathcal{B}$ is (ϵ, δ)-DP.

Since we may view $\mathcal{A}$ as having the input instance $\tilde{\mathbf{w}}$ (obtained by querying $\mathcal{Q}$), the α-approximation guarantee of $\mathcal{A}$ implies that

$$\mathbb{E}_{\sigma}[K(\sigma, \tilde{\mathbf{w}})] \leq \alpha \cdot \mathrm{OPT}(\tilde{\mathbf{w}}). \tag{1}$$

By applying Fact 2 twice, we then arrive at

$$\begin{aligned}
&\mathbb{E}_{\sigma,\tilde{\mathbf{w}}}[K(\sigma, \mathbf{w})] \\
\text{(Fact 2)} &\leq m^2 e + \mathbb{E}_{\sigma,\tilde{\mathbf{w}}}[K(\sigma, \tilde{\mathbf{w}})] \\
(4) &\leq m^2 e + \alpha \cdot \mathrm{OPT}(\tilde{\mathbf{w}}) \\
\text{(Fact 2)} &\leq m^2 e + \alpha \cdot (\mathrm{OPT}(\mathbf{w}) + m^2 e) \\
&= \alpha \cdot \mathrm{OPT}(\mathbf{w}) + (\alpha + 1)m^2 e. \qquad \square
\end{aligned}$$

The above reduction itself can already be applied to several algorithms, although it does not yet give the optimal additive error. As an example, we may use the (non-private) PTAS of Kenyon-Mathieu and Schudy (2007); the PTAS requires *all* w_{ij}'s meaning that $q \leq m^2$ and thus we may let $\mathcal{Q}$ be the algorithm that adds $\mathrm{Lap}\left(0, \frac{m^2}{\epsilon n}\right)$ noise to each query[2]. This immediately implies the following:

Corollary 3. *For any $\xi, \epsilon > 0$, there exists an ϵ-DP $\left(1 + \xi, O_{\xi}\left(\frac{m^4}{\epsilon n}\right)\right)$-approximation algorithm for the rank aggregation problem in the central model.*

Similarly, if we instead add Gaussian noise with standard deviation $O\left(\frac{m}{\epsilon n}\sqrt{\log \frac{1}{\delta}}\right)$ to each query, we arrive at the following result:

Corollary 4. *For any $\xi, \epsilon, \delta > 0$, there exists an (ϵ, δ)-DP $\left(1 + \xi, O_{\xi}\left(\frac{m^3}{\epsilon n}\sqrt{\log \frac{1}{\delta}}\right)\right)$-approximation algorithm for the rank aggregation problem in the central model.*

Reduction II: Improving the Additive Error

A drawback of the reduction in Theorem 1 is that it requires the algorithm to always make at most q queries. This is a fairly strong condition and it cannot be applied, e.g., to QuickSort-based algorithms that we will discuss below. We thus give another reduction that works even when the algorithm makes at most q queries *with high probability*. The specific guarantees are given below.

Theorem 5. *Let $\alpha, e, \epsilon > 0, q \in \mathbb{N}$, and $\zeta, \delta \in [0, 1]$. Suppose that there exists a polynomial-time (not necessarily private) α-approximation algorithm $\mathcal{A}$ for rank aggregation that, with probability $1 - \frac{\zeta}{m^4}$, makes at most q queries. Furthermore, suppose that there exists a polynomial-time query answering algorithm $\mathcal{Q}$ with expected error e that is $(\epsilon/2, \delta)$-DP for answering at most q queries.*

Then, there exists a polynomial-time (ϵ, δ)-DP $(\alpha + \zeta, \beta)$-approximation algorithm $\mathcal{B}$ for the rank aggregation problem where $\beta = O_{\alpha}\left(m^2 e + \frac{1}{\epsilon n}\right)$.

[2] More precisely, to satisfy the probability constraints, we actually add the noise to each w_{ij} only for all $i < j$ and clip it to be between 0 and 1 to arrive at $\tilde{w}_{ij}$. We then let $\tilde{w}_{ji} = 1 - \tilde{w}_{ij}$.

Proof of Theorem 5. Our algorithm $\mathcal{B}$ simply works by running $\mathcal{A}$, and every time $\mathcal{A}$ queries for a pair i, j, it returns the answer using the algorithm $\mathcal{Q}$. If $\mathcal{A}$ ends up using at most q queries, then it returns the output $\sigma^{\mathcal{A}}$ of $\mathcal{A}$. Otherwise, it runs the $\epsilon/2$-DP algorithm from Corollary 3 with $\xi = 1$ and return its output σ^*. The (ϵ, δ)-DP guarantee of the algorithm is immediate from the assumptions and the basic composition property of DP (see, e.g., Dwork and Roth (2014)).

We next analyze the expected Kendall tau distance of the output. To do this, let $\mathscr{E}_q$ denote the event that $\mathcal{A}$ uses more than q queries. From our assumption, we have $\Pr[\mathscr{E}_q] \leq \zeta/m^4$. Furthermore, observe that

$$\mathbb{E}_{\sigma^{\mathcal{A}}}[K(\sigma^{\mathcal{A}}, \tilde{\mathbf{w}})] \geq \Pr[\mathscr{E}_q] \cdot \mathbb{E}_{\sigma^{\mathcal{A}}}[K(\sigma^{\mathcal{A}}, \tilde{\mathbf{w}}) \mid \mathscr{E}_q]. \quad (2)$$

Now, let σ denote the output of our algorithm $\mathcal{B}$. We have

$$\begin{aligned}
&\mathbb{E}_{\sigma,\tilde{\mathbf{w}}}[K(\sigma, \tilde{\mathbf{w}}))] \\
&\leq \Pr[\mathscr{E}_q] \cdot \mathbb{E}_{\sigma^{\mathcal{A}},\tilde{w}}[K(\sigma^{\mathcal{A}}, \mathbf{w}) \mid \mathscr{E}_q] + \Pr[\neg\mathscr{E}_q] \cdot \mathbb{E}_{\sigma^*}[K(\sigma^*, \mathbf{w})] \\
&\leq \mathbb{E}_{\sigma^{\mathcal{A}}}[K(\sigma^{\mathcal{A}}, \tilde{\mathbf{w}})] + \frac{\zeta}{m^4} \cdot \left(2 \cdot \mathrm{OPT}(\mathbf{w}) + O_\alpha\left(\frac{m^4}{\epsilon n}\right)\right) \\
&\leq \alpha \cdot \mathrm{OPT}(\mathbf{w}) + O_\alpha(m^2 e) + \frac{2\zeta}{m^4} \cdot \mathrm{OPT}(\mathbf{w}) + O_\alpha\left(\frac{1}{\epsilon n}\right) \\
&\leq (\alpha + \zeta) \cdot \mathrm{OPT}(\mathbf{w}) + O_\alpha\left(m^2 e + \frac{1}{\epsilon n}\right),
\end{aligned}$$

where the second inequality follows from (2) and the approximation guarantee of the algorithm from Corollary 3, the third inequality follows from similar computations as in the proof of Theorem 1, and the last inequality follows because we may assume w.l.o.g. that $m \geq 2$. □

Concrete Bounds via KwikSort. Ailon, Charikar, and Newman (2008) devise an algorithm *KwikSort*, inspired by QuickSort, to determine a ranking on m candidates and they show that this algorithm yields a 5-approximation for rank aggregation. It is known in the classic sorting literature that with high probability, QuickSort (and by extension, KwikSort) performs $O(m \log m)$ comparisons on m candidates to order these m items. Specifically, with probability at least $1 - \xi/m^4$, the number of comparisons between items would be at most $q = O_\xi(m \log m)$. Plugging this into Theorem 5 where $\mathcal{Q}$ adds $\mathrm{Lap}\left(0, \frac{q}{\epsilon n}\right)$ noise to each query, we get:

Corollary 6. *For any $\xi, \epsilon > 0$, there is an ϵ-DP $\left(5 + \xi, O_\xi\left(\frac{m^3 \log m}{\epsilon n}\right)\right)$-approximation algorithm for the rank aggregation problem in the central model.*

If we use Gaussian noise with standard deviation $O\left(\frac{\sqrt{q}}{\epsilon n}\sqrt{\log \frac{1}{\delta}}\right)$ instead of the Laplace noise, then we get:

Corollary 7. *For any $\xi, \epsilon, \delta > 0$, there exists an (ϵ, δ)-DP $\left(5 + \xi, O_\xi\left(\frac{m^{2.5}\sqrt{\log m}}{\epsilon n}\sqrt{\log \frac{1}{\delta}}\right)\right)$-approximation algorithm for the rank aggregation problem in the central model.*

Although the approximation ratios are now a larger constant (arbitrarily close to 5), the above two corollaries improve upon the additive errors in Corollaries 3 and 4 by a factor of $\tilde{\Theta}(m)$ and $\tilde{\Theta}(\sqrt{m})$ respectively.

For concreteness, we present the idea in Corollaries 6 and 7 as Algorithm 1. We use a global counter c to keep track of how many comparisons have been done. Once $c > q$ (which happens with negligible probability when we set $q = \Omega(m \log m)$), we default to using a PTAS with privacy budget of $\epsilon/2$ (Corollary 3). If $\delta > 0$, we let $\mathcal{N}$ be the Gaussian noise with standard deviation $O_\xi\left(\frac{\sqrt{m \log m}}{\epsilon n}\sqrt{\log \frac{1}{\delta}}\right)$ and if $\delta = 0$, we let $\mathcal{N}$ be drawn from $\mathrm{Lap}\left(0, O_\xi\left(\frac{m \log m}{\epsilon n}\right)\right)$.

Algorithm 1: DPKwikSort.

```
Parameters: $\epsilon, \delta \in (0, 1]$, $q \in \mathbb{N}$
Input: $\Pi = \{\pi_1, \ldots, \pi_n\}$, $\mathcal{U} \subseteq [m]$, $c \leftarrow 0$
if $\mathcal{U} = \emptyset$ then
    return $\emptyset$
$\mathcal{U}_L, \mathcal{U}_R \leftarrow \emptyset$
Pick a pivot $i \in \mathcal{U}$ uniformly at random
for $j \in \mathcal{U} \setminus \{i\}$ do
    $c \leftarrow c + 1$
    if $c > q$ then
        Return PTAS with privacy budget of $\epsilon/2$ as
          final output                        ▷ See Corollary 3
    $\tilde{w}_{ji} \leftarrow w^{\Pi}_{ji} + \mathcal{N}$              ▷ See text
    if $\tilde{w}_{ji} > 0.5$ then
        Add $j$ to $\mathcal{U}_L$
    else
        Add $j$ to $\mathcal{U}_R$
return DPKwikSort($\Pi, \mathcal{U}_L, c$), $i$, DPKwikSort($\Pi, \mathcal{U}_R, c$)
```

Algorithms in the Local Model

We next consider the local model (Kasiviswanathan et al. 2011), which is more stringent than the central model. We only focus on pure-DP algorithms in the local model since there is a strong evidence[3] that approximate-DP does not help improve utility in the local model (Bun, Nelson, and Stemmer 2019; Joseph et al. 2019).

Reduction I: Noising All Entries

Similar to our algorithms in the central model, we start with a simple reduction that works for any non-private ranking algorithm by noising each w_{ji}. This is summarized below.

Theorem 8. *Let $\alpha > 1, \epsilon > 0$, and $\delta \in [0, 1]$. Suppose that there exists a polynomial-time (not necessarily private) α-approximation algorithm $\mathcal{A}$ for the rank aggregation problem. Then, there exists a polynomial-time ϵ-DP $\left(\alpha, O_\alpha\left(\frac{m^3}{\epsilon\sqrt{n}}\right)\right)$-approximation algorithm $\mathcal{B}$ for the rank aggregation problem in the local model.*

The proof of Theorem 8 is essentially the same as that of Theorem 1 in the central model, except that we use the

[3]Bun, Nelson, and Stemmer (2019); Joseph et al. (2019) give a generic transformation from any *sequentially interactive* local approximate-DP protocols to a pure-DP one while retaining the utility. However, this does not apply to the full interaction setting.

query answering algorithm of Duchi, Jordan, and Wainwright (2014), which can answer all the m^2 entries with expected error of $O\left(\frac{m}{\epsilon\sqrt{n}}\right)$ per entry; this indeed results in $O\left(\frac{m^3}{\epsilon\sqrt{n}}\right)$ additive error in the above theorem. We remark that if one were to naively use the randomized response algorithm on each entry, the resulting error would be $O\left(\frac{m^2}{\epsilon\sqrt{n}}\right)$ per entry; in other words, Duchi, Jordan, and Wainwright (2014) saves a factor of $\Theta(m)$ in the error. The full proof of Theorem 8 is deferred to the Supplementary Material (SM).

Plugging the PTAS of Kenyon-Mathieu and Schudy (2007) into Theorem 8, we arrive at the following:

Corollary 9. *For any $\xi, \epsilon > 0$, there exists an ϵ-local DP $\left(1+\xi, O_\xi\left(\frac{m^3}{\epsilon\sqrt{n}}\right)\right)$-approximation algorithm for the rank aggregation problem in the local model.*

Reduction II: Improving the Additive Error

Similar to the algorithm in the central model, intuitively, it should be possible to reveal only the required queries while adding a smaller amount of noise. Formalizing this, however, is more challenging because the algorithm of Duchi, Jordan, and Wainwright (2014) that we use is *non-interactive* in nature, meaning that it reveals the information about the entire vector at once, in contrast to the Laplace or Gaussian mechanisms for which we can add noise and reveal each entry as the algorithm progresses. Fortunately, Bassily (2019) provides an algorithm that has accuracy similar to Duchi, Jordan, and Wainwright (2014) but also works in the adaptive setting. However, even Bassily's algorithm does not directly work in our setting yet: the answers of latter queries depend on that of previous queries and thus we cannot define $\tilde{\mathbf{w}}$ directly as we did in the proof of Theorem 5.

Modifying Bassily's Algorithm. We start by modifying the algorithm of Bassily (2019) so that it fits in the "query answering algorithm" definition we used earlier in the central model. However, while the previous algorithms can always support up to q queries, the new algorithm will only be able to do so with high probability. That is, it is allowed to output $\perp$ with a small probability. This is formalized below.

Lemma 10. *For any $\epsilon, \zeta > 0$ and $q > 10m \log m$, there exists an ϵ-DP query-answering algorithm $\mathcal{Q}$ in the local model such that*

- *If we consider $\perp$ as having zero error, its expected error per query is at most $O\left(\frac{q}{\epsilon\sqrt{nm}}\right)$.*
- *For any (possibly adaptive) sequence $i_1 j_1, \dots, i_q j_q$ of queries, the probability that $\mathcal{Q}$ outputs $\perp$ on any of the queries is at most* $\exp(-0.1q/m)$.

For simplicity of the presentation, we assume that n is divisible by m. We remark that this is without loss of generality as otherwise we can imagine having "dummy" $t < m$ users so that $n + t$ is divisible by m, where these dummy users do not contribute to the queries.

The algorithm is simple: it randomly partitions the n users into sets $\mathcal{P}_1, \dots, \mathcal{P}_m$. Then, on each query ji, it uses a *random* set to answer the query (via the *Randomized Response* mechanism (Warner 1965)) with ϵ_0-DP where $\epsilon_0 = 0.5\epsilon m/q$. Finally, the algorithm outputs $\perp$ if that particular set has already been used at least $2q/m$ times previously.

We remark that the main difference between the original algorithm of Bassily (2019) and our version is that the former partitions the users into q sets and use the tth set to answer the tth query. This unfortunately means that an answer to a query ji may depend on which order it was asked, which renders our utility analysis of DP ranking algorithms invalid because $\tilde{\mathbf{w}}$ is not well defined. Our modification overcomes this issue since the partition used for each ji (denoted by $\ell(j,i)$ below) is independently chosen among the m sets.

Proof of Lemma 10. The full algorithm is described in Algorithm 2; here $(\mathcal{P}_1, \dots, \mathcal{P}_m)$ is a random partition where each set consists of n/m users and $\ell(j,i)$ is i.i.d. uniformly at random from $[m]$. Moreover, c is a vector of counters, where $c(p)$ is initialized to zero for all $p \in [m]$.

Algorithm 2: ϵ-DP Adaptive Query Answering Algorithm in the Local Model.

```
Parameters: $\epsilon > 0$, $q \in \mathbb{N}$, Partition $(\mathcal{P}_1, \dots, \mathcal{P}_m)$
  of $[n]$, $\ell : [m] \times [m] \to [m]$
Input: $\Pi = \{\pi_1, \dots, \pi_n\}, c : [m] \to \mathbb{Z}$
$\epsilon_0 \leftarrow 0.5\epsilon m/q$
$d_\epsilon \leftarrow \frac{e^{\epsilon_0}+1}{e^{\epsilon_0}-1}$
for user $k \in \mathcal{P}_{\ell(j,i)}$ do
    $c(\ell(j,i)) \leftarrow c(\ell(j,i)) + 1$
    if $c(\ell(j,i)) > 2q/m$ then
        return $\perp$
    $w^k_{ji} = \mathbb{1}[\pi_k(j) < \pi_k(i)]$
    Let $\tilde{w}^k_{ji} \leftarrow \begin{cases} d_\epsilon & \text{w.p. } 1/2 \cdot (1 + w^k_{ji}/d_\epsilon) \\ -d_\epsilon & \text{w.p. } 1/2 \cdot (1 - w^k_{ji}/d_\epsilon) \end{cases}$
return $\tilde{w}_{ji} := \frac{m}{n} \cdot \sum_{k \in \mathcal{P}_{\ell(j,i)}} \tilde{w}^k_{ji}$
```

We will now prove the algorithm's privacy guarantee. Let us consider a user k; suppose that this user belongs to $\mathcal{P}_\ell$. By definition of the algorithm, this user applies the Randomized Response algorithm at most $2q/m$ times throughout the entire run. Since each application is ϵ_0-DP (see, e.g., (Kasiviswanathan et al. 2011)), basic composition of DP ensures that the entire algorithm is $(2q/m) \cdot \epsilon_0 = \epsilon$-DP.

We next analyze its accuracy guarantee. Fix a query ji. For every user $k \in [n]$, let B_k denote $\mathbb{1}[k \in \mathcal{P}_{\ell(j,i)}]$. Notice that $\tilde{w}_{ji} = \sum_{k\in[n]} \frac{m}{n} \cdot B_k \cdot \tilde{w}^k_{ji}$ and $\mathbb{E}[B_k \cdot \tilde{w}^k_{ji}] = \frac{1}{m} \cdot w^k_{ji}$. Thus, we have $\mathbb{E}[\tilde{w}_{ji}] = w_{ji}$. Furthermore, we have

$$\mathrm{Var}(\tilde{w}_{ji}) = \left(\frac{m}{n}\right)^2 \cdot \left(\sum_{k\in[n]} \mathrm{Var}(B_k \cdot \tilde{w}^k_{ji}) + \sum_{1\le k<k'\le n} \mathrm{Cov}(B_k \cdot \tilde{w}^k_{ji}, B_{k'} \cdot \tilde{w}^{k'}_{ji}) \right). \quad (3)$$

Observe that $\mathrm{Var}(B_k \cdot \tilde{w}^k_{ji}) \le \mathbb{E}[(B_k \cdot \tilde{w}^k_{ji})^2] = d_\epsilon^2/m = O\left(\frac{1}{m\epsilon_0^2}\right) = O\left(\frac{q^2}{m^3\epsilon^2}\right)$. Moreover, since $\mathcal{P}_\ell$ is a random subset of n/m users, B_k and $B_{k'}$ are negatively correlated, meaning that $\mathrm{Cov}(B_k \cdot \tilde{w}^k_{ji}, B_{k'} \cdot \tilde{w}^{k'}_{ji}) \leq 0$. Plugging these back into (3) yields $\mathrm{Var}(\tilde{w}_{ji}) \leq O\left(\frac{q^2}{nm\epsilon^2}\right)$. This, together with the fact that $\tilde{w}_{ji}$ is an unbiased estimator of w_{ji}, implies that $\mathbb{E}[|\tilde{w}_{ji} - w_{ji}|] \leq O\left(\frac{q}{\epsilon\sqrt{nm}}\right)$ as desired.

Finally, we bound the probability that the algorithm outputs $\perp$. Since the $\ell(j,i)$'s are i.i.d. drawn from $[m]$, we can apply the Chernoff bound and the union bound (over the m partitions) to conclude that the probability that any sequence of q queries will trigger the algorithm to return $\perp$ is at most $m \cdot \exp(-q/(3m)) \leq \exp(-0.1q/m)$, where the inequality follows from our assumption that $q \geq 10m\log m$. □

The Reduction. Having described and analyzed the modified version of Bassily's algorithm, we can now describe the main properties of the second reduction in the local model:

Theorem 11. *Let $\alpha, \epsilon > 0, q \in \mathbb{N}$, and $\zeta, \delta \in [0,1]$ such that $q \geq 10m\log(m/\zeta)$. Suppose that there exists a polynomial-time (not necessarily private) α-approximation algorithm $\mathcal{A}$ for the rank aggregation problem that with probability $1 - \frac{\zeta}{m^4}$ makes at most q queries.*

Then, there is a polynomial-time ϵ-DP $(\alpha + \zeta, \beta)$-approximation algorithm $\mathcal{B}$ for the rank aggregation problem where $\beta = O_\alpha\left(\frac{m^{1.5}q}{\epsilon\sqrt{n}}\right)$ in the local model.

The proof of Theorem 11 follows its counterpart in the central model (Theorem 5); the main difference is that, instead of stopping after q queries previously, we stop upon seeing $\perp$ (in which case we run the PTAS from Corollary 9). The full proof is deferred to SM.

As in the central model, if we apply the concrete bounds for the KwikSort algorithm (which yields 5-approximation and requires $O(m\log m)$ queries w.h.p.) to Theorem 11, we arrive at the following corollary:

Corollary 12. *For any $\xi, \epsilon > 0$, there is an ϵ-DP $\left(5 + \xi, O_\xi\left(\frac{m^{2.5}\log m}{\epsilon\sqrt{n}}\right)\right)$-approximation algorithm for the rank aggregation problem in the local model.*

For concreteness, we also provide the full description in Algorithm 3. Again, similar to the setting of Algorithm 2, we pick $(\mathcal{P}_1, \ldots, \mathcal{P}_m)$ to be a random partition where each set consists of n/m users and $\ell(j,i)$ i.i.d. uniformly at random from $[m]$, and we initialize $c(p) = 0$ for all $p \in [m]$. Here τ is the threshold that is set to be $\Omega(\log m)$.

Lower Bounds

To describe our lower bounds, recall that the "trivial" additive error that is achieved by outputting an *arbitrary* ranking is $O(m^2)$. In this sense, our algorithms achieve "non-trivial" additive error when $n \geq \tilde{O}(m/\epsilon)$ for pure-DP (Corollary 6) and $n \geq \tilde{O}(\sqrt{m}/\epsilon)$ for approximate-DP (Corollary 7) in the central model and when $n \geq \tilde{O}(m/\epsilon^2)$ in the local model (Corollary 12). In this section we show that this is essentially the best possible, even when the multiplicative approximation ratio is allowed to be large. Specifically, we show the following results for pure-DP and approximate-DP respectively in the *central* model.

Algorithm 3: LDPKwikSort.

```
Parameters: $\epsilon > 0, \tau > 0$, Partition $(\mathcal{P}_1, \ldots, \mathcal{P}_m)$
  of $[n]$, $\ell \leftarrow [m] \times [m] \to [m]$
Input: $\Pi = \{\pi_1, \ldots, \pi_n\}, \mathcal{U} \subseteq [m], c : [m] \to \mathbb{Z}$
if $\mathcal{U} = \emptyset$ then
    return $\emptyset$
$\mathcal{U}_L, \mathcal{U}_R \leftarrow \emptyset$
Pick a pivot $i \in \mathcal{U}$ uniformly at random
for $j \in \mathcal{U} \setminus \{i\}$ do
    for user $k \in \mathcal{P}_{\ell(j,i)}$ do
        $c(\ell(j,i)) \leftarrow c(\ell(j,i)) + 1$
        if $c(\ell(j,i)) > \tau$ then
            Run (local) PTAS with privacy budget of
              $\epsilon/2$                          ▷ See Corollary 9
        $\tilde{w}^k_{ji} \leftarrow \begin{cases} d_\epsilon & \text{w.p. } 1/2 \cdot (1 + w^k_{ji}/d_\epsilon) \\ -d_\epsilon & \text{w.p. } 1/2 \cdot (1 - w^k_{ji}/d_\epsilon) \end{cases}$
    $\tilde{w}_{ji} \leftarrow \frac{m}{n} \cdot \sum_{k \in \mathcal{P}_{\ell(j,i)}} \tilde{w}^k_{ji}$
    if $\tilde{w}_{ji} > 1/2$ then
        Add $j$ to $\mathcal{U}_L$
    else
        Add $j$ to $\mathcal{U}_R$
return LDPKwikSort($\Pi, \mathcal{U}_L, c$), $i$, LDPKwikSort($\Pi$,
  $\mathcal{U}_R, c$)
```

Theorem 13. *For any $\alpha, \epsilon > 0$, there is no ϵ-DP $(\alpha, 0.01m^2)$-approximation algorithm for rank aggregation in the central model for $n = o(m/\epsilon)$.*

Theorem 14. *For any constant $\alpha > 0$ and any $\epsilon \in (0,1]$, there exists $c > 0$ (depending on α) such that there is no $(1, o(1/n))$-DP (α, cm^2)-approximation algorithm for rank aggregation in the central model for $n = o(\sqrt{m}/\epsilon)$.*

We stress that any lower bound in the central model also applies to DP algorithms in the local model. Specifically, the sample complexity in Theorem 13 also matches that in Corollary 12 to within a factor of $O(1/\epsilon)$.

Due to space constraints, we will only describe high-level ideas here. The full proofs are deferred to SM.

Proof Overview: Connection to (1-Way) Marginals. We will describe the intuition behind the proofs of Theorems 13 and 14. At the heart of the proofs, we essentially reduce from the 1-way marginal problem. For $d, t \in \mathbb{N}$, let $x \mapsto \pi^x_{d,t}$ denote the mapping from $\{-1,+1\}^d$ to $\mathbb{S}_{2d+t}$ defined by $\pi^x_{d,t}(\ell) = \ell$ for all $\ell \in \{d+1, \ldots, d+t\}$ and

$$\pi^x_{d,t}(j) := \begin{cases} j & \text{if } x_j = 1 \\ j + d + t & \text{if } x_j = -1, \end{cases}$$

$$\pi^x_{d,t}(j + d + t) := \begin{cases} j + d + t & \text{if } x_j = 1 \\ j & \text{if } x_j = -1, \end{cases}$$

for all $j \in [d]$. In words, $\pi^x_{d,t}$ is an ordering where we start from the identity and switch the positions of j and $j + d + t$ if $x_j = +1$.

Recall that in the *(1-way) marginal problem*, we are given vectors $x^1, \dots, x^n \in \{-1, +1\}^d$ and the goal is to determine $\frac{1}{n} \sum_{i \in [n]} x^i$. The connection between this problem and the rank aggregation problem is that a "good" aggregated rank on $\pi^{x^1}_{d,t}, \dots, \pi^{x^n}_{d,t}$ must "correspond" to a d-dimensional vector whose sign is similar to the 1-way marginal. To formalize this, let us define the "inverse" operation of $x \mapsto \pi^x_{d,t}$, which we denote by $\rho_{d,t} : \mathbb{S}_{2d+t} \to \{-1, +1\}^d$ as follows:

$$\rho_{d,t}(\pi)_j = \begin{cases} -1 \text{ if } \pi(j) < \pi(j+d+t) \\ +1 \text{ otherwise.} \end{cases}$$

The aforementioned observation can now be formalized:

Observation 15. *For any $x^1, \dots, x^n \in \{-1, +1\}^d$ and any $\sigma \in \mathbb{S}_{2d}$, we have*

$$K(\sigma, \{\pi^{x^1}_{d,t}, \dots, \pi^{x^n}_{d,t}\}) \geq t \left(\frac{d}{2} - \frac{1}{2} \left\langle \rho_{d,t}(\sigma), \frac{1}{n} \sum_{i \in [n]} x^i \right\rangle \right).$$

The "inverse" of the above statement is not exactly true, since we have not accounted for the inversions of elements in $\{1, \dots, d+1, d+t+1, \dots, 2d+t\}$. Nonetheless, this only contributes at most $2d^2$ to the number of inversions and we can prove the following:

Observation 16. *For any $x^1, \dots, x^n, y \in \{\pm 1\}^d$, we have*

$$K(\pi^y_{d,t}, \{\pi^{x^1}_{d,t}, \dots, \pi^{x^n}_{d,t}\}) \leq t \left(\frac{d}{2} - \frac{1}{2} \left\langle y, \frac{1}{n} \sum_{i \in [n]} x^i \right\rangle \right) + 2d^2.$$

Ignoring the additive $2d^2$ term, Observations 16 and 15 intuitively tell us that if the 1-way marginal is hard for DP algorithms, then so is rank aggregation.

Pure-DP For pure-DP, it is now relatively simple to apply the packing framework of Hardt and Talwar (2010) for proving a DP lower bound: we can simply pick $x^1 = \dots = x^n$ to be a codeword from an error-correcting code. Observation 15 tells us that this is also a good packing for the Kendall tau metric, which immediately implies Theorem 13.

Approximate-DP Although there are multiple lower bounds for 1-way marginals in the approximate-DP setting (e.g., (Bun, Ullman, and Vadhan 2018; Steinke and Ullman 2015; Dwork et al. 2015; Steinke and Ullman 2017)), it does not immediately give a lower bound for the rank aggregation problem because our observations only allow us to recover the *signs* of the marginals, but not their *values*. Fortunately, it is known that signs are already enough to violate privacy (Dwork et al. 2015; Steinke and Ullman 2017) and thus we can reduce from these results[4]. Another complication comes from the additive $O(d^2)$ term in Observation 16. However, it turns out that we can overcome this by simply picking t to be sufficiently large so that this additive factor is small when compared to the optimum.

[4]Another advantage of (Dwork et al. 2015; Steinke and Ullman 2017) is that their the marginal distributions are flexible; indeed, we need distributions which have large standard deviation (i.e., mean is close to -1 or $+1$) in order to get a large approximation ratio.

Other Related Work

In many disciplines (especially in social choice), rank aggregation appears in many different forms with applications to collaborative filtering and more general social computing (Cohen, Schapire, and Singer 1999; Pennock, Horvitz, and Giles 2000; Dwork et al. 2001). It has been shown previously by Arrow (1963) that no voting rule (on at least three candidates) can satisfy certain criteria at once. To circumvent such impossibility results, we could rely on relaxed criteria such as the Condorcet (Young and Levenglick 1978; Young 1995) condition, where we pick a candidate that beats all others in head-to-head comparisons. The Kemeny ranking (Kemeny and Snell 1962) can be used to obtain a Condorcet winner (if one exists) and also rank candidates in such a way as to minimize disagreements between the rankings from the voters. The Kemeny ranking problem is NP-hard even for four voters (Bartholdi, Tovey, and Trick 1989; Cohen, Schapire, and Singer 1999; Dwork et al. 2001). To overcome this, computationally efficient approximation algorithms have been devised (de Borda 1781; Diaconis and Graham 1977; Conitzer, Davenport, and Kalagnanam 2006; Kenyon-Mathieu and Schudy 2007; Ailon, Charikar, and Newman 2008). Our results rely on such algorithms.

In correlation clustering (Bansal, Blum, and Chawla 2004), we are given a graph whose edges are labeled green or red. The goal is to find a clustering that minimizes the number of pairwise disagreements with the input graph, i.e., the number of intra-cluster red edges and inter-cluster green edges. Correlation clustering is closely related to ranking problems (see e.g., (Ailon, Charikar, and Newman 2008)). Bun, Eliáš, and Kulkarni (2021) recently tackled correlation clustering under DP constraints, although their notion of neighboring datsets—roughly corresponding to changing an edge—is very different than ours.

Conclusions & Future Work

In this work, we have provided several DP algorithms and lower bounds for the rank aggregation problem in the central and local models. Since each of our algorithms achieves either the near-optimal approximation ratio or the near-optimal additive error (but not both), an immediate open question here is if one can get the best of both worlds.

Furthermore, recall that our local DP algorithm in Corollary 12 requires interactivity, which seems inherent for any QuickSort-style algorithms. It is interesting if one can achieve similar guarantees with a *non-interactive* local DP algorithm. We point out that separations between interactive and non-interactive local models are known (see e.g., (Dagan and Feldman 2020) and references therein); thus, it is possible that such a separation exists for rank aggregation. Lastly, it is interesting to see if we can extend our results to other related problems—such as consensus and correlation clustering—that rely on the (weighted) minimum FAST problem for their non-private approximation algorithms.

References

Ailon, N.; Charikar, M.; and Newman, A. 2008. Aggregating inconsistent information: Ranking and clustering. *J. ACM*, 55(5): 23:1–23:27.

Arrow, K. 1963. *Social Choice and Individual Values*. Wiley.

Bansal, N.; Blum, A.; and Chawla, S. 2004. Correlation Clustering. *Mach. Learn.*, 56(1-3): 89–113.

Bartholdi, J.; Tovey, C. A.; and Trick, M. A. 1989. Voting schemes for which it can be difficult to tell who won the election. *Social Choice and Welfare*, 6(2): 157–165.

Bassily, R. 2019. Linear Queries Estimation with Local Differential Privacy. In *AISTATS*, 721–729.

Bun, M.; Eliás, M.; and Kulkarni, J. 2021. Differentially Private Correlation Clustering. In *ICML*, volume 139, 1136–1146. PMLR.

Bun, M.; Nelson, J.; and Stemmer, U. 2019. Heavy Hitters and the Structure of Local Privacy. *ACM Trans. Algorithms*, 15(4): 51:1–51:40.

Bun, M.; Ullman, J. R.; and Vadhan, S. P. 2018. Fingerprinting Codes and the Price of Approximate Differential Privacy. *SIAM J. Comput.*, 47(5): 1888–1938.

Cohen, W. W.; Schapire, R. E.; and Singer, Y. 1999. Learning to Order Things. *J. Artif. Intell. Res.*, 10: 243–270.

Conitzer, V.; Davenport, A. J.; and Kalagnanam, J. 2006. Improved Bounds for Computing Kemeny Rankings. In *AAAI*, 620–626.

Dagan, Y.; and Feldman, V. 2020. Interaction is necessary for distributed learning with privacy or communication constraints. In *STOC*, 450–462. ACM.

de Borda, J.-C. 1781. Mémoire sur les élections au scrutin. *Histoire de l'Académie Royale des Sciences*.

Diaconis, P.; and Graham, R. L. 1977. Spearman's Footrule as a Measure of Disarray. *JRS Series B (Methodological)*, 39(2): 262–268.

Duchi, J. C.; Jordan, M. I.; and Wainwright, M. J. 2014. Local Privacy, Data Processing Inequalities, and Statistical Minimax Rates. arXiv:1302.3203.

Duchi, J. C.; and Rogers, R. 2019. COLT. 1161–1191.

Dwork, C.; Kenthapadi, K.; McSherry, F.; Mironov, I.; and Naor, M. 2006a. Our Data, Ourselves: Privacy Via Distributed Noise Generation. In *EUROCRYPT*, 486–503.

Dwork, C.; Kumar, R.; Naor, M.; and Sivakumar, D. 2001. Rank aggregation methods for the Web. In *WWW*, 613–622.

Dwork, C.; McSherry, F.; Nissim, K.; and Smith, A. D. 2006b. Calibrating Noise to Sensitivity in Private Data Analysis. In *TCC*, 265–284.

Dwork, C.; and Roth, A. 2014. The Algorithmic Foundations of Differential Privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3-4): 211–407.

Dwork, C.; Smith, A. D.; Steinke, T.; Ullman, J. R.; and Vadhan, S. P. 2015. Robust Traceability from Trace Amounts. In *FOCS*, 650–669.

Fligner, M. A.; and Verducci, J. S. 1986. Distance Based Ranking Models. *JRS Series B (Methodological)*, 48(3): 359–369.

Hardt, M.; and Talwar, K. 2010. On the geometry of differential privacy. In *STOC*, 705–714.

Hay, M.; Elagina, L.; and Miklau, G. 2017. Differentially Private Rank Aggregation. In *SDM*, 669–677.

Joseph, M.; Mao, J.; Neel, S.; and Roth, A. 2019. The Role of Interactivity in Local Differential Privacy. In *FOCS*, 94–105. IEEE Computer Society.

Kasiviswanathan, S. P.; Lee, H. K.; Nissim, K.; Raskhodnikova, S.; and Smith, A. D. 2011. What Can We Learn Privately? *SIAM J. Comput.*, 40(3): 793–826.

Kemeny, J. G.; and Snell, J. L. 1962. *Mathematical Models in the Social Sciences*. MIT Press.

Kenyon-Mathieu, C.; and Schudy, W. 2007. How to rank with few errors. In *STOC*, 95–103.

Liu, A.; Lu, Y.; Xia, L.; and Zikas, V. 2020. How Private Are Commonly-Used Voting Rules? In *UAI*, 629–638.

Mallows, C. L. 1957. Non-Null Ranking Models. I. *Biometrika*, 44(1/2): 114–130.

Mandhani, B.; and Meila, M. 2009. Tractable Search for Learning Exponential Models of Rankings. In *AISTATS*, 392–399.

Mansouri, B.; Zanibbi, R.; and Oard, D. W. 2021. Learning to Rank for Mathematical Formula Retrieval. In *SIGIR*, 952–961.

McSherry, F.; and Talwar, K. 2007. Mechanism Design via Differential Privacy. In *FOCS*, 94–103.

Meila, M.; Phadnis, K.; Patterson, A.; and Bilmes, J. A. 2007. Consensus ranking under the exponential model. In *UAI*, 285–294.

Pennock, D. M.; Horvitz, E.; and Giles, C. L. 2000. Social Choice Theory and Recommender Systems: Analysis of the Axiomatic Foundations of Collaborative Filtering. In *AAAI*, 729–734.

Schalekamp, F.; and van Zuylen, A. 2009. Rank Aggregation: Together We're Strong. In *ALENEX*, 38–51.

Steinke, T.; and Ullman, J. 2017. Tight Lower Bounds for Differentially Private Selection. In *FOCS*, 552–563.

Steinke, T.; and Ullman, J. R. 2015. Interactive Fingerprinting Codes and the Hardness of Preventing False Discovery. In *COLT*, 1588–1628.

Steinke, T.; and Ullman, J. R. 2016. Between Pure and Approximate Differential Privacy. *J. Priv. Confidentiality*, 7(2).

Warner, S. L. 1965. Randomized Response: A Survey Technique for Eliminating Evasive Answer Bias. *Journal of the American Statistical Association*, 60(309): 63–69.

Wistuba, M.; and Pedapati, T. 2020. Learning to Rank Learning Curves. In *ICML*, 10303–10312.

Yan, Z.; Li, G.; and Liu, J. 2020. Private rank aggregation under local differential privacy. *Int. J. Intell. Syst.*, 35(10): 1492–1519.

Young, H. P.; and Levenglick, A. 1978. A Consistent Extension of Condorcet's Election Principle. *SIAM J. Appl. Math.*, 35(2): 285–300.

Young, P. 1995. Optimal Voting Rules. *J. Econ. Persp.*, 9(1): 51–64.