The Annals of Statistics
2022, Vol. 50, No. 4, 2231–2255
https://doi.org/10.1214/22-AOS2185

LEARNING MIXTURES OF PERMUTATIONS: GROUPS OF PAIRWISE COMPARISONS AND COMBINATORIAL METHOD OF MOMENTS

BY CHENG MAO[1,a] AND YIHONG WU[2,b]

[1]*School of Mathematics, Georgia Institute of Technology,* [a]*cheng.mao@math.gatech.edu*
[2]*Department of Statistics and Data Science, Yale University,* [b]*yihong.wu@yale.edu*

In applications such as rank aggregation, mixture models for permutations are frequently used when the population exhibits heterogeneity. In this work, we study the widely used Mallows mixture model. In the high-dimensional setting, we propose a polynomial-time algorithm that learns a Mallows mixture of permutations on n elements with the optimal sample complexity that is proportional to $\log n$, improving upon previous results that scale polynomially with n. In the high-noise regime, we characterize the optimal dependency of the sample complexity on the noise parameter. Both objectives are accomplished by first studying demixing permutations under a noiseless query model using groups of pairwise comparisons, which can be viewed as moments of the mixing distribution, and then extending these results to the noisy Mallows model by simulating the noiseless oracle.

1. Introduction. *Rank aggregation* is the task that aims to combine different rankings on the same set of alternatives, to obtain a *central ranking* that best represents the population. The problem of rank aggregation has been studied in social choice theory since Jean-Charles de Borda [3] and Marquis de Condorcet [10] in the 18th century. More recently, due to the ubiquity of preference data, rank aggregation has found applications in a variety of areas, including web search, classification and recommender systems [2, 14, 15, 22, 24].

In these practical applications, the population of interest is often *heterogeneous* in the sense that different subpopulations have divided preferences over the alternatives. For example, multiple groups of people may have different preferences for movies or electoral candidates [17, 29]. In such a scenario, rather than seeking a single central ranking, it is preferable to find *a mixture of rankings* to represent the preferences of the population [1, 7, 11, 18, 21, 23, 33, 37].

1.1. *Mallows mixture and related work.* In this work, we adopt a statistical approach to the problem of heterogeneous rank aggregation. Let $\mathcal{S}_n$ denote the set of permutations on $[n] \triangleq \{1, \dots, n\}$. A ranking of n alternatives is described by a permutation $\pi \in \mathcal{S}_n$. We refer to n as the *size* of a permutation. Furthermore, we model the preference of the population by a distribution on the set of permutations $\mathcal{S}_n$. Suppose that N independent permutations are generated from the distribution, each of which represents an observed ranking.

In this paper, we focus on the *Mallows model* $M(\pi, \phi)$ on $\mathcal{S}_n$, with *central permutation* $\pi \in \mathcal{S}_n$ and *noise parameter* $\phi \in (0, 1)$ [27]. In the Mallows model, the probability of generating a permutation $\sigma \in \mathcal{S}_n$ is equal to

$$\frac{1}{Z(\phi)} \phi^{d_{\mathrm{KT}}(\pi,\sigma)},$$

Received September 2020; revised February 2022.
MSC2020 subject classifications. Primary 62F07, 62H30; secondary 68P10.
Key words and phrases. Mixture of permutations, Mallows model, rank aggregation, group of pairwise comparisons, method of moments.

where $Z(\phi)$ is a normalization factor (see (10)) and $d_{\mathsf{KT}}(\pi,\sigma)$ denotes the *Kendall tau* distance between permutations π and σ, defined by

$$d_{\mathsf{KT}}(\pi,\sigma) \triangleq \sum_{i,j\in[n]} \mathbb{1}\{\pi(i)<\pi(j), \sigma(i)>\sigma(j)\}. \tag{1}$$

There have been decades of work studying theoretical properties and efficient learning algorithms for the Mallows model and its generalizations [4–6, 8, 12, 16, 20, 25, 31].

To model a heterogeneous population, we consider the *Mallows mixture*

$$\mathcal{M} \triangleq \sum_{i=1}^{k} w_i M(\pi_i,\phi) \tag{2}$$

with k components, where the ith component has central permutation $\pi_i \in \mathcal{S}_n$, noise parameter $\phi \in (0,1)$, and weight $w_i \geq \gamma$ for some $\gamma > 0$. We assume for simplicity that the noise parameter ϕ is known and common for all components of the mixture. In general, different components may have different, unknown noise parameters $\phi_i \in (0,1)$, which we briefly discuss in Section 5. Let us remark that, the number of components k in a mixture of permutations is typically a small quantity, so we let k be a fixed constant throughout this work. On the other hand, the size n of the permutations is typically large because it represents the number of alternatives.

The Mallows mixture has also received considerable attention in recent years [1, 9, 11, 23, 26, 30]. More specifically, Chierichetti et al. [9] established the identifiability of the Mallows mixture given sufficiently many permutations generated from $\mathcal{M}$ under mild conditions. The first polynomial-time algorithm to learn the Mallows mixture with two components was proposed by Awasthi et al. [1], who particularly showed that the central permutations can be recovered exactly with high probability, when the sample size N exceeds $\mathsf{poly}(n, \frac{1}{\phi(1-\phi)}, \frac{1}{\gamma})$. In the case of the Mallows k-mixture for any fixed constant k, Liu and Moitra [23] introduced a polynomial-time algorithm with sample complexity $\mathsf{poly}(n, \frac{1}{1-\phi}, \frac{1}{\gamma})$ that exactly recovers the central permutations with high probability.

1.2. *Major contributions.* The first main result of this work concerns the sample complexity of learning Mallows mixture when the size of the permutation is large.

THEOREM 1.1 (Informal statement of Corollary 3.5). *There is a polynomial-time algorithm with the following property. Fix any* $0<\delta<0.1$. *Given* $\mathsf{poly}(\frac{1}{1-\phi}, \frac{1}{\gamma}) \cdot \log\frac{n}{\delta}$ *i.i.d. observations from the Mallows k-mixture* (2), *the algorithm exactly recovers the set of central permutations* $\{\pi_1,\dots,\pi_k\}$ *with probability at least* $1-\delta$.

In the above statement, $\mathsf{poly}(\frac{1}{1-\phi}, \frac{1}{\gamma})$ denotes a polynomial in $\frac{1}{1-\phi}$ and $\frac{1}{\gamma}$ whose degree depends on k; see Corollary 3.5 for the explicit expression of this polynomial. Most importantly, this polynomial does not depend on the size n of the permutations, and the sample complexity bound only depends on n logarithmically. This logarithmic dependency on n is a significant improvement over the previous polynomial dependency and is in fact optimal (see the remark after Corollary 3.5).

Complementing Theorem 1.1, the next result makes precise the optimal dependency of the sample complexity on the noise level $\frac{1}{1-\phi}$ when the size of the permutation is fixed.

THEOREM 1.2 (Informal statement of Corollary 4.2). *Consider the equally weighted Mallows k-mixture* (*that is*, (2) *with* $w_1=\cdots=w_k=1/k$) *in the high-noise regime where* ϕ *is close to* 1. *For fixed n and k, the optimal sample complexity for recovering the central permutations is of the order* $(\frac{1}{1-\phi})^{2\lfloor\log_2 k\rfloor+2}$.

1.3. *Logarithmic sample complexity and groups of pairwise comparisons.* To motivate our main methodology based on *pairwise comparisons*, we briefly discuss why the sample complexity for learning the central permutation π in the single-component Mallows model $M(\pi, \phi)$ scales as $\log n$. Mallows showed in his original paper [27] that, for indices $i, j \in [n]$ such that $\pi(i) < \pi(j)$,

$$\mathbb{P}_{\sigma\sim M(\pi,\phi)}\{\sigma(i) < \sigma(j)\} = \frac{\pi(j)-\pi(i)+1}{1-\phi^{\pi(j)-\pi(i)+1}} - \frac{\pi(j)-\pi(i)}{1-\phi^{\pi(j)-\pi(i)}} \geq \frac{1}{2} + \frac{1-\phi}{4}.$$

In other words, the probability that a random permutation σ from $M(\pi, \phi)$ agrees with π on $\{i, j\}$ is at least $1/2$ plus the positive constant $\frac{1-\phi}{4}$. Therefore by Hoeffding's inequality, given N i.i.d. random permutations from $M(\pi, \phi)$, a simple majority vote recovers $\mathbb{1}\{\pi(i) < \pi(j)\}$ correctly with probability at least $1 - e^{-c(1-\phi)^2 N}$ for a constant $c > 0$. As a result, if $N \geq \frac{C \log n}{(1-\phi)^2}$ for a constant $C > 0$, by a union bound, we readily obtain $\mathbb{1}\{\pi(i) < \pi(j)\}$ for all distinct $i, j \in [n]$ with high probability, from which any comparison sort algorithm (such as Quicksort or Heapsort) can be used to recover the central permutation π.

Crucially, the size n of the permutations does not affect the sample complexity of learning each pairwise comparison $\mathbb{1}\{\pi(i) < \pi(j)\}$. Instead, n enters the overall sample complexity only through a union bound of exponentially small probabilities, so that the dependency on n is logarithmic. In fact, this high-level strategy generalizes to the case of learning the Mallows k-mixture. However, the caveat is that pairwise comparisons alone are no longer sufficient for identifying a mixture of permutations; as such, we need to consider *groups of pairwise comparisons*. This framework of demixing permutations using groups of pairwise comparisons is rigorously developed in Section 2 under a noiseless oracle model, which is of independent interest. Later in Section 3, we extend these results to the noisy case by simulating the noiseless oracle using logarithmically many observations drawn from the Mallows mixture model.

1.4. *Method of moments and comparison with Gaussian mixtures.* In the high-noise regime where $\phi \to 1$, the sample complexity $(\frac{1}{1-\phi})^{2\lfloor \log_2 k \rfloor + 2}$ for learning the Mallows k-mixture is achieved by a method of moments of combinatorial flavor, which we now explain informally. For a distribution on the set $\mathcal{S}_n$ of permutations, it is not obvious how to define an appropriate notion of moments. We show in Section 2.2 that, in fact, it is natural to view the set of all groups of m pairwise comparisons as the mth-order moment of the mixing distribution $\sum_{i=1}^k w_i \delta_{\pi_i}$ associated with the Mallows mixture $\mathcal{M} = \sum_{i=1}^k w_i M(\pi_i, \phi)$. Moreover, the exponent of $\frac{1}{1-\phi}$ in the optimal sample complexity is precisely determined by the maximum number of moments two distinct mixtures can match. Namely, there exist two distinct k-mixtures with the same first $\lfloor \log_2 k \rfloor$ moments, but any k-mixture can be identified from the first $\lfloor \log_2 k \rfloor + 1$ moments, giving rise to the optimal sample complexity $(\frac{1}{1-\phi})^{2\lfloor \log_2 k \rfloor + 2}$. From this perspective, learning a Mallows mixture from groups of pairwise comparisons can be viewed as a combinatorial method of moments.

Furthermore, we draw a comparison between the Mallows mixture and the better-studied Gaussian mixture [34]. Specifically, consider the k-component n-dimensional Gaussian location mixture $\sum_{i=1}^k w_i \mathcal{N}(\mu_i, I_n)$, where n and k are both fixed constants. It is known [13, 19, 32, 35] that the sharp sample complexity of learning the mixing distribution $\sum_{i=1}^k w_i \delta_{\mu_i}$ up to an error ε in the Wasserstein W_1-distance is of the order ε^{4k-2}, which can be achieved by a version of the method of moments. In contrast to the exponential growth of the sample complexity in the Gaussian mixture model, for Mallows mixtures the optimal sample complexity scales *polynomially* with the number of components, thanks to the discrete nature of permutations.

1.5. *Relation to Zagier's work on group determinant.* It is worth mentioning that the identifiability of the Mallows mixture model is related to a result of Zagier in mathematical physics [36]. In [36], Theorem 2, Zagier computed the determinant of the matrix $A(\phi) \in \mathbb{R}^{n! \times n!}$ indexed by permutations in $\mathcal{S}_n$ and defined by

$$A(\phi)_{\pi,\sigma} \triangleq \phi^{d_{\mathsf{KT}}(\pi,\sigma)}. \tag{3}$$

This is an instance of the *group determinant* associated with the symmetric group $\mathcal{S}_n$; see Section A.8 for details. In particular, Zagier showed that

$$\det\big(A(\phi)\big) \neq 0 \quad \text{for all } \phi \in (0,1). \tag{4}$$

Note that, up to the normalization factor $1/Z(\phi)$, the row of $A(\phi)$ indexed by π is precisely the probability mass function (PMF) of the Mallows model $M(\pi,\phi)$. Moreover, the rows of $A(\phi)$ are linearly independent since the determinant of $A(\phi)$ is nonzero. Therefore, if two Mallows mixtures $\sum_{i=1}^{k} w_i M(\pi_i,\phi)$ and $\sum_{i=1}^{k} w_i' M(\pi_i',\phi)$ are identical, then the two sets of central permutations must coincide and so do the corresponding weights. Therefore, Zagier's result implies the *identifiability* of the Mallows mixture.

However, in the finite-sample setting, as noted by Liu and Moitra [23], the direct quantitative implication of [36] is very weak, as it only guarantees a sample complexity that is exponential in n for learning the mixture. While the sample complexity is reduced to a polynomial in n in [23], in this paper we take a step further to achieve the optimal logarithmic sample complexity. As in [23], we also use Zagier's result as a building block; see Lemma A.2.

Furthermore, we remark that another group determinant (defined in (A.22) which is a variant of the one studied in [36], Section 3) appears naturally in one of our technical proofs. See Section A.8 for details.

1.6. *Organization.* The remainder of the paper is organized as follows. In Section 2, we define groups of pairwise comparisons and interpret them as moments of a mixture. Moreover, we study learning a mixture of permutations from groups of pairwise comparisons under a generic, noiseless model. Extending these results to the noisy case, in Section 3, we consider the Mallows mixture and present an algorithm that achieves the sample complexity logarithmic in the size of the permutations. In Section 4, we study the sample complexity of learning the Mallows mixture in the high-noise regime. Section 5 discusses potential extensions of our results and proof techniques. The proofs are presented in Section 6 and in the Supplementary Material [28].

1.7. *Notation.* Let $[n] \triangleq \{1,\dots,n\}$ and $\mathbb{N} \triangleq \{1,2,\dots\}$. Let $\mathsf{TV}(\mathcal{P},\mathcal{Q})$ stand for the total variation distance between two probability distributions $\mathcal{P}$ and $\mathcal{Q}$.

Let $\mathcal{S}_n$ denote the set of permutations on $[n]$. When presenting concrete instances of permutations, we use the notation

$$\pi = \big(\pi^{-1}(1), \pi^{-1}(2), \dots, \pi^{-1}(n)\big),$$

so that when π is understood as a ranking, $\pi^{-1}(i)$ is the element that is ranked in the ith place by π. For example, $(3,2,4,1)$ denotes the permutation π with $\pi(3)=1$, $\pi(2)=2$, $\pi(4)=3$ and $\pi(1)=4$.

For a permutation $\pi \in \mathcal{S}_n$ and a subset $J \subset [n]$, we use the notation $\pi(J) \triangleq \{\pi(j) : j \in J\}$. We let $\pi|_J$ denote the restriction of π on J, which is an injection from J to $[n]$. Moreover, let $\pi\|_J$ denote the bijection from J to $[|J|]$ induced by $\pi|_J$. That is, if σ is the increasing bijection from $\pi(J)$ to $[|J|]$, then $\pi\|_J = \sigma \circ \pi|_J$.

For example, consider $\pi = (3,2,4,6,1,5)$ and $J = \{1,4,5\}$. Then $\pi|_J(1) = 5$, $\pi|_J(4) = 3$ and $\pi|_J(5) = 6$, while $\pi\|_J(1) = 2$, $\pi\|_J(4) = 1$ and $\pi\|_J(5) = 3$. We also write $\pi\|_J =$

$(4, 1, 5)$, which can be easily obtained from the notation $\pi = (3, 2, 4, 6, 1, 5)$ by retaining only the elements of J.

Note that $\pi\|_J$ can be viewed as a total order on J. Moreover, by identifying the elements of J with $1, \ldots, |J|$ in the ascending order, we can identify bijections from J to $[|J|]$ with permutations in $\mathcal{S}_{|J|}$. Hence, $\pi\|_J$ can be equivalently understood as a permutation in $\mathcal{S}_{|J|}$. We may therefore refer to $\pi\|_J$ informally as a permutation or a relative order on J. Moreover, for nested sets $J \subset J' \subset [n]$, we clearly have $(\pi\|_{J'})\|_J = \pi\|_J$.

2. Demixing permutations with groups of pairwise comparisons. In this section, we set up a general approach to learning mixtures of permutations: We first formalize the notions of groups of pairwise comparisons and comparison moments, and then characterize when a mixture of permutations can be learned from groups of pairwise comparisons in a generic noiseless model.

2.1. *Groups of pairwise comparisons.* Let M denote a distribution on $\mathcal{S}_n$. In this work, we are interested in the situation where M is a certain model for a mixture of permutations. To motivate the method of learning the mixture M from groups of pairwise comparisons, let us first consider some simple examples:

- If M is the Dirac delta measure δ_π for a fixed permutation $\pi \in \mathcal{S}_n$, we are tasked with identifying the single permutation π. Let us consider the *pairwise comparison oracle*: Given any pair of distinct indices $(i, j) \in [n]^2$, the oracle returns whether i is placed before j by π, that is, $\mathbb{1}\{\pi(i) < \pi(j)\}$. Based on this oracle, any comparison sorting algorithm (e.g., quicksort) can be deployed to identify π.
- For a general distribution M, the pairwise comparison oracle naturally extends to the following: Given any pair of distinct indices $(i, j) \in [n]^2$, the oracle returns the *distribution* of $\mathbb{1}\{\pi(i) < \pi(j)\}$ where $\pi \sim \mathsf{M}$.

  However, as pointed out by Awasthi et al. [1], even for the noiseless 2-mixture $\mathsf{M} = \frac{1}{2}(\delta_{\pi_1} + \delta_{\pi_2})$, the pairwise comparison oracle is not sufficient for identifying M. For example, if the permutations π_1 and π_2 are reversals of each other, then for any pair of distinct indices (i, j), the output of the pairwise comparison oracle is always Bernoulli$(\frac{1}{2})$, which is uninformative.
- Now that comparing one pair of indices at a time does not guarantee identifiability, how about comparing two pairs simultaneously? This motivates the following oracle that returns a group of two pairwise comparisons: Given pairs of distinct indices $(i_1, j_1), (i_2, j_2) \in [n]^2$, the oracle returns the distribution of

$$\begin{pmatrix} \mathbb{1}\{\pi(i_1) < \pi(j_1)\} \\ \mathbb{1}\{\pi(i_2) < \pi(j_2)\} \end{pmatrix} \quad \text{where } \pi \sim \mathsf{M}.$$

  To illustrate why groups of two pairwise comparisons are sufficient for identifying a mixture of two permutations, we consider a mixture $\mathsf{M} = \frac{1}{2}(\delta_{\pi_1} + \delta_{\pi_2})$ where the two permutations satisfy $\pi_1(1) < \pi_1(2)$ and $\pi_2(1) > \pi_2(2)$. When we make a query on the group of pairs $(1, 2)$, (i, j) for any distinct indices $i, j \in [4]$, the oracle returns the mixture of two delta measures at

$$\begin{pmatrix} 1 \\ \mathbb{1}\{\pi_1(i) < \pi_1(j)\} \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} 0 \\ \mathbb{1}\{\pi_2(i) < \pi_2(j)\} \end{pmatrix}$$

  respectively. Therefore, using the pair $(1, 2)$ as a signature for the two permutations in the mixture, we can demix the pairwise comparisons $\mathbb{1}\{\pi_1(i) < \pi_1(j)\}$ and $\mathbb{1}\{\pi_2(i) < \pi_2(j)\}$ for every pair of indices (i, j), from which π_1 and π_2 can be recovered.

  It turns out that this argument can be made rigorous and extended to the case of a general k-mixtures (Theorem 2.6).

Given these considerations, we are ready to formally define a group of pairwise comparisons.

DEFINITION 2.1 (Group of m pairwise comparisons, the strong oracle). Consider a distribution M on $\mathcal{S}_n$ and a random permutation $\pi \sim \mathsf{M}$. For $m \in \mathbb{N}$, let $\mathcal{I}$ be the tuple of m pairs of distinct indices $(i_1, j_1), \ldots, (i_m, j_m) \in [n]^2$. Upon a query on $\mathcal{I}$, the strong oracle of group of m pairwise comparisons returns the distribution of the random vector $\chi(\pi, \mathcal{I})$ in $\{0,1\}^m$, whose rth coordinate is defined by

$$\chi(\pi, \mathcal{I})_r \triangleq \mathbb{1}\{\pi(i_r) < \pi(j_r)\} \quad \text{for } r \in [m]. \tag{5}$$

We emphasize that in the tuple $\mathcal{I}$ of pairs of distinct indices, i_r and j_r are required to be distinct for each $r \in [m]$, but we allow the scenarios where $i_1 = i_2$ or $i_1 = j_2$, for example. Moreover, throughout this work, the queries we consider are *adaptive*: Our algorithms make queries to the oracle in a sequential fashion, where a given query is allowed to depend on the outcomes of previous ones.

In addition, we introduce a weaker oracle of group of pairwise comparisons. This definition is motivated by interpreting a "mixture" as a set of permutations in $\mathcal{S}_n$, rather than a distribution.

DEFINITION 2.2 (Group of m pairwise comparisons, the weak oracle). Consider a set $\{\pi_1, \ldots, \pi_k\}$ of k permutations in $\mathcal{S}_n$. For $m \in \mathbb{N}$, let $\mathcal{I}$ be a tuple of m pairs of distinct indices in $[n]$. Upon a query on $\mathcal{I}$, the weak oracle of group of m pairwise comparisons returns the set of binary vectors $\{\chi(\pi_i, \mathcal{I}) : i \in [k]\}$, where $\chi(\pi_i, \mathcal{I})$ is defined by (5).

If M is a distribution on $\mathcal{S}_n$ supported on $\{\pi_1, \ldots, \pi_k\}$, then the set $\{\chi(\pi_i, \mathcal{I}) : i \in [k]\}$ returned by Definition 2.2 is simply the support of the random vector $\chi(\pi, \mathcal{I})$ returned by Definition 2.1. In this sense, the oracle in Definition 2.2 is weaker. If $|\operatorname{supp}(\mathsf{M})| = k$, then the strong and the weak oracle are equivalent; otherwise the weak oracle is strictly less informative. In the special case of $k = 2$, they are always equivalent. We emphasize that the weak oracle only returns $\{\chi(\pi_i, \mathcal{I}) : i \in [k]\}$ as a collection of (possibly less than k) distinct, unlabeled elements—it does not specify what each $\chi(\pi_i, \mathcal{I})$ is. This weaker notion will be useful later when we study noisy mixtures of permutations.

Besides groups of pairwise comparisons, it is also natural to consider ℓ-wise comparisons, whose strong and weak versions are defined as follows. Recall the notation $\pi\|_J$ for relative order as defined Section 1.7.

DEFINITION 2.3 (ℓ-wise comparison, the strong oracle). Consider a distribution M on $\mathcal{S}_n$ and a random permutation $\pi \sim \mathsf{M}$. For $\ell \in \mathbb{N}$, let J be a subset of $[n]$ of cardinality $|J| = \ell$. Upon a query on J, the strong oracle of ℓ-wise comparison returns the distribution of the relative order $\pi\|_J$.

DEFINITION 2.4 (ℓ-wise comparison, the weak oracle). Consider a set $\{\pi_1, \ldots, \pi_k\}$ of k permutations in $\mathcal{S}_n$. For $\ell \in \mathbb{N}$, let J be a subset of $[n]$ of cardinality $|J| = \ell$. Upon a query on J, the weak oracle of ℓ-wise comparison returns the set of relative orders $\{\pi_i\|_J : i \in [k]\}$.

For $\ell = 2$, the oracle of ℓ-wise comparison simply reduces to the pairwise comparison oracle. Moreover, for $\ell = 2m$, the (strong or weak) oracle of ℓ-wise comparison is stronger than the corresponding oracle of group of m pairwise comparisons. This is because for any tuple $\mathcal{I}$ of m pairs of indices in $[n]$, we can choose $J \subset [n]$ with $|J| = \ell = 2m$ that contains all indices appearing in $\mathcal{I}$. Then, for any permutation π_i, we can obtain the binary vector $\chi(\pi_i, \mathcal{I})$ from the relative order $\pi_i\|_J$.

2.2. *Comparison moments.* We now interpret groups of pairwise comparisons in Definition 2.1 as moments of the random permutation π. Toward this end, we adopt the following notation throughout this paper. For any (random) permutation π in $\mathcal{S}_n$ and a pair of distinct indices $(i, j) \in [n]^2$, we define

$$X^{\pi}_{i,j} \triangleq \mathbb{1}\{\pi(i) < \pi(j)\}. \tag{6}$$

In this work, we frequently identify the permutation π with the array $X^{\pi} = \{X^{\pi}_{i,j}\}_{i \neq j}$. There is certainly redundancy in X^{π} as we lift $\pi \in \mathcal{S}_n$ to $X^{\pi} \in \{0,1\}^{n^2-n}$. For example, $X^{\pi}_{i,j} + X^{\pi}_{j,i} = 1$, and if $X^{\pi}_{i,j} = 1$ and $X^{\pi}_{j,k} = 1$, then we must have $X^{\pi}_{i,k} = 1$.

In Definition 2.1, consider the oracle that returns the distribution of $\chi(\pi, \mathcal{I})$ in the form of its PMF:

$$f_{\chi(\pi,\mathcal{I})}(v) \triangleq \mathbb{P}\{\chi(\pi, \mathcal{I}) = v\} \quad \text{for each } v \in \{0,1\}^m.$$

For example, at the all-ones vector $\mathbf{1}_m \in \{0,1\}^m$,

$$f_{\chi(\pi,\mathcal{I})}(\mathbf{1}_m) = \mathbb{E}[\mathbb{1}\{\chi(\pi, \mathcal{I}) = \mathbf{1}_m\}] = \mathbb{E}\left[\prod_{r=1}^{m} \mathbb{1}\{\pi(i_r) < \pi(j_r)\}\right] = \mathbb{E}\left[\prod_{r=1}^{m} X^{\pi}_{i_r,j_r}\right],$$

which is an mth moment of X^{π}. This motivates the following definition.

DEFINITION 2.5 (Comparison moment). Consider a distribution M on $\mathcal{S}_n$. For a random permutation $\pi \sim \mathsf{M}$, let X^{π} be defined by (6). For $m \in \mathbb{N}$, let $\mathcal{I}$ denote the tuple of m pairs of distinct indices $(i_1, j_1), \ldots, (i_m, j_m) \in [n]^2$. The comparison moment of π with index $\mathcal{I}$ is the vector $\mathfrak{m}(\pi, \mathcal{I}) \in \mathbb{R}^{2^m}$, defined by

$$\mathfrak{m}(\pi, \mathcal{I})_v \triangleq \mathbb{E}\left[\prod_{r=1}^{m} (X^{\pi}_{i_r,j_r})^{v_r} (1 - X^{\pi}_{i_r,j_r})^{1-v_r}\right] \quad \text{for } v \in \{0,1\}^m. \tag{7}$$

Note that the comparison moment defined above is of order at most m in the usual sense, as

$$(X^{\pi}_{i_r,j_r})^{v_r} (1 - X^{\pi}_{i_r,j_r})^{1-v_r} = \begin{cases} X^{\pi}_{i_r,j_r} & \text{if } v_r = 1, \\ X^{\pi}_{j_r,i_r} & \text{if } v_r = 0. \end{cases} \tag{8}$$

Moreover, by (5), (6) and (7), we see that the PMF of the random vector $\chi(\pi, \mathcal{I})$ is precisely the comparison moment $\mathfrak{m}(\pi, \mathcal{I})$ as

$$\begin{aligned} f_{\chi(\pi,I)}(v) &= \mathbb{E}[\mathbb{1}\{\chi(\pi, \mathcal{I}) = v\}] = \mathbb{E}\left[\prod_{r=1}^{m} \mathbb{1}\{X^{\pi}_{i_r,j_r} = v_r\}\right] \\ &= \mathbb{E}\left[\prod_{r=1}^{m} (X^{\pi}_{i_r,j_r})^{v_r} (1 - X^{\pi}_{i_r,j_r})^{1-v_r}\right] = \mathfrak{m}(\pi, \mathcal{I})_v. \end{aligned}$$

As a result, the group of pairwise comparisons on $\mathcal{I}$ can be equivalently defined as the oracle that returns the comparison moment $\mathfrak{m}(\pi, \mathcal{I})$. Learning a mixture of permutations from groups of pairwise comparisons can therefore be viewed a combinatorial method of moments.

2.3. *Efficient learning in a generic model.* With the above definitions formulated, we are ready to study demixing permutations with groups of pairwise comparisons or ℓ-wise comparisons. In this section, we consider the following generic noiseless model for a mixture of k permutations:

$$\mathsf{M} \triangleq \sum_{i=1}^{k} w_i \delta_{\pi_i},$$

where $\pi_1, \ldots, \pi_k$ are permutations in $\mathcal{S}_n$ and $w_1, \ldots, w_k$ are positive weights that sum to one.

It is clear that the more pairs we compare in a group, the more information we obtain. In other words, the larger m is, the stronger the oracle in Definition 2.1 becomes. Similarly, the larger ℓ is, the stronger the oracle in Definition 2.3 becomes. Is there a polynomial-time algorithm that learns the k-mixture M from a polynomial number of groups of m pairwise comparisons for any large n, where m only depends on k but not on n? Furthermore, for a fixed k, what is the weakest oracle we can assume, that is, what is the smallest m, so that such an algorithm exists? The analogous questions can also be asked for the oracle of ℓ-wise comparison. As the main result of this section, the following theorem answers these questions.

THEOREM 2.6. *Let k be a positive integer, and define*

$$m_k^* \triangleq \lfloor \log_2 k \rfloor + 1. \tag{9}$$

(a) *For any mixture $\mathsf{M} = \sum_{i=1}^k w_i \delta_{\pi_i}$ of permutations in $\mathcal{S}_n$, there is a polynomial-time algorithm that recovers M from groups of m_k^* pairwise comparisons, with at most $1 + \frac{k}{2}(n-2)(n+1)$ adaptive queries to the strong oracle in Definition 2.1.*

(b) *Conversely, for $n \geq 2m_k^*$ and $\ell \leq 2m_k^* - 1$, there exist distinct mixtures $\mathsf{M} = \frac{1}{k}\sum_{i=1}^k \delta_{\pi_i}$ and $\mathsf{M}' = \frac{1}{k}\sum_{i=1}^k \delta_{\pi_i'}$ of permutations in $\mathcal{S}_n$, which cannot be distinguished even if all $\binom{n}{\ell}$ ℓ-wise comparisons are queried from the strong oracle in Definition 2.3.*

As we have noted, if $\ell \geq 2m$, then the oracle of ℓ-wise comparison is stronger than the oracle of group of m pairwise comparisons. Therefore, the above theorem implies: (1) The oracle of group of m pairwise comparisons is sufficient for identifying the k mixture if and only if $m \geq m_k^*$; (2) The oracle of ℓ-wise comparison is sufficient for identifying the k mixture if and only if $\ell \geq 2m_k^*$.

In addition to the above theorem which studies the permutation demixing problem assuming the strong oracles, we also have the following result that assumes the weak oracle given by Definition 2.2. Recall that here we view the mixture as a set of permutations rather than a distribution.

THEOREM 2.7. *There is an $O(k^4 n^2)$-time algorithm (see Algorithm 6 in Section 6.3) that learns a set $\{\pi_1, \ldots, \pi_k\}$ of permutations in $\mathcal{S}_n$ from groups of $k+1$ pairwise comparisons, with at most $1 + \frac{k}{2}(n-2)(n-1)$ adaptive queries to the weak oracle in Definition 2.2.*

Unlike Theorem 2.6 where the smallest number m of pairs compared in a query is precisely m_k^*, Theorem 2.7 only shows that m is at most $k+1$ and we do not have a matching lower bound. Nevertheless, the crucial observation is that m again only depends on k, the number of components, but not on n, the size of the permutations.

The algorithms for Theorems 2.6(a) and 2.7 are similar in nature and are both generalizations of Insertion Sort. The latter, called Insertion Demixing, is detailed in Algorithm 6. Furthermore, note that the query complexity for both algorithms is $O(kn^2)$. This is not optimal in general: When $k = 1$ (single component), the problem reduces to comparison sort and the optimal query complexity is $O(n \log n)$, which is achieved by Heapsort, for example. However, Insertion Sort has query complexity $O(n^2)$, and because our algorithms are generalizations of Insertion Sort to the mixture setting, the query complexity with respect to n cannot be improved. It is an interesting open problem to determine the optimal query complexity for the mixture models.

In addition to interest in their own right, the above results have laid the foundation for studying the Mallows mixture in the next two sections. On the one hand, Theorem 2.7 provides a "meta-algorithm" for learning the central permutations, so it suffices to simulate the weak oracle using sample from the Mallows mixture, which we do in Section 3. On the other hand, Theorem 2.6 sheds light on the fundamental limit of learning mixtures of permutations, which we further explore in Section 4 for the Mallows mixture in the high-noise regime.

3. Mallows mixture in high-dimensional regime. Moving from the noiseless to the noisy case, we now turn to the popular Mallows mixture model. Denote the Kendall tau distance between two permutations $\pi, \sigma \in \mathcal{S}_n$ by $d_{\mathsf{KT}}(\pi, \sigma)$ as defined in (1). For a central permutation $\pi \in \mathcal{S}_n$ and a noise parameter $\phi \in (0, 1)$, the Mallows model denoted by $M(\pi, \phi)$ is the distribution on $\mathcal{S}_n$ with PMF

$$f_{M(\pi,\phi)}(\sigma) = \frac{\phi^{d_{\mathsf{KT}}(\sigma,\pi)}}{Z(\phi)} \quad \text{for } \sigma \in \mathcal{S}_n \text{ where } Z(\phi) \triangleq \sum_{\sigma \in \mathcal{S}_n} \phi^{d_{\mathsf{KT}}(\sigma,\mathsf{id})}. \tag{10}$$

Note that ϕ determines the noise level of the Mallows model. As $\phi \to 0$, $M(\pi, \phi)$ converges to the noiseless model, a delta measure at π. On the other hand, as $\phi \to 1$, $M(\pi, \phi)$ converges to the noisiest model, the uniform distribution on $\mathcal{S}_n$. In fact, it is also common [4, 20, 31] to parametrize the noise level by $\beta = 1/\log(1/\phi)$ so that $\phi = e^{-1/\beta}$. Particularly, we have $\beta \approx \frac{1}{1-\phi} \to \infty$ as $\phi \to 1$.

In this work, we consider a mixture $\mathcal{M}$ of k Mallows models $M(\pi_1, \phi), \ldots, M(\pi_k, \phi)$ with a common noise parameter $\phi \in (0, 1)$ and respective weights $w_1, \ldots, w_k > 0$ such that $\sum_{i=1}^k w_i = 1$. In other words, $\mathcal{M}$ is the distribution on $\mathcal{S}_n$ with PMF

$$f_{\mathcal{M}}(\sigma) = \sum_{i=1}^k w_i \frac{\phi^{d_{\mathsf{KT}}(\sigma,\pi_i)}}{Z(\phi)} \quad \text{for } \sigma \in \mathcal{S}_n.$$

We also write $M(\pi_i) \equiv M(\pi_i, \phi)$ and

$$\mathcal{M} = \sum_{i=1}^k w_i M(\pi_i)$$

for brevity. Note that if $\phi = 0$, then $\mathcal{M}$ reduces to the noiseless model $\sum_{i=1}^k w_i \delta_{\pi_i}$ considered in Section 2.

Furthermore, suppose that we are given N i.i.d. observations $\sigma_1, \ldots, \sigma_N$ from the mixture $\mathcal{M}$. Let

$$\mathcal{M}_N \triangleq \frac{1}{N} \sum_{i=1}^N \delta_{\sigma_i}$$

denote the empirical distribution with PMF

$$f_{\mathcal{M}_N}(\sigma) = \frac{1}{N} \sum_{i=1}^N \mathbb{1}\{\sigma_i = \sigma\} \quad \text{for } \sigma \in \mathcal{S}_n.$$

Assuming that the number of components k and the noise parameter ϕ are known, we aim to exactly recover the set of central permutations $\{\pi_1, \ldots, \pi_k\}$ in the mixture. We assume the knowledge of ϕ for technical convenience. In principle, this assumption can be removed, which we discuss in Section 5.

In this section, we consider the "high-dimensional" setting where the size n of the permutations is large, and establish the logarithmic dependency of the sample complexity on n. As

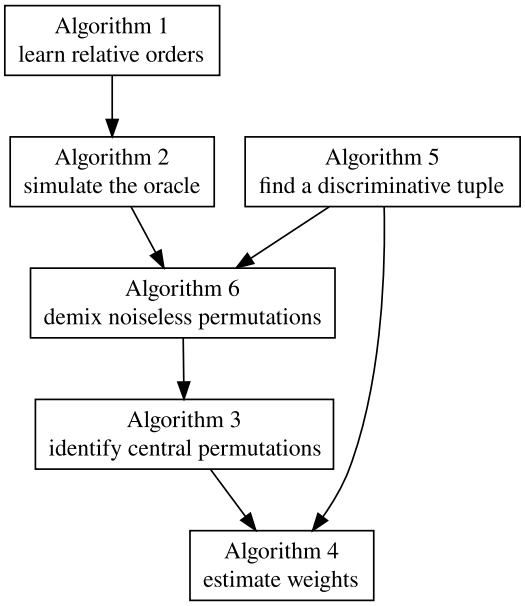

FIG. 1. *Dependency graph of the algorithms.*

hinted earlier, our strategy is to use Algorithm 6 (from Theorem 2.7 for the noiseless case) as a "meta-algorithm" to recover the central permutations of the Mallows mixture. To this end, we need to simulate the weak oracle in Definition 2.2 using noisy observations from the Mallows mixture (which is done by Algorithm 2 below). Furthermore, recall that the weak oracle of ℓ-wise comparison in Definition 2.4 is stronger than the weak oracle of group of m pairwise comparisons in Definition 2.2, provided that $\ell \geq 2m$. Therefore, a main goal of this section is to introduce a subroutine (Algorithm 1) which simulates the weak oracle in Definition 2.4 using logarithmically many observations from the Mallows mixture.

Figure 1 illustrates the dependency among various algorithms in this paper. Specifically, Algorithm 1 learns a set of relative orders on a small set of indices given noisy samples from the Mallows mixture. Algorithm 2 then uses it to simulate the key oracle of groups of pairwise comparisons. This oracle is repeatedly called by Algorithm 6, a recursion, which is the demixing algorithm for the noiseless case. Algorithm 3 is the main algorithm that learns the central permutations for the Mallows mixture based on noisy observations. Given these exactly recovered central permutations, Algorithm 4 then estimates their respective weights in the mixture. Algorithm 5 is a simple subroutine that is used in both Algorithms 6 and 4.

3.1. *Marginalization of Mallows mixture.* Given i.i.d. observations $\sigma_1, \ldots, \sigma_N$ from the Mallows mixture $\mathcal{M} = \sum_{i=1}^{k} w_i M(\pi_i)$ and a subset $J \subset [n]$, the goal of Algorithm 1, denoted by `SubOrder`, is to learn the set of relative orders $\pi_1\|_J, \ldots, \pi_k\|_J$. We recall that the relative order $\pi_i\|_J$ is the bijection from J to $[|J|]$ induced by $\pi_i|_J$; we are not aiming at recovering $\pi_i|_J$ itself.

Toward this end, we consider the marginalization of the Mallows mixture, as well as the observations, as follows. For any distribution $\mathcal{M}$ on $\mathcal{S}_n$ and a set of indices $J \subset [n]$, we let $\mathcal{M}|_J$ denote the marginal distribution of $\sigma|_J$ where $\sigma \sim \mathcal{M}$. That is, the PMF of $\mathcal{M}|_J$ is given by

$$f_{\mathcal{M}|_J}(\rho) = \mathbb{P}_{\sigma\sim\mathcal{M}}\{\sigma|_J = \rho\} \tag{11}$$

for every injection $\rho : J \to [n]$. Moreover, given N i.i.d. observations $\sigma_1, \ldots, \sigma_N$ from $\mathcal{M}$, the empirical version of (11) is given by

$$f_{\mathcal{M}_N|_J}(\rho) = \frac{1}{N}\sum_{m=1}^{N} \mathbb{1}\{\sigma_m|_J = \rho\}. \tag{12}$$

Note that although our goal is to learn the relative order $\pi_i\|_J : J \to [|J|]$ for $i \in [k]$, not the actual values of $\pi_i(j)$ for $j \in J$, the marginalization is with respect to the restriction on J only, and does maintain the values of $\sigma(j)$ for $j \in J$. This is crucial to establishing the following identifiability result for marginalized Mallows mixtures.

PROPOSITION 3.1. *Consider the Mallows mixtures* $\mathcal{M} = \sum_{i=1}^{k} w_i M(\pi_i)$ *and* $\mathcal{M}' = \sum_{i=1}^{k} w_i' M(\pi_i')$ *on* $\mathcal{S}_n$ *with a common noise parameter* $\phi \in (0,1)$. *Let us define* $\gamma \triangleq \min_{i\in[k]}(w_i \wedge w_i') > 0$. *Fix a set of indices* $J \subset [n]$ *and let* $\ell \triangleq |J|$. *Suppose that the two sets of central permutations* $\{\pi_1\|_J, \ldots, \pi_k\|_J\}$ *and* $\{\pi_1'\|_J, \ldots, \pi_k'\|_J\}$ *are not equal* (*as sets*). *Then we have*

$$\mathsf{TV}(\mathcal{M}|_J, \mathcal{M}'|_J) \geq \eta(k, \ell, \phi, \gamma),$$

where

$$\eta(k, \ell, \phi, \gamma) \triangleq \left(\frac{\gamma}{6k}\right)^{(3\ell)^{\ell+1}} \left(\frac{1-\phi}{\ell}\right)^{(4\ell)^{\ell}+2k\ell^2}. \tag{13}$$

Crucially, the above lower bound is *dimension-free*, that is, it does not depend on n. This is one of the two key ingredients (the other being the concentration inequality in Proposition 3.3 below) that enable us to achieve a sample complexity that ultimately depends logarithmically on n. The proof of Proposition 3.1 leverages the notion of *block structure* introduced by Liu and Moitra [23]; see Section A.2 for details.

In addition, we observe a useful property of marginalized Mallows models.

LEMMA 3.2. *For any subset* $J \subset [n]$, *if the central permutations* $\pi, \pi' \in \mathcal{S}_n$ *satisfy* $\pi|_J = \pi'|_J$, *then the marginalized Mallows models* $M(\pi,\phi)|_J$ *and* $M(\pi',\phi)|_J$ *coincide for all* $\phi \in (0,1)$.

PROOF. Let $\tau \in \mathcal{S}_n$ be a relabeling of indices $1, \ldots, n$ such that $\pi' = \pi \circ \tau$. Since $\pi|_J = \pi'|_J$, we have $\tau(j) = j$ for every $j \in J$. It follows that $(\sigma\circ\tau)|_J = \sigma|_J$ for any $\sigma \in \mathcal{S}_n$. Moreover, it holds that $d_{\mathsf{KT}}(\sigma\circ\tau, \pi') = d_{\mathsf{KT}}(\sigma\circ\tau, \pi\circ\tau) = d_{\mathsf{KT}}(\sigma,\pi)$ by the right invariance of the Kendall-tau distance. In view of the definition of the Mallows model and marginalization on J, we reach the conclusion. □

Proposition 3.1 and Lemma 3.2 together motivate the subroutine introduced in the sequel.

3.2. *The subroutine.* We are ready to define the subroutine formally. The first step is to define a set of polynomially many candidate models. Let $\mathcal{S}_{n,J}$ denote the set of injections $\rho : J \to [n]$, which has cardinality at most n^{ℓ} where $\ell = |J|$. For each $\rho \in \mathcal{S}_{n,J}$, fix an arbitrary permutation π_ρ in $\mathcal{S}_n$ such that $\pi_\rho|_J = \rho$. Let L be a positive integer to be determined later. For $\phi \in (0,1)$ and $\gamma \in (0, 1/k]$, we define a set of Mallows mixtures by discretizing the weights

$$\begin{aligned}\mathscr{M} &\equiv \mathscr{M}(n,k,\phi,\gamma,J,L)\\ &\triangleq \left\{\sum_{i=1}^{k}\frac{r_i}{L}M(\pi_{\rho_i},\phi) : \rho_i \in \mathcal{S}_{n,J}, r_i \in [L], r_i \geq \gamma L, \sum_{i=1}^{k} r_i = L\right\}.\end{aligned} \tag{14}$$

Algorithm 1 `SubOrder`

Input: $\sigma_1, \dots, \sigma_N \in \mathcal{S}_n$, $k \in \mathbb{N}$, $\phi \in (0,1)$, $\gamma \in (0, 1/k]$, $N' \in \mathbb{N}$, and $J \subset [n]$
Output: a set $\mathfrak{R}$ of relative orders on J

$\ell \leftarrow |J|$
$\eta \leftarrow \eta(k, \ell, \phi, \gamma)$ as defined in (13)
$L \leftarrow \lceil 3k/\eta \rceil$
$\mathscr{M} \leftarrow \mathscr{M}(n, k, \phi, \gamma, J, L)$ as defined in (14)
$\mathcal{M}_N|_J \leftarrow \frac{1}{N}\sum_{m=1}^{N} \delta_{\sigma_m|_J}$ ▷ *compute the marginalized empirical distribution as in* (12)
$\mathfrak{R} \leftarrow \varnothing$
for $\mathcal{M}' = \sum_{i=1}^{k} \frac{r_i}{L} M(\pi_{\rho_i}, \phi) \in \mathscr{M}$ **do**
    generate N' i.i.d. random permutations $\sigma'_1, \dots, \sigma'_{N'}$ from $\mathcal{M}'$
    $\mathcal{M}'_{N'}|_J \leftarrow \frac{1}{N'}\sum_{m=1}^{N'} \delta_{\sigma'_m|_J}$
    if $\mathsf{TV}(\mathcal{M}'_{N'}|_J, \mathcal{M}_N|_J) \le \eta/2$ **then**
        $\mathfrak{R} \leftarrow \{\pi_{\rho_i}\|_J : i \in [k]\}$
    end if
end for
return $\mathfrak{R}$

Note that the weights r_i/L sum to 1 and each weight is at least γ. Since there are at most L choices for each weight and at most $|\mathcal{S}_{n,J}| \le n^\ell$ choices for each ρ_i, we have $|\mathscr{M}| \le L^k n^{k\ell}$.

In view of the total variation lower bound in Proposition 3.1, it is natural to consider the minimum-distance estimator that selects the Mallows mixture model in $\mathscr{M}$ whose marginal is closest in total variation to that of the empirical distribution $\mathcal{M}_N$; however, without an explicit formula for the marginalized distribution $\mathcal{M}'|_J$ for $\mathcal{M}' \in \mathscr{M}$ it is difficult to directly compute the total variation. Fortunately, we can efficiently sample from $\mathcal{M}'|_J$ and thus approximate the marginalized distribution sufficiently well in polynomial time. This motivates Algorithm 1.

Let us remark that sampling from the Mallows model is computationally efficient with the help of the Repeated Insertion Model of Doignon, Pekeč, and Regenwetter [12] (see also Section 2.2.3 of [26]). In short, to sample from $M(\mathsf{id}, \phi)$, it suffices to start from the empty ranking and repeatedly insert index $i = 1, \dots, n$ into the current ranking at position $j \le i$ with probability $\phi^{i-j}/(1 + \phi + \cdots + \phi^{i-1})$. This sampling procedure can be easily done in $O(n^2)$ time. Furthermore, as an anonymous reviewer pointed out, the time complexity of each insertion step can be improved to $O(\log n)$ by considering a stochastic transition rule on a binary tree with the possible rank positions as the leaves, where the transition probabilities on each edge can be computed explicitly so that the probabilities of outputting each leaf agrees with the values specified above. As a result, sampling from the Mallows model can be done in $O(n \log n)$ time.

Consequently, in Algorithm 1, the computation of $\mathcal{M}'_{N'}|_J$ takes $O(N' n \log n)$ time (where N' will be taken to be logarithmic in n). Moreover, since $\mathcal{M}_N|_J$ and $\mathcal{M}'_{N'}|_J$ are distributions with at most N and N' atoms respectively, computing $\mathsf{TV}(\mathcal{M}'_{N'}|_J, \mathcal{M}_N|_J)$ takes time less than $O(NN'n)$. Furthermore, as we have seen, there are at most $L^k n^{k\ell}$ candidate models where $L = \lceil 3k/\eta \rceil$. We conclude that Algorithm 1 runs in $O((\frac{3k}{\eta})^k NN' n^{k\ell+1} \log n)$ time.

To analyze Algorithm 1, we first state a concentration inequality for the marginalized empirical distribution for the Mallows mixture.

PROPOSITION 3.3. *For $J \subset [n]$, let $\mathcal{M}|_J$ and $\mathcal{M}_N|_J$ be the marginalized Mallows mixture and the marginalized empirical distribution defined by* (11) *and* (12)*, respectively. For any $s \in (0,1)$,*

$$\mathbb{P}\{\mathsf{TV}(\mathcal{M}|_J, \mathcal{M}_N|_J) > s\} \le \exp\left(-N\frac{3s}{10}\right) + 2(2kq)^\ell \exp\left(-N\frac{s^2}{(2kq)^{2\ell}}\right), \tag{15}$$

where $\ell \triangleq |J|$ and $q \triangleq 1 + \frac{1}{1-\phi}\log\frac{8\ell}{s(1-\phi)}$.

Similar to the total variation lower bound in Proposition 3.1, the above concentration inequality is also dimension-free (independent of n). This is possible because although $\mathcal{M}|_J$ is a distribution on $\Theta(n^\ell)$ elements, its "effective support size" is independent of n thanks to a basic property of the Mallows model (Lemma A.1). Propositions 3.1 and 3.3 together enable us to establish the following theoretical guarantee for Algorithm 1.

THEOREM 3.4. *Suppose that we are given i.i.d. observations $\sigma_1, \dots, \sigma_N$ from the Mallows mixture $\mathcal{M} = \sum_{i=1}^k w_i M(\pi_i)$ on $\mathcal{S}_n$ with a noise parameter $\phi \in (0,1)$. Fix a set of indices $J \subset [n]$ and let $\ell \triangleq |J|$. Fix a positive constant $\gamma \le \min_{i\in[k]} w_i$ and a probability of error $\delta \in (0, 0.1)$. Let*

$$\zeta(k,\ell,\phi,\gamma) \triangleq e^{(9\ell)^{\ell+1}} \left(\frac{k}{\gamma}\right)^{(6\ell)^{\ell+1}} \left(\frac{\ell}{1-\phi}\right)^{3(4\ell)^\ell + 8k\ell^2}. \tag{16}$$

If the sample size satisfies $N \ge \zeta(k,\ell,\phi,\gamma)\cdot\log\frac{1}{\delta}$ and we choose an integer $N' \ge \zeta(k,\ell,\phi,\gamma)\cdot\log\frac{n}{\delta}$, then Algorithm 1 returns the set of relative orders $\{\pi_i\|_J : i \in [k]\}$ with probability at least $1-\delta$ in $O((\frac{3k}{\eta})^k N N' n^{k\ell+1}\log n)$ time, where $\eta = \eta(k,\ell,\phi,\gamma)$ is defined in (13).

3.3. *Exact recovery of the central permutations.* Consider a set of indices $J \subset [n]$ and a tuple $\mathcal{I}$ of pairs of distinct indices $(i_1, j_1), \dots, (i_m, j_m) \in J^2$. For any permutation $\pi \in \mathcal{S}_n$, we have $\mathbb{1}\{\pi\|_J(i_r) < \pi\|_J(j_r)\} = \mathbb{1}\{\pi(i_r) < \pi(j_r)\}$ for $r \in [m]$ by the definition of the relative order $\pi\|_J$. Since Algorithm 1 returns the set of relative orders $\{\pi_i\|_J : i \in [k]\}$ with high probability, in particular, we can obtain the set of binary vectors $\{\chi(\pi_i, \mathcal{I}) : i \in [k]\}$, where $\chi(\pi_i, \mathcal{I})$ is defined by (5). This step is formulated as Algorithm 2.

Recall that the set $\{\chi(\pi_i, \mathcal{I}) : i \in [k]\}$ is precisely what we assume the weak oracle in Definition 2.2 returns. Therefore, this oracle is available with high probability for the Mallows mixture provided that the sample size N is sufficiently large. Consequently, Algorithm 6 (recall Theorem 2.7) can be used as a meta-algorithm to recover the central permutations $\{\pi_i : i \in [k]\}$ in the Mallows mixture. We formulate this main algorithm as Algorithm 3. Corollary 3.5 then provides theoretical guarantees for Algorithm 3.

Algorithm 2 `SimulateOracle`

Input: $\sigma_1, \dots, \sigma_N \in \mathcal{S}_n$, $k \in \mathbb{N}$, $\phi \in (0,1)$, $\gamma \in (0, 1/k]$, $N' \in \mathbb{N}$, and a tuple $\mathcal{I}$ of pairs of distinct indices $(i_1, j_2), \dots, (i_m, j_m) \in [n]^2$

Output: a set V of vectors in $\{0,1\}^m$

1: $J \leftarrow$ the set of all indices that appear in the tuple $\mathcal{I}$
2: $\mathfrak{R} \leftarrow$ the set of relative orders on J returned by `SubOrder` (Algorithm 1) with inputs $\sigma_1, \dots, \sigma_N, k, \phi, \gamma, N'$, and J
3: $V \leftarrow \{v^\tau : \tau \in \mathfrak{R}\}$, where $v^\tau \in \{0,1\}^m$ is defined by $v^\tau_r \triangleq \mathbb{1}\{\tau(i_r) < \tau(j_r)\}$ for $r \in [m]$
4: **return** V

Algorithm 3 `DemixMallows`

Input: $\sigma_1, \dots, \sigma_N \in \mathcal{S}_n$, $k \in \mathbb{N}$, $\phi \in (0,1)$, $\gamma \in (0, 1/k]$, and $\delta \in (0, 0.1)$
Output: a set $\mathfrak{S}$ of permutations in $\mathcal{S}_n$

1: $N' \leftarrow \zeta(k, 2k+2, \phi, \gamma) \cdot \log \frac{n^{2k+3}}{\delta}$, where ζ is defined in (16)
2: $\mathfrak{S} \leftarrow$ the set of permutations in $\mathcal{S}_n$ returned by `InsertionDemixing` (Algorithm 6) with the oracle given by `SimulateOracle` (Algorithm 2) with inputs $\sigma_1, \dots, \sigma_N$, k, ϕ, γ, and N'
3: **return** $\mathfrak{S}$

COROLLARY 3.5. *Suppose that we are given i.i.d. observations $\sigma_1, \dots, \sigma_N$ from the Mallows mixture $\mathcal{M} = \sum_{i=1}^{k} w_i M(\pi_i)$ on $\mathcal{S}_n$ with a known noise parameter $\phi \in (0,1)$. Fix a positive constant $\gamma \le \min_{i \in [k]} w_i$ and a probability of error $\delta \in (0, 0.1)$. If the sample size satisfies $N \ge \zeta(k, 2k+2, \phi, \gamma) \cdot \log \frac{n^{2k+2}}{\delta}$ where ζ is defined in (16), then with probability at least $1-\delta$, Algorithm 3 successfully returns the set of central permutations $\{\pi_1, \dots, \pi_k\}$ with time complexity $O(Nn^{2(k^2+k+2)}(\frac{ek}{\gamma(1-\phi)})^{(2k)^{2k+9}} \log \frac{1}{\delta})$.*

PROOF. It suffices to show that Algorithm 1 indeed simulates the oracle in Definition 2.2 with high probability, so that Algorithm 6 returns the set of permutations $\{\pi_1, \dots, \pi_k\}$ as guaranteed by Theorem 2.7. More precisely, we need to prove that Algorithm 2 returns the set of binary vectors $\{\chi(\pi_i, \mathcal{I}) : i \in [k]\}$ for every tuple $\mathcal{I}$ of $k+1$ pairs of distinct indices in $[n]$ with probability at least $1-\delta$.

Recall that J is defined to be the set $\{i_1, j_1, \dots, i_m, j_m\}$ if $\mathcal{I}$ consists of the pairs (i_1, j_1), $\dots$, (i_m, j_m). As we have noted at the beginning of this subsection, it then holds that $\mathbb{1}\{\pi\|_J(i_r) < \pi\|_J(j_r)\} = \mathbb{1}\{\pi(i_r) < \pi(j_r)\}$ for any $\pi \in \mathcal{S}_n$ and $r \in [m]$. Therefore, Algorithm 2 returns the set of binary vectors $\{\chi(\pi_i, \mathcal{I}) : i \in [k]\}$ whenever Algorithm 1 returns the set of relative orders $\{\pi_i\|_J : i \in [k]\}$.

Moreover, Algorithm 6 requires the tuple $\mathcal{I}$ to consist of $k+1$ pairs of indices. Hence J has cardinality at most $2k+2$. Since there are less than n^{2k+2} possible subsets J of $[n]$ that have cardinality at most $2k+2$, we can replace the error probability δ in Theorem 3.4 by δn^{-2k-2} and take a union bound to guarantee that Algorithm 1 returns $\{\pi_i\|_J : i \in [k]\}$ for all such J with probability at least $1-\delta$. This then guarantees the success of Algorithm 2 in simulating the oracle. (Note that although Algorithm 6 only makes at most $1 + \frac{k}{2}(n-2)(n-1)$ queries to the oracle of Definition 2.2 (see Theorem 2.7), we still need to take the union bound over all possible subsets of $[n]$ that have cardinality at most $2k+2$ because the queries are made adaptively.)

Finally, for the time complexity, recall that Algorithms 6 runs in $O(k^4 n^2)$ time and requires $O(kn^2)$ queries (Theorem 2.7). For each query, Algorithm 2 simulates the oracle and the bottleneck of time complexity lies in Algorithm 1. We take $\ell \le 2k+2$ in Algorithm 1, giving a time complexity $O((\frac{3k}{\eta})^k N N' n^{2k^2+2k+1} \log n)$ according to Theorem 3.4 where $\eta = \eta(k, 2k+2, \phi, \gamma)$. Therefore, the overall time complexity is $O(Nn^{2(k^2+k+2)}(\frac{ek}{\gamma(1-\phi)})^{(2k)^{2k+9}} \log \frac{1}{\delta})$ by plugging in the definitions of η, N', and ζ and then simplifying the formula. □

Note that the factor N in the time complexity can be easily incorporated into the other factors, because we can just use a logarithmic number of samples in the algorithm and ignore the rest, which will not hurt the theoretical guarantee on recovering the central permutations.

We remark that the logarithmic dependency of the sample complexity N on the size n of the permutations is optimal, even in the case $k = 1$ where we aim to learn a single central

permutation in the Mallows model. More precisely, the proof of Lemma 10 of [5] established the following information-theoretic lower bound: Given N random observations from the Mallows model $M(\pi, 1/2)$ on $\mathcal{S}_n$, if $N \le c \log n$ for a sufficiently small constant $c > 0$, then any algorithm fails to exactly recover the central permutation π with a constant probability.

3.4. *Learning the weights.* Once the central permutations in the Mallows mixture are recovered exactly according to Corollary 3.5, their corresponding weights can be learned as well. To see the identifiability of the weights, we first establish a total variation bound for two Mallows mixtures with the same set of central permutations but different weights.

PROPOSITION 3.6. *Consider the Mallows mixtures* $\mathcal{M} = \sum_{i=1}^k w_i M(\pi_i)$ *and* $\mathcal{M}' = \sum_{i=1}^k w_i' M(\pi_i)$ *on* $\mathcal{S}_n$ *with a common noise parameter* $\phi \in (0,1)$. *Suppose that* $\xi \triangleq \max_{i\in[k]} |w_i - w_i'| > 0$. *Let* J *be a subset of* $[n]$ *such that* $\pi_i\|_J \ne \pi_j\|_J$ *for any distinct* $i, j \in [k]$. *Define* $\ell \triangleq |J|$ *and define* $\eta(k/2, \ell, \phi, 1)$ *as in* (13). *Then we have*

$$\mathsf{TV}\big(\mathcal{M}|_J, \mathcal{M}'|_J\big) \ge \xi \cdot \eta(k/2, \ell, \phi, 1).$$

Based on the above total variation lower bound, Algorithm 4 provides a method for estimating the weights in the Mallows mixture.

Similar to Algorithm 1, it is not hard to see that Algorithm 4 runs in polynomial time: First, a call to Algorithm 5 takes $O(k^3 n^2)$ time by Lemma 6.2. Then we need to search through the set $\mathcal{R}(L)$ of at most $L^k \approx k^k N^{k/2}$ candidate models, yielding a total time complexity $O(k^k N^{k/2} N N' n \log n)$ for comparing all the empirical models. Therefore, the overall time complexity is $O(k^3 n^2 + k^{k+1} N^{k/2+2} n (\log N)(\log n))$ in view of the definition of N'.

The following theorem bounds the entrywise error for $\hat{w}$ returned by Algorithm 4 and concludes this section.

THEOREM 3.7. *Suppose that we are given* N *i.i.d. observations sampled from the Mallows mixture* $\mathcal{M} = \sum_{i=1}^k w_i M(\pi_i)$ *on* $\mathcal{S}_n$ *with distinct central permutations* $\pi_1, \dots, \pi_k$ *and a known noise parameter* $\phi \in (0,1)$. *Fix a positive constant* $\gamma \le \min_{i\in[k]} w_i$ *and a probability of error* $\delta \in (0, 0.1)$. *Let* $\{\hat{\pi}_1, \dots, \hat{\pi}_k\}$ *be the set of permutations returned by Algorithm 3. Furthermore, let* $\hat{w} \in [0,1]^k$ *be the vector of weights returned by Algorithm 4. If*

Algorithm 4 `EstimateWeights`

Input: $\sigma_1, \dots, \sigma_N \in \mathcal{S}_n$, $\phi \in (0,1)$, $\gamma \in (0, 1/k]$, and distinct permutations $\hat{\pi}_1, \dots, \hat{\pi}_k \in \mathcal{S}_n$
Output: a vector of weights $\hat{w} \in [0,1]^k$

$\mathcal{I} \leftarrow$ the tuple returned by `FindTuple` (Algorithm 5) with inputs $\hat{\pi}_1, \dots, \hat{\pi}_k$
$J \leftarrow$ the set of all indices that appear in the tuple $\mathcal{I}$
$\mathcal{M}_N|_J \leftarrow \frac{1}{N} \sum_{m=1}^N \delta_{\sigma_m|_J}$
$L \leftarrow \lceil k N^{1/2} \rceil$
$N' \leftarrow \lceil k N \log N \rceil$
$\mathcal{R}(L) \leftarrow \{r \in [L]^k : r_i \ge \gamma L, \sum_{i=1}^k r_i = L\}$
for $r \in \mathcal{R}(L)$ **do**
    $\mathcal{M}'(r) \leftarrow \sum_{i=1}^k \frac{r_i}{L} M(\hat{\pi}_i, \phi)$
    generate N' i.i.d. random permutations $\sigma_1', \dots, \sigma_{N'}'$ from the Mallows mixture $\mathcal{M}'(r)$
    $\mathcal{M}'_{N'}(r)|_J \leftarrow \frac{1}{N'} \sum_{m=1}^{N'} \delta_{\sigma_m'|_J}$
end for
$\hat{w} \leftarrow \frac{1}{L} \operatorname{argmin}_{r\in\mathcal{R}(L)} \mathsf{TV}(\mathcal{M}'_{N'}(r)|_J, \mathcal{M}_N|_J)$
return $\hat{w}$

$N \geq \zeta(k, 2k+2, \phi, \gamma) \cdot \log \frac{2n^{2k+2}}{\delta}$ where ζ is defined in (16), then the following holds with probability at least $1-\delta$: Up to a relabeling, for every $i \in [k]$, it holds that $\hat{\pi}_i = \pi_i$ and

$$|\hat{w}_i - w_i| \leq \frac{(\log N)^{\ell+1}}{N^{1/2}} \big(\zeta(k, 2k-2, \phi, 1) \cdot \log(4/\delta)\big)^{1/2}.$$

The time complexity of the entire algorithm is $O(Nn^{2(k^2+k+2)}(\frac{ek}{\gamma(1-\phi)})^{(2k)^{2k+9}} \log \frac{1}{\delta} + k^{k+1}N^{k/2+2}n(\log N)(\log n))$.

4. Mallows mixture in high-noise regime. We turn to study the sample complexity for learning the Mallows mixture in the high-noise regime. For simplicity, we focus on the equally-weighted case. For a Mallows model on $\mathcal{S}_n$ with noise parameter $\phi \in (0, 1)$, we let $\varepsilon \triangleq 1 - \phi$ and consider the high-noise regime where n is fixed and $\varepsilon \to 0$, as which the Mallows model converges to the uniform distribution on $\mathcal{S}_n$. We are interested in how the sample complexity scales with $1/\varepsilon$.

More formally, let $\mathscr{M}_*$ denote the collection of k-mixtures of Mallows models on $\mathcal{S}_n$ with equal weights and a common noise parameter $\phi \in (0, 1)$, that is,

$$\mathscr{M}_* \equiv \mathscr{M}_*(n, k, \phi) \triangleq \left\{ \frac{1}{k} \sum_{i=1}^{k} M(\pi_i, \phi) : \pi_1, \ldots, \pi_k \in \mathcal{S}_n \right\}. \tag{17}$$

Some results in this section can be generalized to mixtures with different weights. However, we focus on the case of equally weighted mixtures to ease the notation, which already includes all the main ideas. The following result characterizes the total variation distance between two Mallows mixtures in the high-noise regime up to constant factors.

THEOREM 4.1. *For $m_k^* = \lfloor \log_2 k \rfloor + 1$ as defined in (9), the following statements hold as $\varepsilon = 1 - \phi \to 0$:*

(a) *Suppose that $k \leq 255$. For any distinct Mallows mixtures $\mathcal{M}$ and $\mathcal{M}'$ in $\mathscr{M}_*$, we have $\mathsf{TV}(\mathcal{M}, \mathcal{M}') = \Omega(\varepsilon^{m_k^*})$.*

(b) *On the other hand, for $n \geq 2m_k^*$, there exist distinct Mallows mixtures $\mathcal{M}$ and $\mathcal{M}'$ in $\mathscr{M}_*$ for which $\mathsf{TV}(\mathcal{M}, \mathcal{M}') = O(\varepsilon^{m_k^*})$.*

The hidden constants in $\Omega(\cdot)$ and $O(\cdot)$ above may depend on n and k.

The key to proving the above theorem is to view groups of pairwise comparisons as moments and relate them to the total variation distance between two Mallows mixtures. After establishing this link, the upper and lower bounds follow naturally from the two parts of Theorem 2.6, respectively.

Note that there is a condition $k \leq 255$ in part (a) of the above theorem. This is purely a technical assumption used in one step of the proof. We conjecture that the same result holds without this restriction on the number of components k in the mixture. See Section A.8 for details.

Theorem 4.1 characterizes the precise exponent of ε in the total variation distance between two Mallows mixtures. From this, we easily obtain matching upper and lower bounds of order $1/\varepsilon^{2m_k^*}$ on the optimal sample complexity for learning a Mallows k-mixture in the high-noise regime.

COROLLARY 4.2. *Suppose that for a Mallows mixture $\mathcal{M} \in \mathscr{M}_*$, we are given i.i.d. observations $\sigma_1, \ldots, \sigma_N \sim \mathcal{M}$, and let $\mathbb{P}_{\mathcal{M}}$ denote the associated probability measure. We let $\varepsilon \triangleq 1 - \phi$ and consider the setting where n is fixed and $\varepsilon \to 0$. For $m_k^* = \lfloor \log_2 k \rfloor + 1$ as defined in (9), the following statements hold*:

(a) *Suppose that* $k \le 255$, *and that* k *and* ϕ *are known. Let* $\mathcal{M}_N$ *denote the empirical distribution of* $\sigma_1, \dots, \sigma_N$ *with PMF* $f_{\mathcal{M}_N}(\sigma) = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}\{\sigma_i = \sigma\}$ *for each* $\sigma \in \mathcal{S}_n$. *Consider the minimum total variation distance estimator*

$$\widehat{\mathcal{M}} \triangleq \operatorname*{argmin}_{\mathcal{M}' \in \mathscr{M}_*} \mathsf{TV}(\mathcal{M}', \mathcal{M}_N). \tag{18}$$

If $N \ge C \log(\frac{1}{\delta})/\varepsilon^{2m_k^*}$ *for a sufficiently large constant* $C = C(n,k) > 0$ *and any* $\delta \in (0,1)$, *then we have*

$$\max_{\mathcal{M} \in \mathscr{M}_*} \mathbb{P}_{\mathcal{M}}\{\widehat{\mathcal{M}} \ne \mathcal{M}\} \le \delta.$$

(b) *On the other hand, if* $n \ge 2m_k^*$ *and* $N \le c/\varepsilon^{2m_k^*}$ *for a sufficiently small constant* $c = c(n,k) > 0$, *then we have*

$$\min_{\widetilde{\mathcal{M}}} \max_{\mathcal{M} \in \mathscr{M}_*} \mathbb{P}_{\mathcal{M}}\{\widetilde{\mathcal{M}} \ne \mathcal{M}\} \ge 1/8,$$

where the estimator $\widetilde{\mathcal{M}}$ *of the mixture is measurable with respect to the observations* $\sigma_1, \dots, \sigma_N$.

The computational complexity of the minimum total variation distance estimator $\widehat{\mathcal{M}}$ is polynomial in the sample size N, which itself depends polynomially on $1/\varepsilon$. Therefore, the estimator is polynomial-time in the high-noise regime where n is fixed and $1/\varepsilon$ grows. On the other hand, the computational cost depends exponentially on n, as it involves an exhaustive search over the class $\mathscr{M}_*$ in (17), whose cardinality grows as $(n!)^k$. Finding a statistically optimal estimator that is polynomial-time in n is an interesting open question.

Before ending this section, we remark that Liu and Moitra [23] proved an algorithmic lower bound for learning the Mallows mixture based on a local query model they proposed. In their model, upon receiving a query over a pair of sets $\{j_1, \dots, j_m\}, \{i_1, \dots, i_m\} \subset [n]$, the oracle returns the probability

$$\mathbb{P}_{\sigma \sim \mathcal{M}}\{\sigma(j_1) = i_1, \dots, \sigma(j_k) = i_k\}$$

up to an additive error $\tau > 0$. The cost of each query is defined to be $1/\tau^2$, and the total cost of an algorithm is the sum of its query costs. For $k = 2^{m-1}$ so that $m = \log_2 k + 1$, they presented two Mallows mixtures $\mathcal{M}$ and $\mathcal{M}'$ with a common noise parameter $\phi = 1 - \sqrt{k/n}$ that cannot be distinguished if $\tau \le (\frac{2k}{n})^{m/2}$. As a result, the query complexity for identifying $\mathcal{M}$ is at least $(\frac{n}{2k})^m = (\frac{1}{\sqrt{2}(1-\phi)})^{2m}$.

This local query complexity is defined in a different way from the sample complexity that we study. However, note that their lower bound $(\frac{1}{\sqrt{2}(1-\phi)})^{2m}$ is very similar to our lower bound of order $(\frac{1}{\varepsilon})^{2m} = (\frac{1}{1-\phi})^{2m}$ in Corollary 4.2(b); in particular, the exponent $2m = 2(\log_2 k + 1)$ is exactly the same in both bounds. This is because, ultimately, both lower bounds are proved by matching the combinatorial moments of two distinct mixtures of permutations. Compared to the particular instance considered by Liu and Moitra, we have formalized the combinatorial method of moments more generally and established matching upper and lower bounds on the sample complexity.

5. Discussion. In this work, we proposed a methodology to learn a mixture of permutations based on groups of pairwise comparisons. We first set up the framework using a generic noiseless model for a mixture of permutations. Then, we studied the Mallows mixture model, and introduced a polynomial-time algorithm for learning the central permutations with a sample complexity logarithmic in the size of the permutations. Finally, we studied the sample complexity for learning the Mallows mixture in a high-noise regime.

For the algorithms in this work, we assumed the knowledge of the noise parameter ϕ. This is indeed restrictive, but we conjecture that our main result on the logarithmic sample complexity in Section 3 continues to hold without this assumption. Specifically, the value of ϕ is needed in the definition of the class of mixtures (14), which is used in Algorithm 1. In the case where ϕ is unknown, we can augment the class of models (14) by allowing ϕ to take values in a fine grid in $(0, 1)$. In view of the continuity of the model in ϕ and good concentration properties of the model, we believe the same sample complexity can be proved without the knowledge of ϕ. We choose not to introduce this technical complication which does not add much to our general methodology.

Moreover, in general, a Mallows k-mixture model allows its components to have different noise parameters $\phi_1, \dots, \phi_k$. While the results in Section 4 depend strictly on the assumption of a common noise parameter ϕ, it is possible to adapt part of our approach in Section 3 to the heterogeneous setting. However, there is a fundamental obstacle which our current proof techniques cannot resolve. Namely, the success of Algorithm 1 relies on Proposition 3.1, which is a dimension-free lower bound on the total variation distance between two marginalized Mallows mixtures whose central permutations do not yield the same set of relative orders on J. The current proof of this lower bound (see Lemma A.7, which is a more general version of Proposition 3.1) leverages a block structure that makes up J and is ultimately based on Lemma A.2, an identifiability result for each block in the block structure. However, Lemma A.2 does not generalize to the setting where we have different noise parameters ϕ_i. For example, in the case where $n = 2$ and $k = 2$, identifiability no longer holds due to the extra degrees of freedom given by ϕ_1 and ϕ_2. We do not know how to get around this difficulty and defer a potential solution to future work.

Last but not least, our general approach of learning a mixture of permutations from groups of pairwise comparisons has potential applications beyond the Mallows mixture model. It would be interesting to apply the framework proposed in Section 2.3 to other models for mixtures of permutations, such as the Plackett-Luce model [37] and variations of the Mallows model [11].

6. Proofs. We now prove our results of Section 2. Additional proofs for Sections 3 and 4 are presented in the Supplementary Material [28].

6.1. *Proof of Theorem* 2.6(a). Throughout the proof, we write $m \equiv m_k^* \triangleq \lfloor \log_2 k \rfloor + 1$ as in (9). We start with a lemma which is the source of the logarithmic dependency of m on k.

LEMMA 6.1. *Consider a set Σ of k distinct permutations in $\mathcal{S}_n$ where $n \geq 2$. There exists $\pi^* \in \Sigma$ and a tuple $\mathcal{I}$ of ℓ pairs of distinct indices $(i_1, j_1), \dots, (i_\ell, j_\ell) \in [n]^2$, such that $\ell \leq \lfloor \log_2 k \rfloor$ and $\chi(\pi^*, \mathcal{I}) \neq \chi(\pi, \mathcal{I})$ for all $\pi \in \Sigma \setminus \{\pi^*\}$, where $\chi(\pi, \mathcal{I})$ is defined by (5). In addition, this tuple $\mathcal{I}$ can be found in polynomial time.*

PROOF. Let us start with $\Sigma_0 = \Sigma$ and apply the following bisection argument iteratively. Given a nonempty set Σ_{r-1} of distinct permutations where $r \geq 1$, it is easy to find a pair of indices $i_r, j_r \in [n]$ such that both of the following sets are nonempty:

$$\Sigma_r^+ = \{\pi \in \Sigma_{r-1} : \pi(i_r) > \pi(j_r)\} \quad \text{and} \quad \Sigma_r^- = \{\pi \in \Sigma_{r-1} : \pi(i_r) < \pi(j_r)\}. \tag{19}$$

Since $\Sigma_r^+ \sqcup \Sigma_r^- = \Sigma_{r-1}$, either Σ_r^+ or Σ_r^- has size at most $|\Sigma_{r-1}|/2$. We call it Σ_r so that $\Sigma_r \subset \Sigma_{r-1}$ and $|\Sigma_r| \leq |\Sigma_{r-1}|/2$. This procedure is iterated until we have $|\Sigma_r| = 1$.

For any $r \geq 1$, we have $|\Sigma_r| \leq k/2^r$ by construction. In particular,

$$|\Sigma_{\lfloor \log_2 k \rfloor}| \leq k/2^{\lfloor \log_2 k \rfloor} < 2.$$

Thus, there exists $\ell \le \lfloor \log_2 k \rfloor$ such that $|\Sigma_\ell| = 1$. We denote the permutation in Σ_ℓ by π^*. Note that by (19) and the definition of Σ_r, we have $\mathbb{1}\{\sigma(i_r) < \sigma(j_r)\} \neq \mathbb{1}\{\pi(i_r) < \pi(j_r)\}$ for any $\sigma \in \Sigma_r$ and $\pi \in \Sigma_{r-1} \setminus \Sigma_r$. Since the sets Σ_r's are nested, it holds that $\mathbb{1}\{\pi^*(i_r) < \pi^*(j_r)\} \neq \mathbb{1}\{\pi(i_r) < \pi(j_r)\}$ for any $\pi \in \Sigma_{r-1} \setminus \Sigma_r$ where $r \in [\ell]$. As a result, if we define $\mathcal{I} \triangleq ((i_1, j_1), \ldots, (i_\ell, j_\ell))$, then $\chi(\pi^*, \mathcal{I}) \neq \chi(\pi, \mathcal{I})$ for any $\pi \in \Sigma_0 \setminus \{\pi^*\}$. It is clear that $\mathcal{I}$ can be found in polynomial time, so the proof is complete. □

We now prove Theorem 2.6(a). Recall that our goal is to recover the k-mixture $\mathsf{M} = \sum_{i=1}^k w_i \delta_{\pi_i}$ of permutations in $\mathcal{S}_n$ from groups of m pairwise comparisons of the form $\chi(\pi, \mathcal{I})$ defined in (5) where $\pi \sim \mathsf{M}$. For this, we do an induction on $n \ge 2$, as the case $n = 1$ is vacuous.

Base case. For $n = 2$ and any $k \ge 1$, we can simply take $\mathcal{I}$ to be the tuple of m copies of $(1, 2)$. The oracle of Definition 2.1 then returns the distribution of $\chi(\pi, \mathcal{I})$, from which we immediately read off the distribution of $\mathbb{1}\{\pi(1) < \pi(2)\}$ and thus the distribution of π.

Induction hypothesis. As the induction hypothesis, we assume that the statement of Theorem 2.6(a) holds for $n - 1$ where $n \ge 3$. Consider a mixture $\sum_{s=1}^k w_s \delta_{\pi_s}$ of permutations in $\mathcal{S}_n$ which we aim to learn. Then each $\pi_s\|_{[n-1]}$ is a permutation in $\mathcal{S}_{n-1}$, and by definition (5), we have $\chi(\pi_s\|_{[n-1]}, \mathcal{I}) = \chi(\pi_s, \mathcal{I})$ for any tuple $\mathcal{I}$ of pairs of indices in $[n-1]$. Hence, the induction hypothesis implies that we can obtain the mixture $\sum_{s=1}^k w_s \delta_{\pi_s\|_{[n-1]}}$. To recover the mixture $\sum_{s=1}^k w_s \delta_{\pi_s}$ of permutations on $[n]$ from those on $[n-1]$, our task is to insert the index n into each permutation on $[n-1]$ at the correct position.

Induction step. Toward this end, let us apply Lemma 6.1 to the distinct elements of the set $\{\pi_1\|_{[n-1]}, \ldots, \pi_k\|_{[n-1]}\}$ of permutations in $\mathcal{S}_{n-1}$. Thus, there exists $s^* \in [k]$ and a tuple $\mathcal{I}$ of ℓ pairs of distinct indices in $[n-1]$, such that $\ell \le \lfloor \log_2 k \rfloor$ and $\chi(\pi_s, \mathcal{I}) \neq \chi(\pi_{s^*}, \mathcal{I})$ for all $s \in [k] \setminus S^*$ where we define

$$S^* \triangleq \big\{s \in [k] : \pi_s\|_{[n-1]} = \pi_{s^*}\|_{[n-1]}\big\}.$$

Next, for any index $r \in [n-1]$, we choose an m-tuple $\mathcal{I}_r$ consisting of all pairs of indices in $\mathcal{I}$ and also the pair (r, n). Such a tuple $\mathcal{I}_r$ can be chosen because $\ell \le \lfloor \log_2 k \rfloor = m - 1$. Then we query the group of m pairwise comparisons on $\mathcal{I}_r$ (Definition 2.1) to obtain the distribution $\sum_{s=1}^k w_s \delta_{\chi(\pi_s, \mathcal{I}_r)}$ for each $r \in [n-1]$. Recall that the definitions of $\mathcal{I}$ and S^* guarantee that $\chi(\pi_s, \mathcal{I}) = \chi(\pi_{s^*}, \mathcal{I})$ if and only if $s \in S^*$. Since $\mathcal{I}_r$ includes all pairs of indices in $\mathcal{I}$, we can distinguish those components of $\sum_{s=1}^k w_s \delta_{\chi(\pi_s, \mathcal{I}_r)}$ supported at $\chi(\pi_s, \mathcal{I}_r)$ with $s \in S^*$ from those with $s \in [k] \setminus S^*$. Therefore, we obtain the measure $\sum_{s \in S^*} w_s \delta_{\chi(\pi_s, \mathcal{I}_r)}$ for any $r \in [n-1]$.

Moreover, since $(r, n) \in \mathcal{I}_r$, from the measure $\sum_{s \in S^*} w_s \delta_{\chi(\pi_s, \mathcal{I}_r)}$, we can easily compute the function $f(r) \triangleq |\{\sum_{s \in S^*} w_s : \pi_s(r) < \pi_s(n)\}|$ where $r \in [n-1]$. In addition, we set $f(0) \triangleq |S^*|$. The measure $\sum_{s \in S^*} w_s \delta_{\pi_s}$ can be recovered from the sequence of numbers $\{f(r)\}_{r=1}^{n-1}$ as follows. By definition, the permutations $\pi_s\|_{[n-1]}$ for $s \in S^*$ are all the same, so by re-indexing $1, \ldots, n-1$, we can assume that they are all equal to the identity permutation $(1, 2, \ldots, n-1)$ to ease the notation. Then $f(r)$ is simply the total weight of permutations π_s that place n after r, so particularly the sequence $\{f(r)\}_{r=0}^{n-1}$ is nonincreasing. Moreover, $f(r) - f(r+1)$ is equal to the total weight of the permutations in the mixture $\sum_{s \in S^*} w_s \delta_{\pi_s}$ satisfying $\pi_s(n) = r + 1$. Therefore, we can recover the measure $\sum_{s \in S^*} w_s \delta_{\pi_s}$ from the sequence $\{f(r)\}_{r=1}^{n-1}$.

Finally, once we have learned the measure $\sum_{s \in S^*} w_s \delta_{\pi_s}$, the task becomes recovering the measure $\sum_{s \notin S^*} w_s \delta_{\pi_s}$ from the measure $\sum_{s \notin S^*} w_s \delta_{\pi_s\|_{[n-1]}}$, which can be done

by repeating the above procedure. Indeed, when querying a group of pairwise comparisons $\sum_{s=1}^{k} w_s \delta_{\chi(\pi_s, \mathcal{I}_r)}$, we can easily subtract the components with $s \in S^*$ to obtain $\sum_{s \notin S^*} w_s \delta_{\chi(\pi_s, \mathcal{I}_r)}$. Therefore, the above procedure can be iterated to eventually yield the entire mixture $\sum_{s=1}^{k} w_s \delta_{\pi_s}$. This completes the induction.

Time and sample complexity. To finish the proof, note that every step in this algorithmic construction is clearly polynomial-time. For the total number of groups of pairwise comparisons, recall that in the base case $n = 2$, we need one query, and in the induction step from $n - 1$ to n, we learn at least one component of the mixture from $n - 1$ queries. In summary, the total number of queries needed is at most $1 + k \sum_{i=2}^{n-1} i = 1 + \frac{k}{2}(n-2)(n+1)$.

6.2. *Proof of Theorem 2.6*(b). Throughout the proof, we write $m \equiv m_k^* \triangleq \lfloor \log_2 k \rfloor + 1$ as in (9) and fix $\ell \leq 2m - 1$. Intuitively, it is harder to identify a k-mixture of permutations in $\mathcal{S}_n$ for larger n and larger k. Indeed, let us justify that we can assume without loss of generality that $n = 2m$ and $k = 2^{m-1}$:

- Suppose that we can prove the statement of part (b) for $n = 2m$, that is, we have two mixtures $\frac{1}{k}\sum_{i=1}^{k} \delta_{\pi_i}$ and $\frac{1}{k}\sum_{i=1}^{k} \delta_{\pi'_i}$ of permutations in $\mathcal{S}_{2m}$ that cannot be identified using ℓ-wise comparisons. For any $n > 2m$, we may extend each of the above permutation to a permutation in $\mathcal{S}_n$ by defining $\pi_s(j) = \pi'_s(j) = j$ for all $s \in [k]$ and $2m < j \leq n$. Then ℓ-wise comparisons still cannot distinguish the two mixtures, because indices larger than $2m$ are completely uninformative. As a result, we may assume $n = 2m$ without loss of generality.
- Suppose that we can establish the desired result for k-mixtures. Then for any $k' > k$, if we define $\pi_s = \pi'_s = \mathsf{id}$ for all $k < s \leq k'$, then the mixtures $\frac{1}{k'}\sum_{i=1}^{k'} \delta_{\pi_i}$ and $\frac{1}{k'}\sum_{i=1}^{k'} \delta_{\pi'_i}$ still cannot be distinguished using groups of ℓ-wise comparisons. Hence the statement of the theorem also holds for k' in replace of k. For any fixed $m \in \mathbb{N}$, the smallest k such that $\lfloor \log_2 k \rfloor + 1 = m$ is equal to 2^{m-1}. Therefore, we may assume that $k = 2^{m-1}$ without loss of generality.

With these simplifications, for a fixed $m \in \mathbb{N}$, we now construct two sets Σ_1 and Σ_2 of permutations in $\mathcal{S}_{2m}$ such that $|\Sigma_1| = |\Sigma_2| = 2^{m-1}$, and such that groups of ℓ-wise comparisons cannot distinguish the two mixtures $\frac{1}{k}\sum_{\pi \in \Sigma_1} \delta_\pi$ and $\frac{1}{k}\sum_{\pi \in \Sigma_2} \delta_\pi$. For each vector $v \in \{0,1\}^m$, we define a permutation $\pi_v \in \mathcal{S}_m$ by

$$\begin{cases} \pi_v(2j-1) = 2j-1, & \pi_v(2j) = 2j \quad \text{if } v_j = 0 \\ \pi_v(2j-1) = 2j, & \pi_v(2j) = 2j-1 \quad \text{if } v_j = 1 \end{cases} \quad \text{for all } j \in [m]. \tag{20}$$

Moreover, we define

$$\Sigma_1 \triangleq \{\pi_v : v \in \{0,1\}^m, \|v\|_1 \text{ is odd}\} \quad \text{and} \quad \Sigma_2 \triangleq \{\pi_v : v \in \{0,1\}^m, \|v\|_1 \text{ is even}\}.$$

It is clear that both Σ_1 and Σ_2 have cardinality 2^{m-1}.

Next, consider an arbitrary set of indices $J \subset [2m]$ with $|J| = 2m - 1$. We claim that

$$\{\pi_v\|_J : v \in \{0,1\}^m, \|v\|_1 \text{ is odd}\} = \{\pi_v\|_J : v \in \{0,1\}^m, \|v\|_1 \text{ is even}\}. \tag{21}$$

To prove this claim, let j_1 denote the only element of $[n] \setminus J$. If j_1 is odd, we let $j_2 = j_1 + 1$; otherwise, we let $j_2 = j_1 - 1$. For any $v \in \{0,1\}^m$ with odd $\|v\|_1$, we define $v' \in \{0,1\}^m$ by $v'_i = 1 - v_i$ for $i = \lceil j_1/2 \rceil$ and $v'_i = v_i$ for $i \neq \lceil j_1/2 \rceil$. Since v' differs from v in only one coordinate, $\|v'\|_1$ must be even. As a result, we have $\pi_v \in \Sigma_1$ and $\pi_{v'} \in \Sigma_2$. This clearly gives a bijection between the sets Σ_1 and Σ_2. Furthermore, by definition (20), π_v and $\pi_{v'}$ only differ on the pair (j_1, j_2). Since J does not contain j_1, we must have $\pi_v\|_J = \pi_{v'}\|_J$. Consequently, equation (21) holds, so that any ℓ-wise comparison (Definition 2.3) returns the same distribution for the two mixtures $\frac{1}{k}\sum_{\pi \in \Sigma_1} \delta_\pi$ and $\frac{1}{k}\sum_{\pi \in \Sigma_2} \delta_\pi$. This completes the proof.

Algorithm 5 `FindTuple`

Input: distinct permutations $\pi_1, \dots, \pi_k$ in $\mathcal{S}_n$
Output: a tuple $\mathcal{I}$ of pairs of indices in $[n]$

$\mathcal{I} \leftarrow [\,]$ ▷ $[\,]$ *denotes the empty tuple*
for $j = 2$ to k **do**
  if there exists $i \in [j-1]$ for which $\chi(\pi_i, \mathcal{I}) = \chi(\pi_j, \mathcal{I})$ **then**
    ▷ *by convention,* $\chi(\pi_1, [\,]) = \chi(\pi_2, [\,])$
    find $r, s \in [n]$ such that $\pi_i(r) < \pi_i(s)$ and $\pi_j(r) > \pi_j(s)$
    $\mathcal{I} \leftarrow [\mathcal{I}, (r, s)]$ ▷ *that is, append* (r, s) *to* $\mathcal{I}$
  end if
end for
return $\mathcal{I}$

6.3. *Proof of Theorem 2.7.* We first establish a lemma which guarantees the success of Algorithm 5 which finds a discriminative tuple for any given set of permutations.

LEMMA 6.2. *Let $\pi_1, \dots, \pi_k$ be k distinct permutations in $\mathcal{S}_n$. Algorithm 5 finds in $O(k^3 n^2)$ time a tuple $\mathcal{I}$ of ℓ pairs of distinct indices in $[n]$ such that $\ell \le k-1$ and $\chi(\pi_i, \mathcal{I}) \ne \chi(\pi_j, \mathcal{I})$ for any distinct $i, j \in [k]$.*

PROOF. First, it is clear that Algorithm 5 returns a tuple $\mathcal{I}$ of $\ell \le k-1$ pairs of indices. It suffices to inductively show that, at step j of the loop in the algorithm where $j = 2, \dots, k$, we have $\chi(\pi_i, \mathcal{I}) \ne \chi(\pi_{i'}, \mathcal{I})$ for any distinct $i, i' \in [j]$.

For $j = 2$, because π_1 and π_2 are assumed to be distinct, we can find indices $r, s \in [n]$ such that $\pi_1(r) < \pi_2(s)$ and $\pi_2(r) > \pi_2(s)$. Therefore, if we let $\mathcal{I}$ consist of the single pair (r, s), then $\chi(\pi_1, \mathcal{I}) \ne \chi(\pi_2, \mathcal{I})$.

Next, at the beginning of step j of the loop where $3 \le j \le k$, we have $\chi(\pi_i, \mathcal{I}) \ne \chi(\pi_{i'}, \mathcal{I})$ for any distinct $i, i' \in [j-1]$ by the induction hypothesis. If $\chi(\pi_i, \mathcal{I}) \ne \chi(\pi_j, \mathcal{I})$ for all $i \in [j-1]$, then we are done with the induction. Otherwise, there exists exactly one $i \in [j-1]$ such that $\chi(\pi_i, \mathcal{I}) = \chi(\pi_j, \mathcal{I})$. Since π_i and π_j are assumed to be distinct, we can find indices $r, s \in [n]$ such that $\pi_i(r) < \pi_i(s)$ and $\pi_j(r) > \pi_j(s)$. As a result, once we append the pair (r, s) to the tuple $\mathcal{I}$, it then holds that $\chi(\pi_i, \mathcal{I}) \ne \chi(\pi_j, \mathcal{I})$. We now have $\chi(\pi_i, \mathcal{I}) \ne \chi(\pi_{i'}, \mathcal{I})$ for any distinct $i, i' \in [j]$, finishing the induction.

Finally, the time complexity of the algorithm is $O(k^3 n^2)$ because it searches through $j = 2, \dots, k$ and $i = 1, \dots, j-1$; for a pair (i, j), binary vectors of length at most $k-1$ are compared; and finding a pair (r, s) takes time $O(n^2)$. □

We now prove Theorem 2.7 for $n \ge 2$ as the case $n = 1$ is trivial. This proof is structurally similar to Theorem 2.6(a), but the key step in the induction is different. We note an intricacy throughout this proof: When queried with the tuple $\mathcal{I}$, the oracle in Definition 2.2 returns the *set* $\{\chi(\pi_i, \mathcal{I}) : i \in [k]\}$ where $\chi(\pi_i, \mathcal{I})$ is defined by (5). This set (as opposed to an ordered tuple or multiset) is represented by its distinct elements without labels, so it is possible that it contains less than k distinct elements and we do not know their multiplicities.

Note that Algorithm 6 is recursive with respect to n; correspondingly, we prove Theorem 2.7 by induction on $n \ge 2$.

Base case (lines 1–9 of Algorithm 6). For $n = 2$, as in Algorithm 6, we simply take $\mathcal{I}$ to be the tuple of $k+1$ copies of $(1, 2)$. Then every entry of $\chi(\pi_i, \mathcal{I})$ is equal to $\mathbb{1}\{\pi_i(1) < \pi_i(2)\}$. The oracle returns the set $\{\chi(\pi_i, \mathcal{I}) : i \in [k]\}$, from which we immediately read off the set $\{\mathbb{1}\{\pi_i(1) < \pi_i(2)\} : i \in [k]\}$ and thus the set $\{\pi_i : i \in [k]\}$ as detailed in the algorithm.

Algorithm 6 `InsertionDemixing`

Input: n, k, and $\{\chi(\pi_i, \mathcal{I}) : i \in [k]\}$ for tuples $\mathcal{I}$ of $k+1$ pairs of distinct indices in $[n]$ queried in the algorithm

Output: a set $\mathfrak{S}_n$ of permutations in $\mathcal{S}_n$

1: **if** $n = 2$ **then**
2: $\quad\mathcal{I} \leftarrow$ the tuple of $k+1$ copies of $(1, 2)$
$\quad\quad\triangleright$ *all entries of* $\chi(\pi_i, \mathcal{I})$ *are equal to* $\mathbb{1}\{\pi_i(1) < \pi_i(2)\}$
3: $\quad$**if** $\{\chi(\pi_i, \mathcal{I})_1 : i \in [k]\} = \{1\}$ **then**
4: $\quad\quad\mathfrak{S}_2 \leftarrow \{(1, 2)\}$
5: $\quad$**else if** $\{\chi(\pi_i, \mathcal{I})_1 : i \in [k]\} = \{0\}$ **then**
6: $\quad\quad\mathfrak{S}_2 \leftarrow \{(2, 1)\}$
7: $\quad$**else** $\quad\triangleright\ \{\chi(\pi_i, \mathcal{I})_1 : i \in [k]\} = \{0, 1\}$
8: $\quad\quad\mathfrak{S}_2 \leftarrow \{(1, 2), (2, 1)\}$
9: $\quad$**end if**
10: **else** $\quad\triangleright\ n \geq 3$
11: $\quad\mathfrak{S}_{n-1} \leftarrow$ the subset of $\mathcal{S}_{n-1}$ returned by `InsertionDemixing` run with inputs $n-1$, k, and $\{\chi(\pi_i, \mathcal{I}) : i \in [k]\}$ for tuples $\mathcal{I}$ of $k+1$ pairs of indices in $[n-1]$
12: $\quad$let $\sigma_1, \ldots, \sigma_{k'} \in \mathcal{S}_{n-1}$ denote the elements of $\mathfrak{S}_{n-1}$
13: $\quad$**if** $k' > |\mathfrak{S}_{n-2}|$ **then**
14: $\quad\quad\mathcal{I} \leftarrow$ the tuple returned by `FindTuple` (Algorithm 5) with inputs $\sigma_1, \ldots, \sigma_{k'}$
15: $\quad$**end if**
16: $\quad\mathfrak{S}_n \leftarrow \varnothing$
17: $\quad$**for** $j = 1$ to k' **do**
18: $\quad\quad$**for** $r = 2$ to $n-1$ **do**
19: $\quad\quad\quad\mathcal{I}_r \leftarrow [\mathcal{I}, \ldots, (\sigma_j^{-1}(r-1), n), (n, \sigma_j^{-1}(r))]$ $\quad\triangleright$ *here* $\ldots$ *means some arbitrary pairs of indices we append to* $\mathcal{I}$ *for concreteness so that* $\mathcal{I}_r$ *consists of exactly* $k+1$ *pairs*
20: $\quad\quad\quad$call the oracle in Definition 2.2 to obtain the set $\{\chi(\pi_i, \mathcal{I}_r) : i \in [k]\}$
21: $\quad\quad\quad\mathfrak{X}_j(r) \leftarrow \{\chi(\pi_i, \mathcal{I}_r) : i \in [k], \chi(\pi_i, \mathcal{I}) = \chi(\sigma_j, \mathcal{I})\}$
22: $\quad\quad\quad$**if** there is $v \in \mathfrak{X}_j(r)$ such that $v_k = v_{k+1} = 1$ **then**
23: $\quad\quad\quad\quad\mathfrak{S}_n \leftarrow \mathfrak{S}_n \cup \{(\sigma_j^{-1}(1), \ldots, \sigma_j^{-1}(r-1), n, \sigma_j^{-1}(r), \ldots, \sigma_j^{-1}(n-1))\}$
24: $\quad\quad\quad$**end if**
25: $\quad\quad$**end for**
26: $\quad\quad$**if** there is $v \in \mathfrak{X}_j(2)$ such that $v_k = 0$ **then**
27: $\quad\quad\quad\mathfrak{S}_n \leftarrow \mathfrak{S}_n \cup \{(n, \sigma_j^{-1}(1), \ldots, \sigma_j^{-1}(n-1))\}$
28: $\quad\quad$**end if**
29: $\quad\quad$**if** there is $v \in \mathfrak{X}_j(n-1)$ such that $v_{k+1} = 0$ **then**
30: $\quad\quad\quad\mathfrak{S}_n \leftarrow \mathfrak{S}_n \cup \{(\sigma_j^{-1}(1), \ldots, \sigma_j^{-1}(n-1), n)\}$
31: $\quad\quad$**end if**
32: $\quad$**end for**
33: **end if**
34: **return** $\mathfrak{S}_n$

Induction hypothesis (*lines* 10–12 *of Algorithm* 6). Fix $n \geq 3$ and assume that the conclusion of the theorem holds for $n-1$. Consider a mixture of permutations $\pi_1, \ldots, \pi_k \in \mathcal{S}_n$ which we aim to learn. Then each $\pi_i\|_{[n-1]}$ where $i \in [k]$ is a permutation in $\mathcal{S}_{n-1}$. By definition (5), we have $\chi(\pi_i\|_{[n-1]}, \mathcal{I}) = \chi(\pi_i, \mathcal{I})$ for any tuple $\mathcal{I}$ of pairs of indices in $[n-1]$. Hence, the induction hypothesis implies that the algorithm returns the set of permutations $\mathfrak{S}_{n-1} = \{\pi_i\|_{[n-1]} : i \in [k]\}$.

Let us denote the distinct elements of $\mathfrak{S}_{n-1}$ by $\sigma_1, \ldots, \sigma_{k'} \in \mathcal{S}_{n-1}$, where $k' \le k$. To obtain the set $\{\pi_i : i \in [k]\}$ of permutations on $[n]$ from those on $[n-1]$, our task is to insert the index n into σ_j at the correct position for each $j \in [k']$. Note that $|\{\pi_i\|_{[n-1]} : i \in [k]\}| \le |\{\pi_i : i \in [k]\}|$ and we may need to obtain more than one permutation in $\mathcal{S}_n$ from each σ_j where $j \in [k']$.

Induction step (*lines 14–32 of Algorithm 6*). Run Algorithm 5 with inputs $\sigma_1, \ldots, \sigma_{k'}$. Lemma 6.2 guarantees that we obtain an ℓ-tuple $\mathcal{I}$ of pairs of distinct indices in $[n-1]$ such that $\ell \le k' - 1$ and $\chi(\sigma_i, \mathcal{I}) \ne \chi(\sigma_j, \mathcal{I})$ for any distinct $i, j \in [k']$. This guarantees that $\pi_i\|_{[n-1]} = \sigma_j$ if and only if $\chi(\pi_i, \mathcal{I}) = \chi(\sigma_j, \mathcal{I})$ for $i \in [k]$ and $j \in [k']$.

We now fix $j \in [k']$ (line 17) and aim to recover those π_i such that $\pi_i\|_{[n-1]} = \sigma_j$. To simplify the notation in the sequel, we assume that σ_j is equal to the identity permutation on $[n-1]$, denoted by $(1, 2, \ldots, n-1)$. Algorithm 6 is stated in full generality, and the proof of its validity is essentially the same. With the assumption $\pi_i\|_{[n-1]} = (1, 2, \ldots, n-1)$, to recover π_i, it suffices to determine where n should be inserted into $(1, 2, \ldots, n-1)$.

Note that $k + 1 \ge k' + 1 \ge \ell + 2$. Hence, for each $r = 2, \ldots, n-1$, we can define an $(k+1)$-tuple $\mathcal{I}_r$ (line 19) containing all the ℓ pairs of indices in $\mathcal{I}$ and the pairs $(r-1, n)$ and (n, r). In the case that $k + 1 > \ell + 2$, the remaining $k - \ell - 1$ pairs of indices in $\mathcal{I}_r$ can be defined arbitrarily for concreteness—we will not use the comparison information on those pairs. Then, we query the group of pairwise comparisons on $\mathcal{I}_r$ according to Definition 2.2 to obtain the set $\{\chi(\pi_i, \mathcal{I}_r) : i \in [k]\}$.

Since $\mathcal{I}_r$ includes all pairs of indices in $\mathcal{I}$, we can compute

$$\mathfrak{X}_j(r) \triangleq \big\{\chi(\pi_i, \mathcal{I}_r) : i \in [k],\, \chi(\pi_i, \mathcal{I}) = \chi(\sigma_j, \mathcal{I})\big\} \subset \big\{\chi(\pi_i, \mathcal{I}_r) : i \in [k]\big\}.$$

Note that $\pi_i\|_{[n-1]} = \sigma_j$ if and only if $\chi(\pi_i, \mathcal{I}) = \chi(\sigma_j, \mathcal{I})$. By the definition of $\mathfrak{X}_j(r)$, for each fixed $r = 2, \ldots, n-1$, we have that $\pi_i\|_{[n-1]} = \sigma_j$ if and only if $\chi(\pi_i, \mathcal{I}_r) \in \mathfrak{X}_j(r)$ for $i \in [k]$. In particular, when we take $v \in \mathfrak{X}_j(r)$ in the algorithm, $v = \chi(\pi_i, \mathcal{I}_r)$ for some π_i such that $\pi_i\|_{[n-1]} = \sigma_j$.

It remains to recover π_i for which $\pi_i\|_{[n-1]} = \sigma_j$ from the collection of sets $\{\mathfrak{X}_j(r)\}_{r=2}^{n-1}$. First, fix r and recall that the pairs $(r-1, n)$ and (n, r) are both in $\mathcal{I}_r$. If

$$\pi_i = (1, \ldots, r-1, n, r, \ldots, n-1)$$

for some $i \in [k]$, then $\pi_i\|_{[n-1]} = \sigma_j$ and the set $\mathfrak{X}_j(r)$ must contain a vector $v = \chi(\pi_i, \mathcal{I}_r)$ whose entries $v_k = \mathbb{1}\{\pi_i(r-1) < \pi_i(n)\}$ and $v_{k+1} = \mathbb{1}\{\pi_i(n) < \pi_i(r)\}$ are both equal to 1. Conversely, if $\mathfrak{X}_j(r)$ contains some vector v with $v_k = v_{k+1} = 1$, then we know that $\pi_i\|_{[n-1]} = \sigma_j$ and π_i must be equal to $(1, \ldots, r-1, n, r, \ldots, n-1)$. As a result, we successfully recover this π_i (line 23).

This argument clearly works for π_i equal to $(n, 1, \ldots, n-1)$ or $(1, \ldots, n-1, n)$ as well, in the respective cases (lines 26–31):

- $r = 2$ and $v_k = \chi(\pi_i, \mathcal{I}_2)_k = \mathbb{1}\{\pi_i(1) < \pi_i(n)\} = 0$;
- $r = n-1$ and $v_{k+1} = \chi(\pi_i, \mathcal{I}_{n-1})_{k+1} = \mathbb{1}\{\pi_i(n) < \pi_i(n-1)\} = 0$.

Therefore, we are able to recover all distinct π_i such that $\pi_i\|_{[n-1]} = \sigma_j$ for $i \in [k]$.

Finally, repeating the above procedure for each $j \in [k']$ yields the set $\{\pi_i : i \in [k]\}$.

Time and sample complexity. In each recursion of Algorithm 6, the bottleneck of time complexity is a call to Algorithm 5 (line 14) which takes $O(k^3 n^2)$ time, but this step only needs to be run at most k times. Moreover, the step of inserting n to the current mixture takes less than $O(k^4 n)$ time (lines 17–32). As a result, the overall time complexity is $O(k^4 n^2)$.

For the total number of groups of pairwise comparisons, recall that in the base case $n = 2$, we need one query, and in the induction step from $n-1$ to n, we learn at least one component of the mixture from $n-2$ queries. In summary, the total number of queries needed is at most $1 + k\sum_{i=3}^{n}(i-2) = 1 + \frac{k}{2}(n-2)(n-1)$.

Acknowledgements. The authors thank the anonymous reviewers for their helpful comments.

Funding. C.M. was supported in part by the NSF Grant DMS-2053333. Y.W. was supported in part by the NSF Grant CCF-1900507, the NSF CAREER award CCF-1651588, and an Alfred Sloan fellowship.

SUPPLEMENTARY MATERIAL

Supplement to "Learning mixtures of permutations: Groups of pairwise comparisons and combinatorial method of moments" (DOI: 10.1214/22-AOS2185SUPP; .pdf). We provide additional technical proofs of our results in this supplement.

REFERENCES

[1] AWASTHI, P., BLUM, A., SHEFFET, O. and VIJAYARAGHAVAN, A. (2014). Learning mixtures of ranking models. In *Advances in Neural Information Processing Systems* 2609–2617.

[2] BALTRUNAS, L., MAKCINSKAS, T. and RICCI, F. (2010). Group recommendations with rank aggregation and collaborative filtering. In *Proceedings of the Fourth ACM Conference on Recommender Systems* 119–126.

[3] BORDA, J. C. (1781). Mémoire sur les élections au scrutin. *Histoire de l'Academie Royale des Sciences Pour*.

[4] BRAVERMAN, M. and MOSSEL, E. (2009). Sorting from noisy information. Preprint. Available at arXiv:0910.1191.

[5] BUSA-FEKETE, R., FOTAKIS, D., SZÖRÉNYI, B. and ZAMPETAKIS, M. (2019). Optimal learning of Mallows block model. In *Proceedings of the Thirty-Second Conference on Learning Theory* (A. Beygelzimer and D. Hsu, eds.) **99** 529–532.

[6] BUSA-FEKETE, R., HÜLLERMEIER, E. and SZÖRÉNYI, B. (2014). Preference-based rank elicitation using statistical models: The case of Mallows. In *Proceedings of the* 31*st International Conference on Machine Learning* **32** II–1071. JMLR.org.

[7] BUSSE, L. M., ORBANZ, P. and BUHMANN, J. M. (2007). Cluster analysis of heterogeneous rank data. In *Proceedings of the* 24*th International Conference on Machine Learning* 113–120. ACM, New York.

[8] CARAGIANNIS, I., PROCACCIA, A. D. and SHAH, N. (2013). When do noisy votes reveal the truth? In *Proceedings of the Fourteenth ACM Conference on Electronic Commerce* 143–160.

[9] CHIERICHETTI, F., DASGUPTA, A., KUMAR, R. and LATTANZI, S. (2015). On learning mixture models for permutations. In *ITCS'*15*—Proceedings of the* 6*th Innovations in Theoretical Computer Science* 85–92. ACM, New York. MR3418998

[10] CONDORCET, M. J. (1785). *Essai sur L'application de L'analyse à la Probabilité des Décisions Rendues à la Pluralité des Voix*.

[11] DE, A., O'DONNELL, R. and SERVEDIO, R. (2018). Learning sparse mixtures of rankings from noisy information. Preprint. Available at arXiv:1811.01216.

[12] DOIGNON, J.-P., PEKEČ, A. and REGENWETTER, M. (2004). The repeated insertion model for rankings: Missing link between two subset choice models. *Psychometrika* **69** 33–54. MR2272438 https://doi.org/10.1007/BF02295838

[13] DOSS, N., WU, Y., YANG, P. and ZHOU, H. H. (2020). Optimal estimation of high-dimensional Gaussian mixtures. Preprint. Available at arXiv:2002.05818.

[14] DWORK, C., KUMAR, R., NAOR, M. and SIVAKUMAR, D. (2001). Rank aggregation methods for the web. In *Proceedings of the* 10*th International Conference on World Wide Web* 613–622.

[15] FAGIN, R., KUMAR, R. and SIVAKUMAR, D. (2003). Efficient similarity search and classification via rank aggregation. In *Proceedings of the* 2003 *ACM SIGMOD International Conference on Management of Data* 301–312.

[16] FLIGNER, M. A. and VERDUCCI, J. S. (1986). Distance based ranking models. *J. Roy. Statist. Soc. Ser. B* **48** 359–369. MR0876847

[17] GORMLEY, I. C. and MURPHY, T. B. (2008). Exploring voting blocs within the Irish electorate: A mixture modeling approach. *J. Amer. Statist. Assoc.* **103** 1014–1027. MR2528824 https://doi.org/10.1198/016214507000001049

[18] GORMLEY, I. C. and MURPHY, T. B. (2008). A mixture of experts model for rank data with applications in election studies. *Ann. Appl. Stat.* **2** 1452–1477. MR2655667 https://doi.org/10.1214/08-AOAS178

[19] HEINRICH, P. and KAHN, J. (2018). Strong identifiability and optimal minimax rates for finite mixture estimation. *Ann. Statist.* **46** 2844–2870. MR3851757 https://doi.org/10.1214/17-AOS1641

[20] IRUROZKI, E., CALVO, B. and LOZANO, J. A. (2019). Mallows and generalized Mallows model for matchings. *Bernoulli* **25** 1160–1188. MR3920369 https://doi.org/10.3150/17-bej1017

[21] JORDAN, M. I. and JACOBS, R. A. (1994). Hierarchical mixtures of experts and the EM algorithm. *Neural Comput.* **6** 181–214.

[22] KORBA, A., CLÉMENÇON, S. and SIBONY, E. (2017). A learning theory of ranking aggregation. In *Artificial Intelligence and Statistics* 1001–1010.

[23] LIU, A. X. and MOITRA, A. (2018). Efficiently learning mixtures of Mallows models. In 59*th Annual IEEE Symposium on Foundations of Computer Science—FOCS* 2018 627–638. IEEE Computer Soc., Los Alamitos, CA. MR3899628 https://doi.org/10.1109/FOCS.2018.00066

[24] LIU, Y.-T., LIU, T.-Y., QIN, T., MA, Z.-M. and LI, H. (2007). Supervised rank aggregation. In *Proceedings of the* 16*th International Conference on World Wide Web* 481–490.

[25] LU, T. and BOUTILIER, C. (2011). Learning Mallows models with pairwise preferences. In *Proceedings of the* 28*th International Conference on International Conference on Machine Learning* 145–152.

[26] LU, T. and BOUTILIER, C. (2014). Effective sampling and learning for Mallows models with pairwise-preference data. *J. Mach. Learn. Res.* **15** 3783–3829. MR3317212

[27] MALLOWS, C. L. (1957). Non-null ranking models. I. *Biometrika* **44** 114–130. MR0087267 https://doi.org/10.1093/biomet/44.1-2.114

[28] MAO, C. and WU, Y. (2022). Supplement to "Learning mixtures of permutations: Groups of pairwise comparisons and combinatorial method of moments." https://doi.org/10.1214/22-AOS2185SUPP

[29] MARDEN, J. I. (1995). *Analyzing and Modeling Rank Data. Monographs on Statistics and Applied Probability* **64**. CRC Press, London. MR1346107

[30] MEILĂ, M. and CHEN, H. (2010). Dirichlet process mixtures of generalized Mallows models. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence* 358–367.

[31] MEILĂ, M., PHADNIS, K., PATTERSON, A. and BILMES, J. (2007). Consensus ranking under the exponential model. In *Proceedings of the Twenty-Third Conference on Uncertainty in Artificial Intelligence* 285–294.

[32] MOITRA, A. and VALIANT, G. (2010). Settling the polynomial learnability of mixtures of Gaussians. In 2010 *IEEE* 51*st Annual Symposium on Foundations of Computer Science—FOCS* 2010 93–102. IEEE Computer Soc., Los Alamitos, CA. MR3024779

[33] MURPHY, T. B. and MARTIN, D. (2003). Mixtures of distance-based models for ranking data. *Comput. Statist. Data Anal.* **41** 645–655. MR1973732 https://doi.org/10.1016/S0167-9473(02)00165-2

[34] PEARSON, K. (1894). Contributions to the mathematical theory of evolution. *Philos. Trans. R. Soc. Lond. A* **185** 71–110.

[35] WU, Y. and YANG, P. (2020). Optimal estimation of Gaussian mixtures via denoised method of moments. *Ann. Statist.* **48** 1981–2007. MR4134783 https://doi.org/10.1214/19-AOS1873

[36] ZAGIER, D. (1992). Realizability of a model in infinite statistics. *Comm. Math. Phys.* **147** 199–210. MR1171767

[37] ZHAO, Z., PIECH, P. and XIA, L. (2016). Learning mixtures of Plackett–Luce models. In *International Conference on Machine Learning* 2906–2914.