The Kendall and Mallows Kernels for Permutations

Yunlong Jiao, Jean-Philippe Vert

The Kendall and Mallows Kernels for Permutations

Yunlong Jiao[1,2,3], Jean-Philippe Vert[1,2,3]

[1]MINES ParisTech, PSL Research University,
CBIO – Centre for Computational Biology,
77300 Fontainebleau, France
[2]Institut Curie, 75248 Paris Cedex, France
[3]INSERM, U900, 75248 Paris Cedex, France
[4]Ecole Normale Supérieure,
Department of Mathematics and their Applications,
75005 Paris, France

Emails: [name].[surname]@mines-paristech.fr

Abstract

We show that the widely used Kendall tau correlation coefficient, and the related Mallows kernel, are positive definite kernels for permutations. They offer computationally attractive alternatives to more complex kernels on the symmetric group to learn from rankings, or learn to rank. We show how to extend these kernels to partial rankings, multivariate rankings and uncertain rankings. Examples are presented on how to formulate typical problems of learning from rankings such that they can be solved with state-of-the-art kernel algorithms. We demonstrate promising results on clustering heterogeneous rank data and high-dimensional classification problems in biomedical applications.

1 Introduction

A permutation is a 1-to-1 mapping from a finite set into itself. Assuming the finite set is ordered, a permutation can equivalently be represented by a total ranking of the elements of the set. Permutations are ubiquitous in many applications involving preferences, rankings or matching, such as modeling and analyzing data describing the preferences or votes of a population [18, 52], learning or tracking correspondences between sets of objects [32], or estimating a consensus ranking that best represents a collection of individual rankings [19, 1, 3]. Another potentially rich source of rank data comes from real-valued vectors in which the relative ordering of the values of multiple features is more important than their absolute magnitude. For example, in the case of high-dimensional gene expression data,

[24] showed that simple classifiers based on binary comparisons between the expression of different genes in a sample show competitive prediction accuracy with much more complex classifiers built on quantitative gene expression levels, a line of thoughts that have been further investigated by [71, 79, 49]. In these approaches, an n-dimensional feature vector is first transformed into a vector of ranks by sorting its entries, which are presented as input to training a classifier.

Working with permutations is, however, computationally challenging. There are $n!$ permutations over n items, suggesting that various simplifications or approximations are necessary in pursuit of efficient algorithms to analyze or learn permutations. Such simplifications include for example, reducing ranks to a series of binary decisions [1, 6], or estimating a parametric distribution over permutations [47, 31, 32].

Kernel algorithms form a class of methods that have been proved successful in numerous applications and enjoy great popularity in the machine learning community [13, 75, 60, 65]. The essential idea behind these methods is to define a symmetric positive definite kernel $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ over an input space $\mathcal{X}$, which expresses our belief of similarities between pairs of points in the input space, and which implicitly defines an embedding $\Phi : \mathcal{X} \to \mathcal{F}$ of the input space $\mathcal{X}$ to a Hilbert space $\mathcal{F}$ in which the kernel becomes an inner product:

$$\forall \mathbf{x}, \mathbf{x}' \in \mathcal{X}, \quad K(\mathbf{x}, \mathbf{x}') = \langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \rangle_{\mathcal{F}} .$$

Key to kernel methods is the fact that kernel algorithms only manipulate data through evaluation of the kernel function, allowing to work implicitly in the potentially high- or even infinite-dimensional space $\mathcal{F}$. This *kernel trick* is particularly interesting when $K(\mathbf{x}, \mathbf{x}')$ is inexpensive to evaluate, compared to $\Phi(\mathbf{x})$ and $\Phi(\mathbf{x}')$. In particular, kernel methods have found many applications where the input data are discrete or structured, such as strings or graphs, thanks to the development of numerous kernels for these data [30, 36, 23, 65, 63, 76].

In this context, it is surprising that relatively little attention has been paid to the problem of defining positive definite kernels between permutations, which could pave the way to benefiting from computationally efficient kernel methods in problems involving permutations. A notable exception is the work of [41, 43], who exploit the fact that the right-invariant positive definite kernels on the symmetric group are fully characterized by Bochner's theorem [41, 22]. They derive interesting kernels, such as a diffusion kernel for rankings or partial rankings, and demonstrate that kernel methods are flexible to handle rank data of diverse types. However, the kernels proposed in their papers have typically a computational complexity that grows exponentially with the number of items to rank, and remain prohibitive to compute for more than a few items.

In this paper we study new computationally attractive positive definite kernels for permutations and rankings. Our main contribution is to show that two widely-used and computationally efficient measures of similarity between permutations, the Kendall tau correlation coefficient and the Mallows kernel, are positive definite. Although these measures compare two permutations of n items in terms of $\binom{n}{2}$ pairwise comparisons, they can be computed in $O(n \log n)$, which allows us to use kernel methods for problems involving rank data over a large number of items. We show how these kernels can be extended to partial rankings, multivariate rankings, and uncertain rankings which are particularly relevant when the rankings are obtained by sorting a real-valued vector where ties or almost-ties occur. We illustrate

the benefit of kernel learning with the new kernels on two applications, one concerning the unsupervised clustering of rank data with kernel k-means, one focusing on the supervised classification of genomic data with Support Vector Machines (SVMs), reaching in both cases state-of-the-art performances.

The paper is organized as follows. In Section 2, we prove our main theorem showing that the Kendall and Mallows kernels are positive definite. We extend them to partial, multivariate and uncertain rankings respectively in Section 3, 4 and 5. We highlight the relation to the diffusion kernel of [43] in Section 6. Finally we illustrate the relevance of kernel methods for unsupervised (Section 7) and supervised (Section 8) tasks. Data and `R` codes for generating all the plots in this paper and reproducing more experiments are available via `https://github.com/YunlongJiao/kendallkernel_demo`.

2 The Kendall and Mallows kernels for permutations

Let us first fix some notations. Given a list of n items $\{x_1, x_2, \ldots, x_n\}$, a *total ranking* is a strict ordering on the n items of the form

$$x_{i_1} \succ x_{i_2} \succ \cdots \succ x_{i_n} \,, \tag{1}$$

where $\{i_1, \ldots, i_n\}$ are distinct indices in $\{1, 2, \ldots, n\} =: [1, n]$. A *permutation* is a 1-to-1 mapping from a finite set into itself, i.e., $\sigma : [1, n] \to [1, n]$ such that $\sigma(i) \neq \sigma(j)$ for $i \neq j$. Each total ranking can be equivalently represented by a permutation σ in the sense that $\sigma(i) = j$ indicates that a ranker assigns rank j to item i where higher rank coincides higher preference. For example, the ranking $x_2 \succ x_4 \succ x_3 \succ x_1$ is associated to the permutation $\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 4 & 2 & 3 \end{pmatrix}$, meaning $\sigma(1) = 1$, $\sigma(2) = 4$, $\sigma(3) = 2$ and $\sigma(4) = 3$. There are $n!$ different total rankings, and we denote by $\mathbb{S}_n$ the set of all permutations over n items. Endowed with the composition operation $(\sigma_1 \sigma_2)(i) = \sigma_1(\sigma_2(i))$, $\mathbb{S}_n$ is a group called the *symmetric group*.

Given two permutations $\sigma, \sigma' \in \mathbb{S}_n$, the number of concordant and discordant pairs between σ and σ' are respectively

$$n_c(\sigma, \sigma') = \sum_{i<j} \left[\mathbb{1}_{\{\sigma(i)<\sigma(j)\}} \mathbb{1}_{\{\sigma'(i)<\sigma'(j)\}} + \mathbb{1}_{\{\sigma(i)>\sigma(j)\}} \mathbb{1}_{\{\sigma'(i)>\sigma'(j)\}} \right],$$
$$n_d(\sigma, \sigma') = \sum_{i<j} \left[\mathbb{1}_{\{\sigma(i)<\sigma(j)\}} \mathbb{1}_{\{\sigma'(i)>\sigma'(j)\}} + \mathbb{1}_{\{\sigma(i)>\sigma(j)\}} \mathbb{1}_{\{\sigma'(i)<\sigma'(j)\}} \right].$$

As their names suggest, $n_c(\sigma, \sigma')$ and $n_d(\sigma, \sigma')$ count how many pairs of items are respectively in the same or opposite order in the two rankings σ and σ'. n_d is frequently used as a distance between permutations, often under the name *Kendall tau distance*, and underlies two popular similarity measures between permutations:

- The *Mallows kernel* defined for any $\lambda \geq 0$ by

$$K_M^{\lambda}(\sigma, \sigma') = e^{-\lambda n_d(\sigma, \sigma')} \,, \tag{2}$$

- The *Kendall kernel* defined as

$$K_\tau(\sigma,\sigma') = \frac{n_c(\sigma,\sigma') - n_d(\sigma,\sigma')}{\binom{n}{2}}\,. \tag{3}$$

The Mallows kernel plays a role on the symmetric group similar to the Gaussian kernel on Euclidean space, for example for statistical modeling of permutations [50, 15, 21, 53] or nonparametric smoothing [47], and the Kendall kernel [38, 39] is probably the most widely-used measure of rank correlation coefficient. In spite of their pervasiveness, to the best of our knowledge the following property has been overlooked:

Theorem 1. *The Mallows kernel K_M^λ, for any $\lambda \geq 0$, and the Kendall kernel K_τ are positive definite.*

Proof. Consider the mapping $\Phi : \mathbb{S}_n \to \mathbb{R}^{\binom{n}{2}}$ defined by

$$\Phi(\sigma) = \left(\frac{1}{\sqrt{\binom{n}{2}}}(\mathbb{1}_{\{\sigma(i)>\sigma(j)\}} - \mathbb{1}_{\{\sigma(i)<\sigma(j)\}})\right)_{1\leq i<j\leq n}\,.$$

Then one immediately sees that, for any $\sigma, \sigma' \in \mathbb{S}_n$,

$$K_\tau(\sigma,\sigma') = \Phi(\sigma)^\top \Phi(\sigma')\,,$$

showing that K_τ is positive definite, and that

$$\begin{aligned}\|\Phi(\sigma) - \Phi(\sigma')\|^2 &= K_\tau(\sigma,\sigma) + K_\tau(\sigma',\sigma') - 2K_\tau(\sigma,\sigma') \\ &= 1 + 1 - 2\Big(\frac{n_c(\sigma,\sigma') - n_d(\sigma,\sigma')}{\binom{n}{2}}\Big) \\ &= \frac{4}{\binom{n}{2}} n_d(\sigma,\sigma')\,,\end{aligned} \tag{4}$$

showing that n_d is conditionally positive definite and therefore that K_M^λ is positive definite for all $\lambda \geq 0$ [59]. □

Although the Kendall and Mallows kernels correspond respectively to a linear and Gaussian kernel on an $\binom{n}{2}$-dimensional embedding of $\mathbb{S}_n$ such that they can in particular be computed in $O(n^2)$ time by a naive implementation of pair-by-pair comparison, it is interesting to notice that more efficient algorithms based on divide-and-conquer strategy can significantly speed up the computation, up to $O(n \log n)$ using a technique based on Merge Sort algorithm [40]. Computing in $O(n \log n)$ a kernel corresponding to an $O(n^2)$-dimensional embedding of $\mathbb{S}_n$ is a typical example of the kernel trick, which allows to scale kernel methods to larger values of n than what would be possible for methods working with the explicit embedding.

3 Extension to partial rankings

In this section we show how the Kendall and Mallows kernels can efficiently be adapted to partial rankings, a situation frequently encountered in practice. For example, in a movie recommender system, each user only grades a few movies that he has watched based on personal interest. As another example, in a chess tournament, each game results in a relative ordering between two contestants, and one would typically like to find a single ranking of all players that globally best represents the large collection of binary outcomes.

As opposed to a total ranking (1), *partial rankings* arise in diverse form which can be generally described by

$$X_1 \succ X_2 \succ \cdots \succ X_k,$$

where $X_1, \ldots, X_k$ are k disjoint subsets of n items $\{x_1, \ldots, x_n\}$. For example, $\{x_2, x_4\} \succ x_6 \succ \{x_3, x_8\}$ in a social survey could represent the fact that items 2 and 4 are ranked higher by an interviewee than item 6, which itself is ranked higher than items 3 and 8. Note that it is uninformative of the relative order between items 2 and 4, nor of how item 1 is rated. For ease of analysis, a partial ranking is often associated with a subset $R \subset \mathbb{S}_n$ of permutations which are compatible with all partial orders described by the partial ranking. In this study, two particularly interesting types are:

(i) Interleaving partial rankings. Such a partial ranking is of the form

$$x_{i_1} \succ x_{i_2} \succ \cdots \succ x_{i_k}, \quad k \leq n,$$

where we have a total ranking for k out of n items. This type of partial ranking is frequently encountered in real life, for example in a social survey an interviewer is inexperienced to rank all items listed so that there exist interleaved inaccessible values. The interleaving partial ranking corresponds to the set of permutations compatible with it:

$$A_{i_1,\ldots,i_k} = \{\sigma \in \mathbb{S}_n | \sigma(i_a) > \sigma(i_b) \text{ if } a < b, a, b \in [1, k]\}. \tag{5}$$

(ii) Top-k partial rankings. Such a partial ranking is of the form

$$x_{i_1} \succ x_{i_2} \succ \cdots \succ x_{i_k} \succ X_{\text{rest}}, \quad k \leq n,$$

where we have a total ranking for k out of n items and also know that these k items are ranked higher than all the other items. For example, the top k hits returned by a search engine leads to a top k partial ranking; under a voting system in election, voters express their vote by ranking some (or all) of the candidates in order of preference. The top-k partial ranking corresponds to the set of compatible permutations:

$$B_{i_1,\ldots,i_k} = \{\sigma \in \mathbb{S}_n | \sigma(i_a) = n + 1 - a, a \in [1, k]\}. \tag{6}$$

To extend any kernel K over $\mathbb{S}_n$ to a kernel over the set of partial rankings, we propose to represent a partial ranking by its compatible subset $R \subset \mathbb{S}_n$ of permutations, and define a kernel between two partial rankings R and R' by adopting the *convolution kernel*, written with a slight abuse of notations as

$$K(R, R') = \frac{1}{|R||R'|} \sum_{\sigma \in R} \sum_{\sigma' \in R'} K(\sigma, \sigma'). \tag{7}$$

As a convolution kernel, it is positive definite as long as K is positive definite [30]. However, a naive implementation to compute (7) typically requires $O((n-k)!(n-k')!)$ operations when the number of observed items in partial rankings R, R' is respectively $k, k' < n$, which can quickly become prohibitive. Fortunately Theorem 2 guarantees that we can circumvent the computational burden of naively implementing (7) with the Kendall kernel K_τ on at least the two particular cases of partial rankings (5) or (6).

Theorem 2. *The Kendall kernel K_τ between two interleaving partial rankings of respectively k and m observed items, or between a top-k partial ranking and a top-m partial ranking, of form (7) can be computed in $O(k \log k + m \log m)$ operations.*

Proof and explicit algorithms are postponed to the appendices. Note that the convolution kernel (7) taking the Mallows kernel K_M^λ is not straightforward to evaluate, which will be further discussed in Section 6. However, since we have extended the Kendall kernel to partial rankings, an exponential kernel can be constructed trivially following (4), for which the computation remains just as simple as the extended Kendall kernel. Since this technique also applies in following sections, we focus mainly on extending Kendall kernel henceforth.

4 Extension to multivariate rankings

In contrast to the rankings defined in previous sections, a *multivariate ranking* can be seen as a collection of multiple (univariate) partial/total rankings from the same ranker based on different sources. For example, a commercial survey is designed to analyze the preference routines of a customer based on various categories such as music, movies and novels, where the item sets are generally incomparable in crossing categories; an electoral system asks a voter to express his opinion in consecutive sessions across years, where the candidates are usually different across elections. In that case, it is desirable to process and integrate the rank data from different sources when extensively comparing the similarity between two rankers. Known as "data fusion", this problem is well studied in the kernel learning literature [46, 63].

Let us now denote that a ranker is represented by a multivariate ranking $\mathbf{R} = (R_1, \ldots, R_p)$, in which each component R_j for $1 \leq j \leq p$ is a partial ranking over n_j items, i.e., a subset of permutations (or exactly one permutation when all n_j items are totally ranked) in $\mathbb{S}_{n_j}$. Suppose K is any kernel for univariate rankings, a kernel for multivariate rankings that integrates information from several variates can be constructed by a weighted average of the kernels evaluated individually for each variate, written with a slight abuse of notations as

$$K(\mathbf{R}, \mathbf{R}') = \sum_{j=1}^{p} \mu_j K(R_j, R'_j) \quad \text{s.t.} \quad \sum_{j=1}^{p} \mu_j = 1\,, \tag{8}$$

where a kernel K for partial rankings has been defined in (7). A practically simple approach would be to set the weights $\mu_j = 1/p$ for $1 \leq j \leq p$ in (8), but the weights can be learned as well through multiple kernel learning under appropriate setting [45, 4, 69, 26].

5 Extension to uncertain rankings

When data to analyze are n-dimensional real-valued quantitative vectors, converting them to permutations in $\mathbb{S}_n$ by ranking their entries can be beneficial in cases where we trust more the relative ordering of the values than their absolute magnitudes. For example in social surveys or recommender systems, users are sometimes asked to rate a score for each item individually instead of providing a preference order on the item set. The scale of ratings usually varies according to personal preference of each user and it can therefore be safer to adopt ranking-based methods to analyze such score-based rating data [35]. As another example, an interesting line of work in the analysis of gene expression data promotes the development of classifiers based on relative gene expression within a sample, based on the observations that gene expression measurements are subject to various measurement errors such as technological biases and normalization issues, while assessing whether a gene is more expressed than another gene is generally a more robust task [24, 71, 79, 49]. This suggests that the Kendall kernel can be relevant for analyzing quantitative vectors.

The Kendall kernel for quantitative vectors now takes exactly the same form as for permutations, i.e.,

$$K_\tau(\mathbf{x}, \mathbf{x}') = \Phi(\mathbf{x})^\top \Phi(\mathbf{x}')\,, \tag{9}$$

where $\Phi : \mathbb{R}^n \to \mathbb{R}^{\binom{n}{2}}$ is defined for $\mathbf{x} = (x_1, \ldots, x_n)^\top \in \mathbb{R}^n$ by

$$\Phi(\mathbf{x}) = \left(\frac{1}{\sqrt{\binom{n}{2}}} (\mathbb{1}_{\{x_i > x_j\}} - \mathbb{1}_{\{x_i < x_j\}}) \right)_{1 \le i < j \le n}. \tag{10}$$

In this case, the interpretation of the Kendall kernel in terms of concordant and discordant pairs (3) is still valid, with the caveats that in the presence of ties between entries of $\mathbf{x}$, say two coordinates i and j such that $x_i = x_j$, the tied pair $\{x_i, x_j\}$ will be neither concordant nor discordant. This implies in particular that if $\mathbf{x}$ has ties or so does $\mathbf{x}'$, then $|K_\tau(\mathbf{x}, \mathbf{x}')| < 1$ strictly. Notably in the presence of ties, the fast implementation of Kendall kernel still applies to quantitative vectors in $O(n \log n)$ time [40]. However, feature mapping (10) is by construction very sensitive to the presence of entry pairs that are ties or almost-ties. In fact, each entry of $\Phi(\mathbf{x})$ is, up to a normalization constant, the Heaviside step function which takes discrete values in $\{-1, 0, +1\}$, and thus can change abruptly even when $\mathbf{x}$ changes slightly but reverses the ordering of two entries whose values are close. In addition to pairwise relative ordering as defined in (10), it can be wise to also exploit the information given by pairwise absolute difference in the feature values.

We propose to make the mapping more robust by assuming a random noise $\epsilon \sim \mathcal{P}$ added to the feature vector $\mathbf{x}$ and checking where $\Phi(\mathbf{x} + \epsilon)$ is on average (similarly to, e.g., [55]). In other words, we consider a smoother mapping $\Psi : \mathbb{R}^n \to \mathbb{R}^{\binom{n}{2}}$ defined by

$$\Psi(\mathbf{x}) = \mathbb{E}\Phi(\mathbf{x} + \epsilon) =: \mathbb{E}\Phi(\tilde{\mathbf{x}}), \tag{11}$$

where ϵ is an n-dimensional random noise and $\tilde{\mathbf{x}} := \mathbf{x} + \epsilon$ denotes the random-jittered vector of $\mathbf{x}$. The corresponding kernel is the underlying dot product as usual:

$$G\,(\mathbf{x}, \mathbf{x}') = \Psi(\mathbf{x})^\top \Psi(\mathbf{x}') = \mathbb{E}\Phi(\tilde{\mathbf{x}})^\top \mathbb{E}\Phi(\tilde{\mathbf{x}}') = \mathbb{E}K_\tau(\tilde{\mathbf{x}}, \tilde{\mathbf{x}}')\,, \tag{12}$$

where $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{x}}'$ are independently noise-perturbed versions of $\mathbf{x}$ and $\mathbf{x}'$. In fact, we can deduce from (10) that Ψ is equivalently written as

$$\Psi(\mathbf{x}) = \left(\frac{1}{\sqrt{\binom{n}{2}}} (\mathbb{P}(\tilde{x}_i > \tilde{x}_j) - \mathbb{P}(\tilde{x}_i < \tilde{x}_j)) \right)_{1 \le i < j \le n} .$$

Depending on the noise distribution, various kernels are thus obtained. For example, assuming specifically that $\epsilon \sim (\mathcal{U}[-\frac{a}{2}, \frac{a}{2}])^n$ the n-dimensional uniform noise of window size a centered at 0, the (i,j)-th entry of $\Psi(\mathbf{x})$ for all $i < j$ becomes

$$\Psi_{ij}(\mathbf{x}) = \frac{1}{\sqrt{\binom{n}{2}}} g_a(x_i - x_j) , \tag{13}$$

where

$$g_a(t) := \begin{cases} 1 & t \ge a \\ 2(\frac{t}{a}) - (\frac{t}{a})^2 & 0 \le t \le a \\ 2(\frac{t}{a}) + (\frac{t}{a})^2 & -a \le t \le 0 \\ -1 & t \le -a \end{cases} .$$

Note that g_a is odd, continuous, piecewise quadratic between $[-a, a]$ and constant elsewhere at ± 1, and thus can be viewed as smoothed version of the Heaviside step function to compare any two entries x_i and x_j from their difference $x_i - x_j$ (Figure 1).

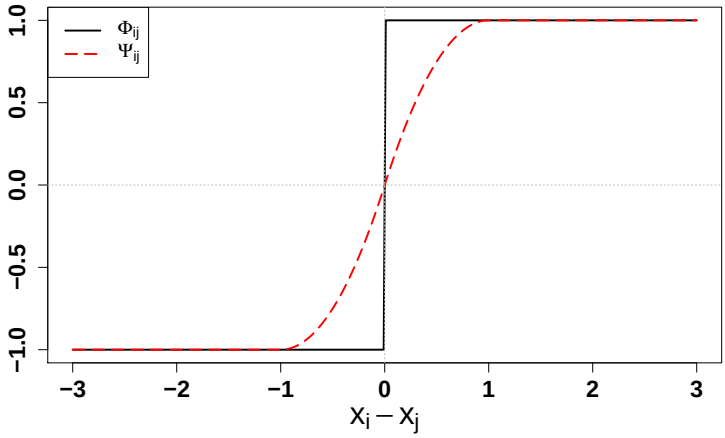


Figure 1: Smooth approximation (in red) of the Heaviside function (in black) used to define the mapping (13) for $a = 1$.

Although the smoothed kernel (12) can be an interesting alternative to the Kendall kernel (9), we unfortunately lose for G the computational trick that allows to compute K_τ in $O(n \log n)$. Specifically, we have two ways to compute G:

(i) Exact evaluation. The first alternative is to compute explicitly the $\binom{n}{2}$-vector representation Ψ in the feature space, and then take the dot product to obtain G. While the kernel evaluation is exact, an analytic form of the smoothed mapping (11) is required and the computational cost is linear with the dimension of the feature space, i.e., $O(n^2)$.

(ii) Monte Carlo approximation. The second alternative requires the observation that the smoothed mapping $\Psi(\mathbf{x}) = \mathbb{E}\Phi(\tilde{\mathbf{x}})$ appears in the form of expectation and can thus be approximated by a D-sample mean of jittered points mapped by Φ into the feature space:

$$\Psi_D(\mathbf{x}) = \frac{1}{D}\sum_{j=1}^{D} \Phi(\tilde{\mathbf{x}}^j)\,,$$

where $\tilde{\mathbf{x}}^1, \ldots, \tilde{\mathbf{x}}^D$ are i.i.d. noisy versions of $\mathbf{x}$. The dot product induces a kernel:

$$G_D(\mathbf{x}, \mathbf{x}') = \Psi_D(\mathbf{x})^\top \Psi_D(\mathbf{x}') = \frac{1}{D^2}\sum_{i=1}^{D}\sum_{j=1}^{D} K_\tau\left(\tilde{\mathbf{x}}^i, \tilde{\mathbf{x}}'^j\right)\,, \tag{14}$$

which is a D^2-sample empirical estimate of $G(\mathbf{x}, \mathbf{x}') = \mathbb{E}K_\tau(\tilde{\mathbf{x}}, \tilde{\mathbf{x}}')$ when $\mathbf{x}, \mathbf{x}'$ are independently jittered with identically distributed noise. Since K_τ is of computational complexity $O(n \log n)$, computing G_D requires $O(D^2 n \log n)$.

Note that the second alternative is faster to compute than the first one as long as, up to constants, $D^2 < n/\log n$, and small values of D are thus favored on account of computational consideration. In that case, however, the approximation performance can be unappealing. To better understand the trade-off between the two alternatives, the question should be addressed upon how large D should be so that the approximation error is not detrimental to the performance of a learning algorithm if we use the approximate kernel G_D instead of G. Lemma 1 provides a first answer to this question, showing that the approximation error of the kernel is upper bounded by $O(1/\sqrt{D})$ with high probability:

Lemma 1. *For any $0 < \delta < 1$, the following holds:*
(a) For any $\mathbf{x} \in \mathbb{R}^n$, with probability greater than $1 - \delta$,

$$\|\Psi_D(\mathbf{x}) - \Psi(\mathbf{x})\| \leq \frac{1}{\sqrt{D}}\left(2 + \sqrt{8 \log \frac{1}{\delta}}\right)\,.$$

(b) For any $\mathbf{x}_1, \ldots, \mathbf{x}_m \in \mathbb{R}^n$, with probability greater than $1 - \delta$,

$$\sup_{i=1,\ldots,m} \|\Psi_D(\mathbf{x}_i) - \Psi(\mathbf{x}_i)\| \leq \frac{1}{\sqrt{D}}\left(2 + \sqrt{8 \log \frac{m}{\delta}}\right)\,.$$

Proof is referred to the appendices. The uniform approximation bound of Lemma 1 in turn implies that learning with the approximate kernel G_D can be almost as good with the kernel G, as we now discuss. For that purpose, let us consider for example the case where the smoothed kernel G is used to train a Support Vector Machine (SVM) from a training set $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^m \subset (\mathbb{R}^n \times \{-1, +1\})^m$, specifically to estimate a function $h(\mathbf{x}) = \mathbf{w}^\top \Psi(\mathbf{x})$ by solving

$$\min_{\mathbf{w}} F(\mathbf{w}) = \frac{\lambda}{2}\|\mathbf{w}\|^2 + \widehat{R}(\mathbf{w}), \tag{15}$$

where $\widehat{R}(\mathbf{w}) = \frac{1}{m}\sum_{i=1}^m \ell(y_i \mathbf{w}^\top \Psi(\mathbf{x}_i))$ is the empirical loss, with $\ell(y_i \mathbf{w}^\top \Psi(\mathbf{x}_i)) = \max(0, 1 - y_i \mathbf{w}^\top \Psi(\mathbf{x}_i))$ the hinge loss associated to the i-th point, λ the regularization parameter. Now

suppose that instead of training the SVM with smoothed feature mapping on the original points $\{\Psi(\mathbf{x}_i)\}_{i=1,\dots,m}$, we first randomly jitter $\{\mathbf{x}_i\}_{i=1,\dots,m}$ D times at each point, resulting in $\{\tilde{\mathbf{x}}_i^j\}_{i=1,\dots,m;j=1,\dots,D}$, and then replace each $\Psi(\mathbf{x}_i)$ by the D-sample empirical average of jittered points mapped by Φ into the feature space, that is

$$\Psi_D(\mathbf{x}_i) := \frac{1}{D}\sum_{j=1}^{D}\Phi(\tilde{\mathbf{x}}_i^j)\,.$$

Note that $\Psi_D(\mathbf{x}_i)^\top\Psi_D(\mathbf{x}_j) = G_D(\mathbf{x}_i,\mathbf{x}_j)$, hence training an SVM with the Monte Carlo approximate G_D instead of exact version G is equivalent to solving (15) with $\{\Psi_D(\mathbf{x}_i)\}_{i=1,\dots,m}$ in the hinge loss instead of $\{\Psi(\mathbf{x}_i)\}_{i=1,\dots,m}$. Theorem 3 quantifies the approximation performance in terms of objective function F which helps to answer the question on the trade-off between G and G_D in computational complexity and learning accuracy.

Theorem 3. *For any $0 \le \delta \le 1$, the solution $\widehat{\mathbf{w}}_D$ of the SVM trained with the Monte Carlo approximation (14) with D random-jittered samples for each training point satisfies, with probability greater than $1-\delta$,*

$$F(\widehat{\mathbf{w}}_D) \le \min_{\mathbf{w}} F(\mathbf{w}) + \sqrt{\frac{8}{\lambda D}}\left(2 + \sqrt{8\log\frac{m}{\delta}}\right)\,.$$

Proof is referred to the appendices. It is known that compared to the exact solution of (15), an $O(m^{-1/2})$-approximate solution is sufficient to reach the optimal statistical accuracy [9]. This accuracy can be attained in our analysis when $D = O(m/\lambda)$, and since typically $\lambda \sim m^{-1/2}$ [70], this suggests that it is sufficient to take D of order $m^{3/2}$. Going back to the comparison strategy of the two alternatives G and G_D, we see that the computational cost of computing the full $m \times m$ Gram matrix with the exact evaluation is $O(m^2n^2)$, while the cost of computing the approximate Gram matrix with $D = O(m^{3/2})$ random samples is $O(m^2D^2n\log n) = O(m^5 n\log n)$. This shows that, up to constants and logarithmic terms, the Monte Carlo approximation is interesting when $m = o(n^{1/3})$, otherwise the exact evaluation using explicit computation in the feature space is preferable.

Interestingly we can look at the extended Kendall kernel (12) to uncertain rankings from the perspective of Hilbert space embeddings of probability distributions [68]. In fact, for x fixed, the smoothed mapping $\Psi(\mathbf{x}) = \mathbb{E}\Phi(\mathbf{x}+\epsilon)$ is exactly an embedding for the distribution $\mathcal{P}$ of an additive noise ϵ in the reproducing kernel Hilbert space (RKHS) associated with Kendall kernel. As a consequence, the idea of smoothed kernel $G(\mathbf{x},\mathbf{x}')$ for $\mathbf{x},\mathbf{x} \in \mathcal{X}$ is essentially equivalent to how [55, Lemma 4] defines kernels on two probability distributions from $\{\mathcal{P}+\mathbf{x}|\mathbf{x}\in\mathcal{X}\}$ using the Kendall kernel as the level-1 embedding kernel and linear inner product as the level-2 kernel in the feature space. As a result, given a fixed training set $\mathcal{D}$, training an SVM with G in place of K_τ is equivalent to training a Flex-SVM instead of an ordinary SVM with K_τ [55]. In this case, Theorem 3 provides an error bound in terms of the optimal accuracy for cases when training a Flex-SVM if exact evaluation of G is intractable and its Monte Carlo approximate G_D is employed. This serves to obtain a trade-off between computation complexity and approximation accuracy which is particularly interesting when we are working in high dimensions.

6 Relation to the diffusion kernel on $\mathbb{S}_n$

It is interesting to relate the Mallows kernel (2) to the diffusion kernel on the symmetric group proposed by [43], which is the diffusion kernel [42] on the Cayley graph of $\mathbb{S}_n$ generated by adjacent transpositions with left-multiplication. This graph, illustrated for a specific case of $n = 4$ in Figure 2, is defined by $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $\mathcal{V} = \mathbb{S}_n$ as vertices, and undirected edge set $\mathcal{E} = \big\{\{\sigma, \pi\sigma\} : \sigma \in \mathbb{S}_n, \pi \in Q\big\}$, where $Q = \{(i, i+1)|i = 1, \ldots, n-1\}$ the set of all adjacent transpositions. Note Q is symmetric in the sense that $\pi \in Q \Leftrightarrow \pi^{-1} \in Q$, and the graph adjacency relation is a right-invariant relation, that is $\sigma \sim \sigma' \Leftrightarrow \sigma'\sigma^{-1} \in Q$. The corresponding graph Laplacian is the matrix Δ with

$$\Delta_{\sigma,\sigma'} = \begin{cases} 1 & \text{if } \sigma \sim \sigma' \\ -(n-1) & \text{if } \sigma = \sigma' \\ 0 & \text{otherwise} \end{cases},$$

where $n-1$ is the degree of vertex σ (number of edges connected with vertex σ), and the *diffusion kernel* on $\mathbb{S}_n$ is finally defined as

$$K^{\beta}_{\text{dif}}(\sigma, \sigma') = [e^{\beta\Delta}]_{\sigma,\sigma'} \tag{16}$$

for some diffusion parameter $\beta \in \mathbb{R}$, where $e^{\beta\Delta}$ is the matrix exponential. K^{β}_{dif} is a right-invariant kernel on the symmetric group [43, Proposition 2], and we denote by $\kappa^{\beta}_{\text{dif}}$ the positive definite function induced by K^{β}_{dif} such that $K^{\beta}_{\text{dif}}(\sigma, \sigma') = \kappa^{\beta}_{\text{dif}}(\sigma'\sigma^{-1})$. Since the Mallows kernel K^{λ}_{M} is straightforwardly right-invariant, we denote by κ^{λ}_{M} the positive definite function induced by the Mallows kernel K^{λ}_{M} such that $K^{\lambda}_{M}(\sigma, \sigma') = \kappa^{\lambda}_{M}(\sigma'\sigma^{-1})$. One way to interpret the diffusion kernel (16) is by the heat equation on the Cayley graph

$$\frac{d}{d\beta} K^{\beta}_{\text{dif}} = \Delta K^{\beta}_{\text{dif}} \quad \text{s.t. } K^{\beta}_{\text{dif}}|_{\beta=0} = I.$$

K^{β}_{dif} is thus the product of a continuous process, expressed by the graph Laplacian Δ, gradually transforming local structure $K^{\beta}_{\text{dif}}|_{\beta=0} = I$ to a kernel with stronger and stronger off-diagonal effects as β increases.

Interestingly, the Mallows kernel can also be interpreted with the help of the Cayley graph. Indeed, it is well-known that the Kendall tau distance $n_d(\sigma, \sigma')$ is the minimum number of adjacent swaps required to bring σ to σ', i.e. $n_d(\sigma, \sigma')$ equals to the shortest path distance on the Cayley graph, or simply written

$$n_d(\sigma, \sigma') = d_{\mathcal{G}}(\sigma, \sigma')\,.$$

Different from the diffusion kernel for which communication between permutations is a diffusion process over the graph, the Mallows kernel

$$K^{\lambda}_{M}(\sigma, \sigma') = e^{-\lambda n_d(\sigma,\sigma')} = e^{-\lambda d_{\mathcal{G}}(\sigma,\sigma')}$$

considers exclusively the shortest path over the graph when expressing the similarity between permutations σ, σ'.

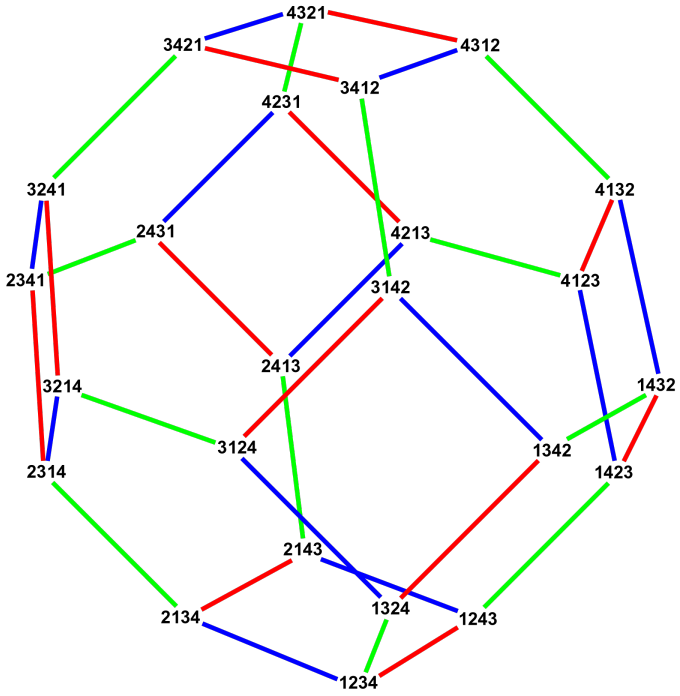


Figure 2: Cayley graph of $\mathbb{S}_4$, generated by the transpositions (1 2) in blue, (2 3) in green, and (3 4) in red.

A notable advantage of the Mallows kernel over the diffusion kernel is that the Mallows kernel enjoys faster evaluation. On one hand if data instances are total rankings, i.e. $\sigma, \sigma' \in \mathbb{S}_n$, evaluating $K^{\beta}_{\text{dif}}(\sigma, \sigma')$ would require exponentiating an $n!$-dimensional Laplacian matrix by naive implementation, and can reduce to exponentiating matrices of smaller sizes by careful analysis in the Fourier space, which still remains problematic if working dimension n is large [43]. However, evaluating $K^{\lambda}_{M}(\sigma, \sigma')$ only takes $O(n \log n)$ time. On the other hand if data instances are partial ranking of size $k \ll n$, i.e. $R, R' \subset \mathbb{S}_n$, and we take convolution kernel (7) to extend the two kernels, the analysis of exploring the sparsity of the Fourier coefficients of the group algebra of partial rankings R, R' of size k reduces the evaluation of both the diffusion kernel and the Mallows kernel to $O((2k)^{2k+3})$ time, provided that the exponential kernel Fourier matrices $[\hat{\overline{\kappa}}(\mu)]_{\geq [\dots]_{n-k}}$ are precomputed before any kernel evaluations take place [43, Theorem 13].

7 Application: Clustering and modeling rank data

In this section we illustrate the potential benefit of kernel-based algorithms using the Kendall and Mallows kernels for the purpose of unsupervised cluster analysis, i.e., partitioning a collection of rank data into sub-groups and/or estimating densities of a collection of rank data. This is in particular of great practical interest in social choice theory in order to

explore the heterogeneity and identify typical sub-groups of voters with a common behavior to understand, for example, their political support for various parties [28, 29, 52].

7.1 Clustering with kernel k-means

Let $\{\sigma_i\}_{i=1}^m \subset \mathbb{S}_n$ be a collection of m permutations representing, say, the preferences of customers over n products or the votes of electorate over n candidates. We aim at partitioning these permutations into $c \leq m$ clusters $\{S_j\}_{j=1}^c$. One approach to cluster rank data is to follow a method similar to k-means in the symmetric group. Assuming that each cluster S_j has a "center" $\pi_j \in \mathbb{S}_n$ serving as a prototype permutation of that cluster, the classic k-means clustering attempts to put each point in the cluster with the nearest center so as to minimize the sum of Kendall tau distance of each permutation to the corresponding center of its cluster. Specifically, when the number of clusters c is fixed, the objective is to find:

$$\underset{\{S_j,\pi_j\in\mathbb{S}_n\}}{\arg\min} \sum_{j=1}^{c} \sum_{i:\sigma_i\in S_j} n_d(\sigma_i,\pi_j)\,. \tag{17}$$

Note that (17) reduces to a single-ranking aggregation problem when $c = 1$, where the center π is commonly known as Kemeny consensus [37] which is NP-hard to find [7]. With the objective in (17) being non convex, Lloyd's algorithm is usually employed to find local minima in an iterative manner consisting of two steps: the *assignment step* assigns each point to its closest cluster, and the *update step* updates each of the c cluster centers using the points assigned to that cluster; the algorithm repeats until all the cluster centers remain unchanged in an iteration. While the assignment step is usually fast, the update step is indeed equivalent to solving a Kemeny consensus problem for each cluster, i.e., $\arg\min_{\pi_j\in\mathbb{S}_n} \sum_{i:\sigma_i\in S_j} n_d(\sigma_i,\pi_j)$. Since the exact Kemeny-optimal ranking is difficult to find, approximate techniques are usually employed in practice such as Borda Count [16] or Copeland's method [11].

As the Kendall tau distance is conditionally positive definite, we can propose as an alternative to use the kernel k-means approach [25, 80] that relaxes the assumption that the cluster center are permutations, and instead works implicitly in the feature space where cluster centers can be any vector in $\mathbb{R}^{\binom{n}{2}}$ by considering the problem:

$$\underset{\left\{S_j,\mu_j\in\mathbb{R}^{\binom{n}{2}}\right\}}{\arg\min} \sum_{j=1}^{c} \sum_{i:\sigma_i\in S_j} \left\|\Phi(\sigma_i)-\mu_j\right\|^2\,,$$

for which local minima can be found efficiently by Algorithm 1. Note that μ_j does not match a true permutation $\pi_j \in \mathbb{S}_n$ in general, and the Kemeny consensus problem in the update step is thus bypassed. It is worthwhile to note that the algorithm is not exclusive for clustering permutations, kernel k-means clustering can be applied respectively to total/partial/multivariate/uncertain rankings with appropriate kernels defined.

7.2 Mallows mixture model with kernel trick

An alternative to k-means clustering is to consider mixture models, which provide a method for modeling heterogeneous population in data by assuming a mixture of standard models

Algorithm 1 Kernel k-means for clustering heterogeneous rank data.

Input: a collection of permutations $\{\sigma_i\}_{i=1}^m$ and a kernel function K over $\mathbb{S}_n$, or a kernel matrix evaluated between pairwise data points $\mathbf{K} = (K(\sigma_i, \sigma_j))_{1 \le i,j \le m}$; the number of clusters $c \le m$.

1: Randomly initialize cluster assignment for each data points and form c clusters $S_1, \ldots, S_c$.

2: For each data point, find its new cluster assignment, i.e., for $i = 1, \ldots, m$,

$$j^*(\sigma_i) = \arg\min_j d_{ij} \,,$$

where

$$d_{ij} := \left\| \Phi(\sigma_i) - \frac{1}{|S_j|} \sum_{\sigma_\ell \in S_j} \Phi(\sigma_\ell) \right\|^2 = K(\sigma_i, \sigma_i) - \frac{2}{|S_j|} \sum_{\sigma_\ell \in S_j} K(\sigma_i, \sigma_\ell) + \frac{1}{|S_j|^2} \sum_{\sigma_v, \sigma_\ell \in S_j} K(\sigma_v, \sigma_\ell) \,.$$

3: Form updated clusters, i.e., for $j = 1, \ldots, c$,

$$S_j = \{\sigma_i : j = j^*(\sigma_i), i = 1, \ldots, m\} \,.$$

4: Repeat 2-3 until all cluster assignments remain unchanged in an iteration.

Output: Cluster assignments $\{S_j\}_{j=1}^c$.

for rankings in each homogeneous sub-population. Mixture models not only allow to cluster data, but more generally to estimate a distribution on the space of permutation that can then be used for other purposes, such as combining evidences. One popular choice of probabilistic distribution over $\mathbb{S}_n$ is the Mallows model [50], which takes the form in expressing the occurring probability of σ by

$$f(\sigma|\pi,\lambda) = C(\lambda)\exp[-\lambda n_d(\sigma,\pi)]\,, \tag{18}$$

where the central ranking $\pi \in \mathbb{S}_n$ and the precision $\lambda \geq 0$ are model parameters, and the normalization constant $C(\lambda) = 1/\sum_{\sigma'\in\mathbb{S}_n}\exp[-\lambda n_d(\sigma',\pi)]$ is chosen so that $f(\cdot|\pi,\lambda)$ is a valid probability mass function over $\mathbb{S}_n$. Notably, $C(\lambda)$ does not depend on the center π due to the symmetry of $\mathbb{S}_n$.

We follow the mixture modeling setup in [56]. Now suppose that a population consists of c sub-populations, a Mallows mixture model assumes that an observation comes from group j with probability $p_j \geq 0$ for $j = 1,\ldots,c$ and, given that the observation belongs to sub-population j, it is generated from a Mallows model with central ranking π_j and precision λ_j, i.e., the occurring probability of σ in the Mallows mixture model is written as

$$f(\sigma) = \sum_{j=1}^{c} p_j f(\sigma|\pi_j,\lambda_j) = \sum_{j=1}^{c} p_j C(\lambda_j)\exp[-\lambda_j n_d(\sigma,\pi_j)]\,. \tag{19}$$

Denoting $\underline{\pi} = \{\pi_j\}_{j=1}^c$, $\underline{\lambda} = \{\lambda_j\}_{j=1}^c$, $\underline{p} = \{p_j\}_{j=1}^c$ such that $\sum_{j=1}^c p_j = 1$, the log-likelihood of a collection of m i.i.d. permutations $\underline{\sigma} = \{\sigma_i\}_{i=1}^m$ is therefore:

$$L(\underline{\pi},\underline{\lambda},\underline{p}|\underline{\sigma}) = \sum_{i=1}^{m}\log f(\sigma_i) = \sum_{i=1}^{m}\log\left\{\sum_{j=1}^{c} p_j C(\lambda_j)\exp[-\lambda_j n_d(\sigma_i,\pi_j)]\right\}. \tag{20}$$

The Mallows mixture model is usually fitted by maximum likelihood using the EM algorithm. Specifically, by introducing latent (membership) variables $\underline{z} = \{z_{ij} : i = 1,\ldots,m, j = 1,\ldots,c\}$ where $z_{ij} = 1$ if σ_i belongs to group j and 0 otherwise, the complete log-likelihood of data is

$$L_C(\underline{\pi},\underline{\lambda},\underline{p}|\underline{\sigma},\underline{z}) = \sum_{i=1}^{m}\sum_{j=1}^{c} z_{ij}[\log p_j + \log C(\lambda_j) - \lambda_j n_d(\sigma_i,\pi_j)]\,.$$

The EM algorithm can be implemented to find local maximum likelihood estimates following two steps iteratively until convergence: the *E-step* calculates the expected value of membership variables $\hat{\underline{z}}$ conditioned on the current estimates of the model parameters $\underline{\pi},\underline{\lambda},\underline{p}$, and the *M-step* updates the model parameters $\underline{\pi},\underline{\lambda},\underline{p}$ by maximizing the expected complete log-likelihood $\hat{L}_C = L_C(\underline{\pi},\underline{\lambda},\underline{p}|\underline{\sigma},\hat{\underline{z}})$ where membership variables are replaced by their expected values. The final estimate $\hat{z}_{ij}$ amounts to our belief of σ_i belonging to group j, and can thus be used to form clusters $\{S_j\}_{j=1}^c$ serving a partition of data where

$$S_j = \left\{\sigma_i : \hat{z}_{ij} = \max_{\ell}\hat{z}_{i\ell}, i = 1,\ldots,m\right\}. \tag{21}$$

A closer look at the EM algorithm reveals that optimizing $\hat{L}_C$ with respect to $\underline{\pi}$ alone in the M-step is indeed equivalent to finding a (weighted) Kemeny consensus for each group,

i.e., solving $\arg\min_{\pi_j \in \mathbb{S}_n} \sum_{i=1}^{m} \hat{z}_{ij} n_d(\sigma_i, \pi_j)$, for which exact solution is difficult as above-mentioned in the context of k-means clustering. Similarly to the idea of kernel k-means in contrast to classic k-means, we propose to seek ways to bypass the Kemeny consensus problem by working in the feature space instead. Note that the Mallows probability mass function (18) is equivalently written as $f(\sigma|\pi, \lambda) \propto \exp\left[-\lambda \left\|\Phi(\sigma) - \Phi(\pi)\right\|^2\right]$ up to a constant scaling on λ by using (4), we propose to relax the constraint that the center has to match a true permutation $\pi \in \mathbb{S}_n$ and consider the following two alternatives in place of f following the mixture modeling approach stated above:

(i) Kernel Mallows. The Mallows probability mass function over $\mathbb{S}_n$ (18) is generalized to admit any point in the feature space $\mu \in \mathbb{R}^{\binom{n}{2}}$ to be the population center, i.e.,

$$g(\sigma|\mu, \lambda) = C(\mu, \lambda) \exp\left[-\lambda \left\|\Phi(\sigma) - \mu\right\|^2\right], \tag{22}$$

where the normalization constant $C(\mu, \lambda) = 1/\sum_{\sigma' \in \mathbb{S}_n} \exp\left[-\lambda \left\|\Phi(\sigma') - \mu\right\|^2\right]$ is chosen so that $g(\cdot|\mu, \lambda)$ is a valid probability mass function over $\mathbb{S}_n$. Notably, $C(\mu, \lambda)$ now depends on the center μ as well.

If we replace the probability mass function of classic Mallows f in (20) by that of kernel Mallows g, the Kemeny consensus problem is averted when the EM algorithm is used to fit a local maximum likelihood estimate. However, another computational setback arises that the expected complete log-likelihood $\hat{L}_C$ to maximize in the M-step of the EM algorithm is separately concave with respect to μ or λ, but not jointly concave. Hence alternating optimization is often used in practice with the caveats of intensive computation and no guarantee to attain global optima for the M-step optimization at each iteration.

(ii) Kernel Gaussian. Note that (22) has a similar form to the Gaussian density, therefore we consider for $\sigma \in \mathbb{S}_n$,

$$g^\dagger(\sigma|\mu, \lambda) = \sqrt{\left(\frac{\lambda}{\pi}\right)^{\binom{n}{2}}} \exp\left[-\lambda \left\|\Phi(\sigma) - \mu\right\|^2\right], \tag{23}$$

which is exactly $\mathcal{N}(\Phi(\sigma)|\mu, (2\lambda)^{-1} I)$, i.e., the $\binom{n}{2}$-dimensional Gaussian distribution with mean μ and isotropic covariance matrix $(2\lambda)^{-1} I$ injected by $\Phi(\sigma)$. Notably, $g^\dagger(\cdot|\mu, \lambda)$ is not a valid probability mass function over $\mathbb{S}_n$.

The mixture modeling approach stated above using $g^\dagger$ instead of f is in fact equivalently stated in Algorithm 2. It is worthwhile to note that the algorithm also applies to total/partial/multivariate/uncertain rankings with appropriate kernels defined as [77, Table 2] provides the counterpart of Algorithm 2 in case that a kernel matrix evaluated between data points is given instead. However, since $g^\dagger$ itself is not a valid probability mass function over $\mathbb{S}_n$, an evident drawback is that we now lose the probabilistic interpretation of the mixture distribution as in (19).

7.3 Experiments

Clustering 1980 APA election data. In the 1980 American Psychological Association (APA) presidential election, voters were asked to rank 5 candidates in order of preference, and 5738 votes in form of total rankings were reported and thus used in our experiment. The dataset was thoroughly studied by [18].

Algorithm 2 Kernel trick embedded Gaussian mixture model for clustering heterogeneous rank data.

Input: a collection of permutations $\{\sigma_i\}_{i=1}^m$ and a kernel function K over $\mathbb{S}_n$; the number of clusters $c \leq m$.

1: Compute feature points $\Phi(\sigma_i) \in \mathbb{R}^{\binom{n}{2}}$ mapped by the Kendall embedding.
2: Fit a Gaussian mixture model for $\{\Phi(\sigma_i)\}_{i=1}^m$ in $\mathbb{R}^{\binom{n}{2}}$ using maximum likelihood with the EM algorithm under the constraint of isotropic covariance matrix, i.e., $\Sigma = (2\lambda)^{-1}I$.
3: Use the membership estimates $\hat{\underline{z}}$ to form clusters by (21).

Output: Cluster assignments $\{S_j\}_{j=1}^c$.

We first use k-means approaches to cluster the data. We compare the proposed kernel k-means algorithm (Algorithm 1 with Kendall kernel K_τ) to the classic k-means algorithm formulated as (17). For the classic k-means where cluster centers are required to be a prototype permutation, three methods are employed in the center-update step for each iteration: brute-force search of Kemeny-optimal ranking, approximate ranking induced by Borda Count and Copeland's method. In each case, we vary the number of clusters ranging from 2 to 10 and the algorithm is repeated 50 times with randomly initialized configurations for each fixed number of clusters. We observe from Figure 3 (Left) that the kernel k-means or classic k-means with approximate centers runs much faster than optimal k-means for which the Kemeny-optimal ranking is time-consuming to find by a brute-force search. Further, Figure 3 (Middle) shows that the kernel k-means outperforms all three methods based on classic k-means in terms of the average silhouette scores of the clustering results, which justifies that the kernel k-means splits the data into more consistent sub-groups in the sense that instances, measured by Kendall tau distance on average, are more similar in the same cluster and more dissimilar in different clusters. We also observe that kernel k-means returns more robust clusters in case of perturbation in data (see appendices).

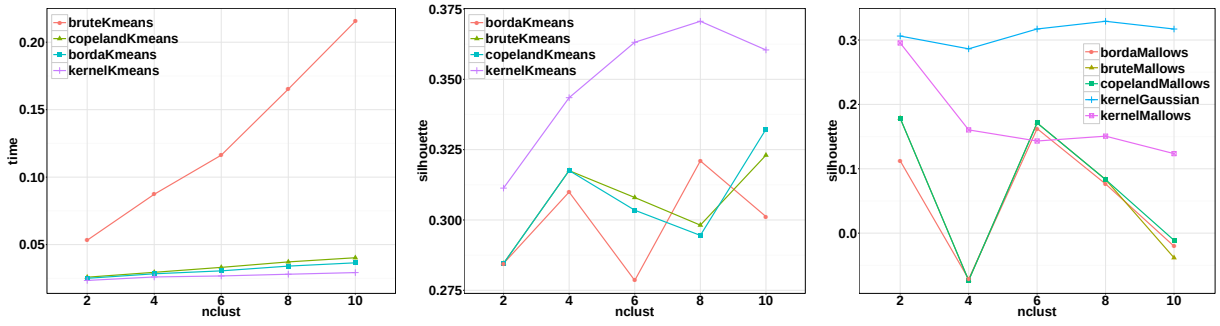


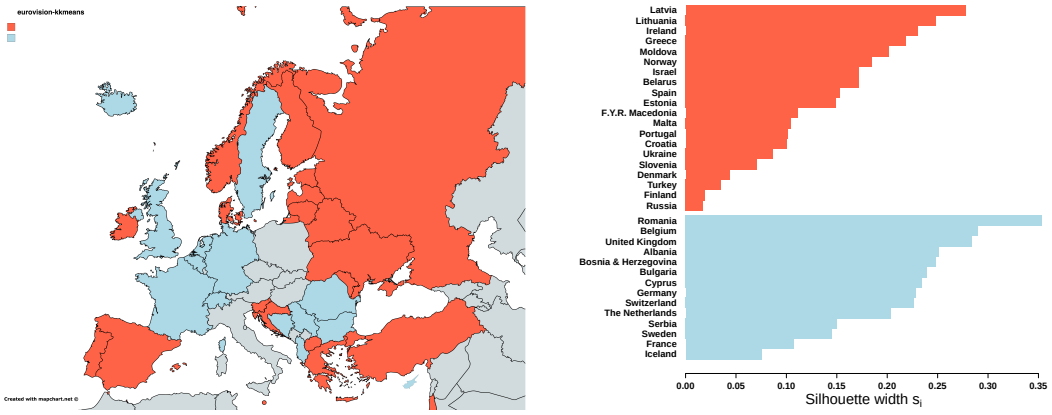
Figure 3: Across different number of clusters: **Left**: Computational time (in seconds) of k-means algorithms per run. **Middle**: Average silhouette scores of k-means methods. **Right**: Average silhouette scores of Mallows mixture modeling methods.

Mixture modeling is then used to fit the data and a partition of the votes is converted from the fitted models forming a clustering result. Baseline models are the Mallows mixture models fitted by the EM algorithm [56] using three different center-update algorithms at each

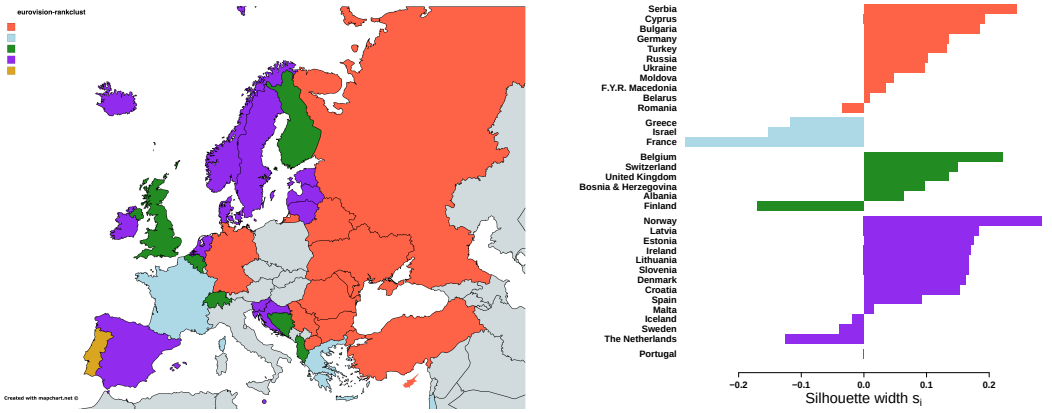
iteration: brute-force search for Kemeny-optimal ranking, approximate ranking induced by Borda Count and Copeland's method. As proposed in this paper, we embed the kernel trick in Mallows mixture modeling with two alternatives g (22) and $g^\dagger$ (23) in place of f (18). In each case, we vary the number of clusters ranging from 2 to 10 and the algorithm is repeated 50 times with randomly initialized configurations for each fixed number of clusters. As shown in Figure 3 (Right), modeling a Gaussian mixture to data in the feature space, or equivalently using $g^\dagger$ instead of f, provides a preferable split of the data into sub-groups with higher average silhouette scores across different number of clusters selected *a priori*.

Clustering ESC voting data. We finally perform clustering on a dataset of multivariate partial rankings. In the finale of Eurovision Song Contest (ESC), each participating country casts one top-k vote over the finalists who represent their home country. Taken from [34], the dataset consists of 34 multivariate ranking instances, each being a series of 6 partial votes over top 8 finalists from 2007 to 2012 respectively.

In comparison with the mixture of Insertion Sorting Rank (ISR) model for clustering multivariate partial rank data proposed by [34], we implement the kernel k-means algorithm (Algorithm 1) with the extended Kendall kernel to multivariate rankings (8) and equal weights $\mu_j = 1/p$ where $p = 6$ corresponding to the six contests across years. For each fixed number of clusters, the kernel k-means algorithm is repeated 100 times with randomly initialized configurations while 10 times for the ISR mixture modeling approach. We vary the number of clusters ranging from 2 to 6, and the optimal number is selected to be 2 for kernel k-means with respect to highest average silhouette score while 5 for the ISR mixture model with respect to highest BIC value. It consumes 2 hours in total to fit the ISR mixture model in order for clustering while it only takes less than 10 seconds to form the partition of data with kernel k-means. Although it is beyond the scope of this study to further explore the meaningful voting blocs, the colored map of Asia-Europe in terms of clustering results of participating countries to the ESC according to their voting behavior (Figure 4, Left) depicts that there exists interesting geographical alliances between countries in the voting data. For example, country-clusters returned by both algorithms present a sign of strong amity within Eastern Europe. Silhouette plots for both algorithms are shown in Figure 4 (Right). Despite a relatively small number of clusters selected for the kernel k-means, the silhouette plot (Figure 4a, Right) attests that the underlying clusters are well formed. Note that both silhouette plots opt for the same distance used by kernel k-means, which may show bias against a clustering scheme based on probabilistic modeling with ISR mixtures. However, the two approaches behave distinctly in identifying subgroups. For example, the ISR mixture model distinguishes Portugal as a singleton among all countries, while interpreting such clustering results remains to be studied. On the other hand, the k-means based approach tends to find more evenly distributed subgroups, in the sense that the number of individuals in each subgroup is more consistent. Therefore kernel k-means clustering is favored if the study of interest lies in populous behaviors in voting despite of potential outlier individuals. Notably the detection of outliers can be done by other kernel algorithms (Section 9).



(a) Country-clusters returned by kernel k-means, where the number of clusters 2 is selected with respect to highest silhouette score averaged over all countries.



(b) Country-clusters returned by ISR mixture modeling, where the number of clusters 5 (including in particular "Portugal" as a singleton) is selected with respect to highest fitted BIC value.

Figure 4: Clustering results of participating countries to the ESC according to their voting behavior illustrated by geographic map (Left) and silhouette plot (Right).

8 Application: Supervised classification of biomedical data

In this section we illustrate the relevance of supervised classification of rank data with an SVM using the Kendall kernel, when the ranking are derived from a high-dimensional real-valued vector. More precisely, we investigate the performance of classifying high-dimensional biomedical data, motivated by previous work demonstrating the relevance of replacing numerical features by pairwise comparisons in this context [24, 71, 79, 49].

For that purpose, we collected 10 datasets related to human cancer research publicly available online [48, 64, 66], as summarized in Table 1. The features are proteomic spectra relative intensities for the *Ovarian Cancer* dataset and gene expression levels for all the others. The contrasting classes are typically "Non-relapse v.s. Relapse" in terms of cancer prognosis, or "Normal v.s. Tumor" in terms of cancer identification. The datasets have no missing values, except the *Breast Cancer 1* dataset for which we performed additional

Table 1: Summary of biomedial datasets.

Dataset	No. of features	No. of samples (training/test)		Reference
		C_1	C_2	
Breast Cancer 1	23624	44/7 (Non-relapse)	32/12 (Relapse)	[74]
Breast Cancer 2	22283	142 (Non-relapse)	56 (Relapse)	[17]
Breast Cancer 3	22283	71 (Poor Prognosis)	138 (Good Prognosis)	[78]
Colon Tumor	2000	40 (Tumor)	22 (Normal)	[2]
Lung Adenocarcinoma 1	7129	24 (Poor Prognosis)	62 (Good Prognosis)	[8]
Lung Cancer 2	12533	16/134 (ADCA)	16/15 (MPM)	[27]
Medulloblastoma	7129	39 (Failure)	21 (Survivor)	[58]
Ovarian Cancer	15154	162 (Cancer)	91 (Normal)	[57]
Prostate Cancer 1	12600	50/9 (Normal)	52/25 (Tumor)	[67]
Prostate Cancer 2	12600	13 (Non-relapse)	8 (Relapse)	[67]

preprocessing to remove missing values as follows: first we removed two samples (both labeled "relapse") from the training set that have around 10% and 45% of missing gene values; next we discarded any gene whose value was missing in at least one sample, amounting to a total of 3.5% of all genes.

We compare the Kendall kernel to three standard kernels, namely linear kernel, homogeneous 2nd-order polynomial kernel and Gaussian RBF kernel with bandwidth set with "median trick", using SVM (with regularization parameter C) and Kernel Fisher Discriminant (KFD, without tuning parameter) as classifiers. In addition, we include in the benchmark classifiers based on Top Scoring Pairs (TSP) [24], namely (1-)TSP, k-TSP [71][1] and APMV (all-pairs majority votes, i.e. $\binom{n}{2}$-TSP). Finally we also test SVM with various kernels using as input only top features selected by TSP [66].

In all experiments, each kernel is centered (on the training set) and scaled to unit norm in the feature space. For KFD-based models, we add 10^{-3} on the diagonal of the centered and scaled kernel matrix, as suggested by [54]. The Kendall kernel we use in practice is a soft version to (9) in the sense that the extremes ± 1 can still be attained in the presence of ties, specifically we use

$$K_\tau(\mathbf{x}, \mathbf{x}') = \frac{n_c(\mathbf{x}, \mathbf{x}') - n_d(\mathbf{x}, \mathbf{x}')}{\sqrt{(n_0 - n_1)(n_0 - n_2)}},$$

where $n_0 = \binom{n}{2}$ and n_1, n_2 are the number of tied pairs in $\mathbf{x}, \mathbf{x}'$ respectively.

For the three datasets that are split into training and test sets, we report the performance on the test set; otherwise we perform a 5-fold cross-validation repeated 10 times and report the mean performance over the $5 \times 10 = 50$ splits to evaluate the performance of the different methods. In addition, on each training set, an internal 5-fold cross-validation is performed to tune parameters, namely the C parameter of SVM-based models (optimized over a grid ranging from 10^{-2} to 10^3 in log scale), and the number k of k-TSP in case of feature selection (ranging from 1 to 5000 in log scale).

Table 2 and Figure 5 (Left) summarize the performance of each model across the datasets.

[1]While the original k-TSP algorithm selects only top k *disjoint* pairs with the constraint that k is *less than* 10, we do not restrict ourselves to any of these two conditions since we consider k-TSP in this study essentially a feature pair scoring algorithm.

Table 2: Prediction accuracy (%) of different methods across biomedical datasets. Models are named after candiate methods (SVM or KFD) and candiate kernels, namely linear kernel (linear), 2nd-order homogeneous polynomial kernel (poly), Gaussian RBF kernel (rbf) or Kendall kernel (kdt), and whether feature selection is combined (TOP) or not (ALL).

	Average	BC1	BC2	BC3	CT	LA1	LC2	MB	OC	PC1	PC2
SVMkdtALL	79.39	78.95	71.31	67.34	85.78	70.98	97.99	63.67	99.48	100	58.4
SVMlinearTOP	77.16	84.21	69.29	67.11	84.19	63.92	97.32	65.17	99.41	85.29	55.7
SVMlinearALL	76.09	78.95	71.67	64.27	86.73	70.23	97.99	62.67	99.64	73.53	55.17
SVMkdtTOP	75.5	52.63	70.61	65.81	85.46	67.7	97.99	58.33	99.92	97.06	59.47
SVMpolyALL	74.54	68.42	71.62	63.66	78.43	70.53	98.66	61.17	99.28	79.41	54.23
KFDkdtALL	74.33	63.16	59.41	67.22	85.46	59.08	99.33	59.33	98.73	97.06	54.57
kTSP	74.03	57.89	58.22	64.47	87.23	61.7	97.99	56	99.92	100	56.83
SVMpolyTOP	73.99	63.16	69.44	66.26	79.14	65.98	99.33	60	99.21	88.24	49.1
KFDlinearALL	71.81	63.16	60.43	67.52	77.26	57.24	97.99	59.5	100	73.53	61.43
KFDpolyALL	71.39	63.16	60.48	67.38	75.1	58.52	97.99	60.33	100	73.53	57.43
TSP	69.71	68.42	49.58	57.8	85.61	58.96	95.97	52.67	99.8	76.47	51.83
SVMrbfALL	69.31	63.16	71.41	65.87	81.18	70.84	93.96	63.83	98.85	26.47	57.5
KFDrbfALL	66.5	63.16	60.38	66.17	84.33	58.62	97.32	60.17	98.34	26.47	50
APMV	61.91	84.21	65.98	33.96	64.49	33.6	89.93	42.17	85.19	73.53	46

An SVM with the Kendall kernel achieves the highest average prediction accuracy overall (79.39%), followed by a linear SVM trained on a subset of features selected from the top scoring pairs (77.16%) and a standard linear SVM (76.09%). The SVM with Kendall kernel outperforms all the other methods at a P-value of 0.07 according to a Wilcoxon rank test. We note that even though models based on KFD generally are less accurate than those based on SVMs, the relative order of the different kernels is consistent between KFD and SVM, adding evidence that the Kendall kernel provides an interesting alternative to other kernels in this context. The performance of TSP and k-TSP, based on majority vote rules, are comparatively worse than SVMs using the same features, as already observed by [66].

Figure 5 further shows how the performance of different kernels depends on the choice of the C parameter or the SVM (Middle), and on the number of features used (Right), on some representative datasets. We observe that compared to other kernels, an SVM with the Kendall kernel is relatively insensitive to hyper-parameter C especially when C is large, which corresponds to a hard-margin SVM. This may explain in part the success of SVMs in this setting, since the risk of choosing a bad C during training is reduced. Regarding the number of features used in case of feature selection, we notice that it does not seem to be beneficial to perform feature selection in this problem, explaining why the Kendall kernel which uses all pairwise comparisons between features outperforms other kernels restricted to a subset of these pairs. In particular, the feature space of the Kendall and Mallows kernels is precisely the space of binary pairwise comparisons defined by [24], and the results show that instead of selecting a few features in this space as the Top Scoring Pairs (TSP)-family classifiers do [24, 71, 79, 49], one can simply work with *all* pairs with the kernel trick.

Finally, as a proof of concept we empirically compare on one dataset the smooth alternative (12) and its Monte Carlo approximate (14) with the original Kendall kernel. Figure 6 shows how the performance varies with the amount of noise added to the samples (Left),

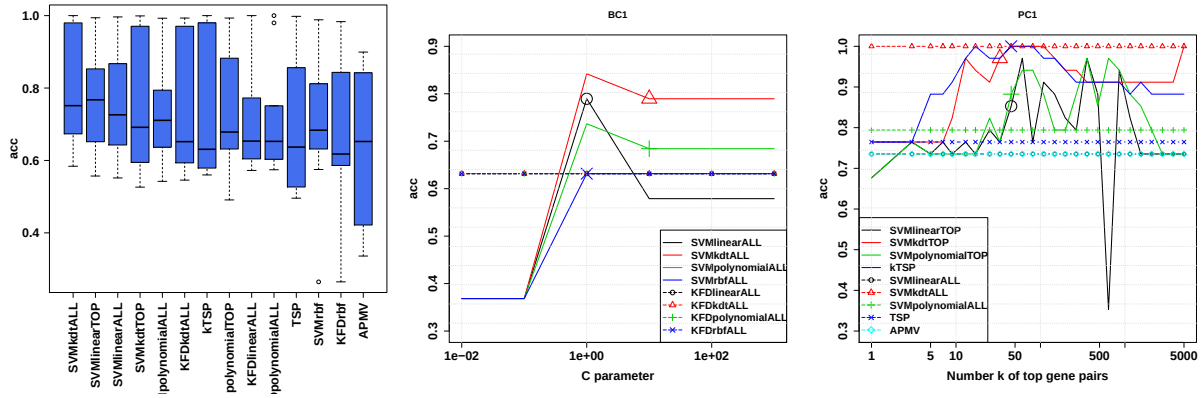


Figure 5: **Left**: Model performance comparison (ordered by decreasing average accuracy across datasets). **Middle**: Sensitivity of kernel SVMs to C parameter on the *Breast Cancer 1* dataset. **Right**: Impact of TSP feature selection on the *Prostate Cancer 1* dataset. (Special marks on SVM lines denote the parameter returned by cross-validation.)

and how the performance varies with the number of samples in the Monte Carlo scheme for a given amount of noise (Right). It confirms that the smooth alternative (12) can improve the performance of the Kendall kernel, and that the amount of noise (window size) should be considered as a parameter of the kernel to be optimized. Although the D^2-sample Monte Carlo approximate kernel (14) mainly serves as a fast estimate to the exact evaluation of (12), it shows that the idea of jittered input with specific noise can also bring a tempting benefit for data analysis with Kendall kernel, even when D is small. This also justifies the motivation of our proposed smooth alternative (12). Last but not least, despite the fact that the convergence rate of D^2-sample Monte Carlo approximate to the exact kernel evaluation is guaranteed by Theorem 3, experiments show that the convergence in practice is typically faster than the theoretical bound, and even faster in case that the window size a is small. This is due to the fact that the convergence rate is also dependent of the observed data distribution in the input space, for which we have not made any specific assumption in our analysis.

9 Conclusion and discussion

Based on the observation that the popular Kendall tau correlation between total rankings is a positive definite kernel, we presented some extensions and applications pertaining to learning with the Kendall kernel and the related Mallows kernel. We showed that both kernels can be evaluated efficiently in $O(n \log n)$ time, and that the Kendall kernel can be extended to partial rankings containing k items out of n in $O(k \log k)$ time as well as to multivariate rankings. When permutations are obtained by sorting real-valued vectors, we proposed an extension of the Kendall kernel based on random perturbations of the input vector to increase its robustness to small variations, and discussed two possible algorithms to compute it. We further highlighted a connection between the fast Mallow kernel and the diffusion kernel of [43]. We also reported promising experimental results on clustering of

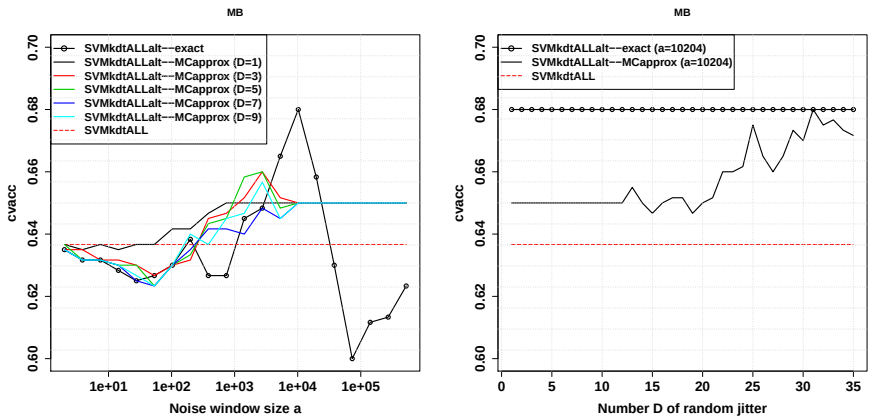


Figure 6: **Left**: Empirical performance of smoothed alternative to Kendall kernel on the *Medulloblastoma* dataset. **Right**: Empirical convergence of Monte Carlo approximate at the fixed window size attaining maximum underlying accuracy from the left plot.

heterogeneous rank data and classifying biomedical data demonstrating that for highly noisy data, the Kendall kernel is competitive or even outperforms other state-of-the-art kernels.

We believe that computationally efficient kernels over the symmetric group pave the way to numerous applications beyond the ones we pursued in this paper. In unsupervised data mining, kernel density estimation for example can be applied to modeling the distribution over a collection of rankings, and by the representer theorem the resulting distribution depends solely on the observed data points circumventing the exponentially large cardinality of the symmetric group, from which a consensus ranking that best represents the data is the one with the highest probability. As more complicated cases, there is much interest beyond finding a single consensus ranking typically in the context of political votes or social choices: groups of homogeneous sub-populations in data can be clustered by algorithms such as kernel k-means or spectral clustering [20]; dependencies or principle structural factors in data can be found by kernel canonical correlation analysis [44] or kernel principle component analysis [61]; outliers in a collection of rank data can be detected with one-class SVMs [62, 72]. In a more predictive setting, Support Vector Machines and kernel ridge regression are representative delegates for solving classification and regression problems amongst many other kernel algorithms [60]. Notably, the input/output kernels formalism allows us to predict rankings as well as learn from rankings where a wealth of algorithms such as multi-class SVMs or structural SVMs [14, 73, 5] are ready to suit the problem at hand.

Deeper understanding of the Kendall and Mallows kernels calls for more theoretical work of the proposed kernels. In particular, a detailed analysis of the Fourier spectra of the Kendall and Mallows kernels is provided in [51]. Those authors also introduced a tractable family of normalized polynomial kernels of degree p that interpolates between Kendall (degree one) and Mallows (infinite degree) kernels.

There are many interesting extensions of the current work. One direction would be to include high-order comparisons in measuring the similarity between permutations. Since the fast computation of the Kendall and Mallows kernels is balanced by the fact that they only rely on pairwise statistics between the ranks, computationally tractable extension to higher-

order statistics, such as three-way comparisons, could potentially enhance the discriminative power of the proposed kernels. Another interesting direction would be to extend the proposed kernels to rankings on partially ordered set. In fact, the current work lies on the assumption that a (strict) total order can be associated with the (finite) set of items given to rank $\{x_1, \ldots, x_n\}$, which is implicitly presumed when we label the items by the subscripts $[1, n]$ and then define the Kendall and Mallows kernels by comparing all item pairs (i, j) for $i < j$ (Section 2). However, there are cases when the item set is intrinsically associated with a (strict) partial order such that some item pairs are conceptually incomparable. In that case, we can collect all comparable item pairs into a set denoted by E and define the kernels by comparing only those item pairs (i, j) in E. Notably evaluating the extended kernels is still fast as we can simply replace the Merge Sort algorithm for total orders (Section 2) by a topological sort algorithm for partial orders [12, Section 22.4]. We leave further investigations of this generalization to future work.

Acknowledgments

This work was supported by the European Union 7th Framework Program through the Marie Curie ITN MLPM grant No 316861, the European Research Council grant ERC-SMAC-280032, the Miller Institute for Basic Research in Science [to JPV]; and the Fulbright Foundation [to JPV]. We thank anonymous reviewers for interesting comments, in particular the possibility to extend the kernels to partial orders.

References

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

[2] U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proc. Natl. Acad. Sci. U. S. A.*, 96(12): 6745–6750, 1999.

[3] K. J. Arrow. *Social Choice and Individual Values*, volume 12. Yale Univ Press, 2012.

[4] F. R. Bach, G. R. G. Lanckriet, and M. I. Jordan. Multiple kernel learning, conic duality, and the SMO algorithm. In *Proceedings of the Twenty-First International Conference on Machine Learning*, page 6, New York, NY, USA, 2004. ACM.

[5] G. H. Bakir, T. Hofmann, B. Schölkopf, A. J. Smola, B. Taskar, and S. V. N. Vishwanathan. *Predicting Structured Data*. MIT Press, 2007.

[6] M.-F. Balcan, N. Bansal, A. Beygelzimer, D. Coppersmith, J. Langford, and G. B. Sorkin. Robust reductions from ranking to classification. *Mach. Learn.*, 72(1–2):139–153, 2008.

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

[8] D. G. Beer, S. L. R. Kardia, C.-C. Huang, T. J. Giordano, A. M. Levin, D. E. Misek, L. Lin, G. Chen, T. G. Gharib, D. G. Thomas, M. L. Lizyness, R. Kuick, S. Hayasaka, J. M. G. Taylor, M. D. Iannettoni, M. B. Orringer, and S. Hanash. Gene-expression profiles predict survival of patients with lung adenocarcinoma. *Nat. Med.*, 8(8):816–824, Aug 2002.

[9] L. Bottou and O. Bousquet. The tradeoffs of large scale learning. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Adv. Neural Inform. Process. Syst.*, volume 20, pages 161–168. Curran Associates, Inc., 2008.

[10] S. Boucheron, G. Lugosi, and P. Massart. *Concentration Inequalities*. Oxford Univ Press, 2013.

[11] A. H. Copeland. A reasonable social welfare function. In *University of Michigan Seminar on Applications of Mathematics to the Social Sciences*. 1951.

[12] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. The MIT Press, 3rd edition, 2009.

[13] C. Cortes and V. Vapnik. Support-vector networks. *Mach. Learn.*, 20(3):273–297, 1995. ISSN 0885-6125.

[14] K. Crammer and Y. Singer. On the algorithmic implementation of multiclass kernel-based vector machines. *J. Mach. Learn. Res.*, 2:265–292, 2002.

[15] D. E. Critchlow. *Metric methods for analyzing partially ranked data*. Springer, 1985.

[16] J. C. de Borda. Mémoire sur les élections au scrutin. *Histoire de l'Academie Royale des Sciences*, 1781.

[17] C. Desmedt, F. Piette, S. Loi, Y. Wang, F. Lallemand, B. Haibe-Kains, G. Viale, M. Delorenzi, Y. Zhang, M. S. d'Assignies, J. Bergh, R. Lidereau, P. Ellis, A. L. Harris, J. G. M. Klijn, J. A. Foekens, F. Cardoso, M. J. Piccart, M. Buyse, C. Sotiriou, and T. R. A. N. S. B. I. G Consortium . Strong time dependence of the 76-gene prognostic signature for node-negative breast cancer patients in the TRANSBIG multicenter independent validation series. *Clin. Cancer Res.*, 13(11):3207–3214, 2007.

[18] P. Diaconis. *Group representations in probability and Statistics*, volume 11 of *Lecture Notes–Monograph Series*. Institut of Mathematical Statistics, Hayward, CA, 1988.

[19] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank aggregation methods for the web. In *Proceedings of the Tenth International Conference on World Wide Web*, pages 613–622. ACM, 2001.

[20] M. Filippone, F. Camastra, F. Masulli, and S. Rovetta. A survey of kernel and spectral methods for clustering. *Pattern Recogn.*, 41(1):176–190, 2008.

[21] M. A. Fligner and J. S. Verducci. Distance based ranking models. *J. R. Stat. Soc. Ser. B*, 48(3):359–369, 1986.

[22] K. Fukumizu, A. Gretton, B. Schölkopf, and B. K. Sriperumbudur. Characteristic kernels on groups and semigroups. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Adv. Neural Inform. Process. Syst.*, volume 21, pages 473–480. 2008.

[23] T. Gärtner, J.W. Lloyd, and P.A. Flach. Kernels and distances for structured data. *Mach. Learn.*, 57(3):205–232, 2004.

[24] D. Geman, C. d'Avignon, D. Q. Naiman, and R. L. Winslow. Classifying gene expression profiles from pairwise mRNA comparisons. *Stat. Appl. Genet. Mol. Biol.*, 3(1):Article19, 2004.

[25] M. Girolami. Mercer kernel-based clustering in feature space. *IEEE Trans. Neural Network*, 13(3):780–784, 2002.

[26] M. Gönen and E. Alpaydın. Multiple kernel learning algorithms. *J. Mach. Learn. Res.*, 12:2211–2268, 2011.

[27] G. J. Gordon, R. V. Jensen, L.-L. Hsiao, S. R. Gullans, J. E. Blumenstock, S. Ramaswamy, W. G. Richards, D. J. Sugarbaker, and R. Bueno. Translation of microarray data into clinically relevant cancer diagnostic tests using gene expression ratios in lung cancer and mesothelioma. *Cancer Res.*, 62(17):4963–4967, 2002.

[28] I. C. Gormley and T. B. Murphy. Analysis of irish third-level college applications data. *J. Roy. Stat. Soc. Stat. Soc.*, 169(2):361–379, 2006.

[29] I. C. Gormley and T. B. Murphy. Exploring voting blocs within the irish electorate: A mixture modeling approach. *J. Am. Stat. Assoc.*, 103(483):1014–1027, 2008.

[30] D. Haussler. Convolution kernels on discrete structures. Technical Report UCSC-CRL-99-10, UC Santa Cruz, 1999.

[31] D. P. Helmbold and M. K. Warmuth. Learning permutations with exponential weights. *J. Mach. Learn. Res.*, 10:1705–1736, 2009.

[32] J. Huang, C. Guestrin, and L. Guibas. Fourier theoretic probabilistic inference over permutations. *J. Mach. Learn. Res.*, 10:997–1070, 2009.

[33] L. Hubert and P. Arabie. *Comparing partitions*, volume 2. Springer, 1985.

[34] J. Jacques and C. Biernacki. Model-based clustering for multivariate partial ranking data. *J. Stat. Plann. Infer.*, 149:201–217, 2014.

[35] T. Kamishima. Nantonac collaborative filtering: recommendation based on order responses. In *Proceedings of the Ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 583–588. ACM, 2003.

[36] H. Kashima, K. Tsuda, and A. Inokuchi. Marginalized kernels between labeled graphs. In T. Faucett and N. Mishra, editors, *Proceedings of the Twentieth International Conference on Machine Learning*, pages 321–328, New York, NY, USA, 2003. AAAI Press.

[37] J. G. Kemeny and J. L. Snell. *Mathematical models in the social sciences*, volume 9. Ginn New York, 1962.

[38] M. G. Kendall. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93, 1938.

[39] M. G. Kendall. *Rank correlation methods*. Griffin, 1948.

[40] W. R. Knight. A computer method for calculating Kendall's tau with ungrouped data. *J. Am. Stat. Assoc.*, 61(314):436–439, 1966.

[41] I. R. Kondor. *Group theoretical methods in machine learning*. PhD thesis, Columbia University, 2008.

[42] I. R. Kondor and J. Lafferty. Diffusion kernels on graphs and other discrete input spaces. In *Proceedings of the Nineteenth International Conference on Machine Learning*, volume 2, pages 315–322, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc.

[43] R. I. Kondor and M. S. Barbosa. Ranking with kernels in fourier space. In A. T. Kalai and M. Mohri, editors, *COLT 2010 - The 23rd Conference on Learning Theory, Haifa, Israel, June 27-29, 2010*, pages 451–463. Omnipress, 2010.

[44] P. L. Lai and C. Fyfe. Kernel and nonlinear canonical correlation analysis. *Int. J. Neural Syst.*, 10(05):365–377, 2000.

[45] G. R. G. Lanckriet, N. Cristianini, P. Bartlett, L. El Ghaoui, and M. I. Jordan. Learning the kernel matrix with semidefinite programming. *J. Mach. Learn. Res.*, 5:27–72, 2004.

[46] G. R. G. Lanckriet, T. De Bie, N. Cristianini, M. I. Jordan, and W. S. Noble. A statistical framework for genomic data fusion. *Bioinformatics*, 20(16):2626–2635, Nov 2004.

[47] G. Lebanon and Y. Mao. Non-parametric modeling of partially ranked data. *J. Mach. Learn. Res.*, 9:2401–2429, 2008.

[48] J. Li, H. Liu, and L. Wong. Mean-entropy discretized features are effective for classifying high-dimensional biomedical data. In M. J. Zaki, J. T.-L. Wang, and H. Toivonen, editors, *Proceedings of the 3nd ACM SIGKDD Workshop on Data Mining in Bioinformatics (BIOKDD 2003), August 27th, 2003, Washington, DC, USA*, pages 17–24, 2003.

[49] X. Lin, B. Afsari, L. Marchionni, L. Cope, G. Parmigiani, D. Naiman, and D. Geman. The ordering of expression among a few genes can provide simple cancer biomarkers and signal BRCA1 mutations. *BMC Bioinformatics*, 10:256, 2009.

[50] C. L. Mallows. Non-null ranking models. i. *Biometrika*, 44(1/2):114–130, 1957.

[51] H. Mania, A. Ramdas, M. J. Wainwright, M. I. Jordan, and B. Recht. Universality of Mallows' and degeneracy of Kendall's kernels for rankings. *arXiv:1603.08035*, 2016.

[52] J. I. Marden. *Analyzing and modeling rank data*. CRC Press, 1996.

[53] M. Meilă, K. Phadnis, A. Patterson, and J. Bilmes. Consensus ranking under the exponential model. In *Proceedings of the Twenty-Third Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-07)*, pages 285–294, Corvallis, Oregon, 2007. AUAI Press.

[54] S. Mika, G. Rätsch, J. Weston, B. Schölkopf, and K.R. Müller. Fisher discriminant analysis with kernels. In Y.-H. Hu, J. Larsen, E. Wilson, and S. Douglas, editors, *Neural Networks for Signal Processing IX*, pages 41–48. IEEE, 1999.

[55] K. Muandet, K. Fukumizu, F. Dinuzzo, and B. Schölkopf. Learning from distributions via support measure machines. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Adv. Neural Inform. Process. Syst.*, volume 25, pages 10–18. Curran Associates, Inc., 2012.

[56] T. B. Murphy and D. Martin. Mixtures of distance-based models for ranking data. *Comput. Stat. Data Anal.*, 41(3):645–655, 2003.

[57] E. F. Petricoin, A. M. Ardekani, B. A. Hitt, P. J. Levine, V. A. Fusaro, S. M. Steinberg, G. B. Mills, C. Simone, D. A. Fishman, E. C. Kohn, and L. A. Liotta. Use of proteomic patterns in serum to identify ovarian cancer. *Lancet*, 359(9306):572–577, 2002.

[58] S. L. Pomeroy, P. Tamayo, M. Gaasenbeek, L. M. Sturla, M. Angelo, M. E. McLaughlin, J. Y. H. Kim, L. C. Goumnerova, P. M. Black, C. Lau, J. C. Allen, D. Zagzag, J. M. Olson, T. Curran, C. Wetmore, J. A. Biegel, T. Poggio, S. Mukherjee, R. Rifkin, A. Califano, G. Stolovitzky, D. N. Louis, J. P. Mesirov, E. S. Lander, and T. R. Golub. Prediction of central nervous system embryonal tumour outcome based on gene expression. *Nature*, 415(6870):436–442, 2002.

[59] I. J. Schoenberg. Metric spaces and positive definite functions. *Trans. Am. Math. Soc.*, 44(3):522–536, 1938.

[60] B. Schölkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, 2002.

[61] B. Schölkopf, A. J. Smola, and K. R. Müller. Kernel principal component analysis. In *Advances in kernel methods*, pages 327–352. MIT Press, 1999.

[62] B. Schölkopf, R. C. Williamson, A. J. Smola, J. Shawe-Taylor, and J. C. Platt. Support vector method for novelty detection. In *Adv. Neural Inform. Process. Syst.*, volume 12, pages 582–588. 1999.

[63] B. Schölkopf, K. Tsuda, and J.-P. Vert. *Kernel Methods in Computational Biology*. MIT Press, The MIT Press, Cambridge, Massachussetts, 2004.

[64] M. Schroeder, B. Haibe-Kains, A. Culhane, C. Sotiriou, G. Bontempi, and J. Quackenbush. *breastCancerTRANSBIG: Gene expression dataset published by Desmedt et al. [2007] (TRANSBIG).*, 2011. R package version 1.2.0.

[65] J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, New York, NY, USA, 2004.

[66] P. Shi, S. Ray, Q. Zhu, and M. A. Kon. Top scoring pairs for feature selection in machine learning and applications to cancer outcome prediction. *BMC Bioinformatics*, 12:375, 2011.

[67] D. Singh, P. G. Febbo, K. Ross, D. G. Jackson, J. Manola, C. Ladd, P. Tamayo, A. A. Renshaw, A. V. D'Amico, J. P. Richie, E. S. Lander, M. Loda, P. W. Kantoff, T. R. Golub, and W. R. Sellers. Gene expression correlates of clinical prostate cancer behavior. *Cancer Cell*, 1(2):203–209, 2002.

[68] A. Smola, A. Gretton, L. Song, and B. Schölkopf. *A Hilbert Space Embedding for Distributions*, pages 13–31. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007. ISBN 978-3-540-75225-7. doi: 10.1007/978-3-540-75225-7_5.

[69] S. Sonnenburg, G. Rätsch, C. Schäfer, and B. Schölkopf. Large scale multiple kernel learning. *J. Mach. Learn. Res.*, 7:1531–1565, 2006.

[70] I. Steinwart. Consistency of support vector machines and other regularized kernel classifiers. *IEEE Trans. Inform. Theory*, 51(1):128–142, 2005.

[71] A. C. Tan, D. Q. Naiman, L. Xu, R. L. Winslow, and D. Geman. Simple decision rules for classifying human cancers from gene expression profiles. *Bioinformatics*, 21(20): 3896–3904, 2005.

[72] D. M. Tax and R. P. Duin. Support vector data description. *Mach. Learn.*, 54(1):45–66, 2004.

[73] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *J. Mach. Learn. Res.*, 6:1453–1484, 2005.

[74] L. J. van 't Veer, H. Dai, M. J. van de Vijver, Y. D. He, A. A. M. Hart, M. Mao, H. L. Peterse, K. van der Kooy, M. J. Marton, A. T. Witteveen, G. J. Schreiber, R. M. Kerkhoven, C. Roberts, P. S. Linsley, R. Bernards, and S. H. Friend. Gene expression profiling predicts clinical outcome of breast cancer. *Nature*, 415(6871):530–536, 2002.

[75] V. N. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.

[76] S. V. N. Vishwanathan, N. N. Schraudolph, R. Kondor, and K. M. Borgwardt. Graph kernels. *J. Mach. Learn. Res.*, 10:1–41, 2009.

[77] J. Wang, J. Lee, and C. Zhang. Kernel trick embedded gaussian mixture model. In *Algorithmic Learning Theory*, pages 159–174. Springer, 2003.

[78] Y. Wang, F. S. Makedon, J. C. Ford, and J. Pearlman. HykGene: a hybrid approach for selecting marker genes for phenotype classification using microarray gene expression data. *Bioinformatics*, 21(8):1530–1537, 2005.

[79] L. Xu, A. C. Tan, D. Q. Naiman, D. Geman, and R. L. Winslow. Robust prostate cancer marker genes emerge from direct integration of inter-study microarray data. *Bioinformatics*, 21(20):3905–3911, 2005.

[80] R. Zhang and A. Rudnicky. A large scale clustering scheme for kernel k-means. In *Proceedings of the Sixteenth International Conference on Pattern Recognition*, volume 4, pages 289–292. IEEE, 2002.

A Proof of theorems

Proof of Theorem 2. The proof is constructive. We show here explicitly how to compute the Kendall kernel between two interleaving partial rankings while the idea remains similar for the case of top-k partial rankings. Denote by $[1, n]$ the item set to be ranked and by $A_{i_1,\ldots,i_k}, A_{j_1,\ldots,j_m} \subset \mathbb{S}_n$ two interleaving partial rankings of size k, m respectively, whose subsets of item indices are denoted by $I := \{i_1, \ldots, i_k\}$ and $J := \{j_1, \ldots, j_m\}$. We will lighten the notation by writing $A_I := A_{i_1,\ldots,i_k}$ and $A_J := A_{j_1,\ldots,j_m}$ and recall that by definition,

$$A_I = \{\pi \in \mathbb{S}_n | \pi(i_a) > \pi(i_b) \text{ if } a < b, a, b \in [1, k]\},$$
$$A_J = \{\pi' \in \mathbb{S}_n | \pi'(j_a) > \pi'(j_b) \text{ if } a < b, a, b \in [1, m]\}$$

are subsets of $\mathbb{S}_n$ compatible with the two partial rankings respectively. In particular, $|A_I| = n!/k!$ and $|A_J| = n!/m!$. Note that every item that does not appear in the partial ranking corresponding to A_I (or A_J) can be interleaved at any possible order with the other items for some permutation in that set.

Key observation to our proof is the "symmetry" of A_I (or A_J) in the sense that (i) for every item pair $\{i, j\}$ such that $i, j \in I$, all permutations in A_I are identical on the relative order of items i and j; (ii) for every item pair $\{i, j\}$ such that $i, j \in I^{\complement}$, there exists a unique permutation $\rho = (i, j) \circ \pi \in A_I$ for each $\pi \in A_I$ by swapping the ranks of items i, j in π such that $(\pi(i) - \pi(j))(\rho(i) - \rho(j)) < 0$ and ρ is identical with π on the absolute ranks of all the other items.

By the definition of convolution kernel and Theorem 1, we have

$$K_\tau(A_I, A_J) = \frac{1}{|A_I||A_J|} \sum_{\substack{\pi \in A_I \\ \pi' \in A_J}} \frac{1}{\binom{n}{2}} \sum_{1 \le i < j \le n} \text{sign}(\pi(i) - \pi(j)) \text{sign}(\pi'(i) - \pi'(j))$$
$$= \sum_{1 \le i < j \le n} \frac{k!m!}{(n!)^2 \binom{n}{2}} \sum_{\substack{\pi \in A_I \\ \pi' \in A_J}} \text{sign}(\pi(i) - \pi(j)) \text{sign}(\pi'(i) - \pi'(j)) \,. \quad (24)$$

As we will always regard the item set $[1, n]$ as the universe, we will write the complement of set $S \subset [1, n]$ as $S^{\complement} := [1, n] \setminus S$. Since the item set can be divided into four disjoint subsets that are $[1, n] = (I \cap J) \sqcup (I \setminus J) \sqcup (J \setminus I) \sqcup (I \cup J)^{\complement}$, any (unordered) item pair $\{i, j\}$ can be categorized uniquely into one out of ten cases:

1 both items in $I \cap J$.

2 one item in $I \cap J$, the other in $I \setminus J$.

3 one item in $I \cap J$, the other in $J \setminus I$.

4 one item in $I \cap J$, the other in $(I \cup J)^{\complement}$.

5 one item in $I \setminus J$, the other in $J \setminus I$.

6 both items in $I \setminus J$.

7 both items in $J \setminus I$.

8 both items in $(I \cup J)^{\complement}$.

9 one item in $I \setminus J$, the other in $(I \cup J)^{\complement}$.

10 one item in $J \setminus I$, the other in $(I \cup J)^{\complement}$.

Now we can split and case-by-case regroup the additive terms in (24) into ten parts. We denote by s_1 to s_{10} the subtotal corresponding to cases 1 to 10, i.e.,

$$K_\tau(A_I, A_J) = \sum_{l=1}^{10} s_l := \sum_{l=1}^{10} \Big\{ \sum_{\substack{\{i,j\} \text{ in} \\ \text{case } l}} \frac{k!m!}{(n!)^2 \binom{n}{2}} \times \sum_{\substack{\pi \in A_I \\ \pi' \in A_J}} \operatorname{sign}(\pi(i) - \pi(j)) \operatorname{sign}(\pi'(i) - \pi'(j)) \Big\}.$$

It is straightforward to see that s_6 to s_{10} are all equal to 0 due to the symmetry of A_I and/or A_J. For example for every item pair $\{i, j\}$ in case 6, since both items i and j appear in I, their relative order is fixed in the sense that $\operatorname{sign}(\pi(i) - \pi(j))$ remains constant for all $\pi \in A_I$; since both items are absent from J, we can pair up permutations $\pi', \rho' \in A_J$ such that $\operatorname{sign}(\pi'(i) - \pi'(j)) = -\operatorname{sign}(\rho'(i) - \rho'(j))$. As a result all additive terms in s_6 cancel out each other and thus $s_6 = 0$.

Now we will take efforts to compute s_1 to s_5. For every item pair $\{i, j\}$ in case 1 such that $i, j \in I \cap J$, since $i, j \in I$, their relative order remains unchanged for all $\pi \in A_I$ and let us denote by $\tau \in \mathbb{S}_{|I \cap J|}$ the total ranking of the observed items indexed by $I \cap J$ with respect to A_I. Since also $i, j \in J$, we can denote by $\tau' \in \mathbb{S}_{|I \cap J|}$ the total ranking of the observed

items indexed by $I \cap J$ with respect to A_J. Therefore we have

$$\begin{aligned} s_1 &= \sum_{\substack{1\le i<j\le n \\ i,j\in I\cap J}} \frac{k!m!}{(n!)^2\binom{n}{2}} \sum_{\substack{\pi\in A_I \\ \pi'\in A_J}} \operatorname{sign}(\pi(i)-\pi(j))\operatorname{sign}(\pi'(i)-\pi'(j)) \\ &= \frac{1}{\binom{n}{2}} \sum_{\substack{1\le i<j\le n \\ i,j\in I\cap J}} \operatorname{sign}(\tau(i)-\tau(j))\operatorname{sign}(\tau'(i)-\tau'(j)) \\ &= \frac{\binom{|I\cap J|}{2}}{\binom{n}{2}} K_\tau(\tau,\tau')\,, \end{aligned}$$

where the last line is by the definition of Kendall kernel between τ and τ' on the common items in $I \cap J$.

For every item pair $\{i, j\}$ in case 2, we may assume without loss of generality that $i \in I \cap J, j \in I \setminus J$, or equivalently $i, j \in I$ and $i \in J, j \notin J$. The relative order of i, j in $\pi \in A_I$ is thus determined by τ but not fixed for all $\pi' \in A_J$. Let us denote by $\sigma \in \mathbb{S}_k$ the total ranking corresponding to the k observed items in A_I and by $\sigma' \in \mathbb{S}_m$ the total ranking of the m observed items in A_J. In fact, there are $(m+1)$ possible positions for j to interleave in some $\pi' \in A_J$ and the number of positions with a lower relative order of j to i is $\sigma'(i)$. Therefore we have

$$\begin{aligned} s_2 =& \sum_{\substack{i\in I\cap J \\ j\in I\setminus J}} \frac{k!m!}{(n!)^2\binom{n}{2}} \sum_{\substack{\pi\in A_I \\ \pi'\in A_J}} \operatorname{sign}(\pi(i)-\pi(j))\operatorname{sign}(\pi'(i)-\pi'(j)) \\ =& \frac{1}{\binom{n}{2}} \sum_{\substack{i\in I\cap J \\ j\in I\setminus J}} \operatorname{sign}(\tau(i)-\tau(j)) \frac{m!}{n!} \sum_{\pi'\in A_J} \operatorname{sign}(\pi'(i)-\pi'(j)) \\ =& \frac{1}{\binom{n}{2}} \sum_{i\in I\cap J} \sum_{j\in I\setminus J} \Big\{ \operatorname{sign}(\tau(i)-\tau(j)) \frac{m!}{n!} \\ & \qquad \times \frac{n!}{(m+1)!} \big[\sigma'(i) - ((m+1)-\sigma'(i))\big] \Big\} \\ =& \frac{1}{\binom{n}{2}(m+1)} \sum_{i\in I\cap J} \big[2\sigma'(i)-m-1\big] \sum_{j\in I\setminus J} \operatorname{sign}(\tau(i)-\tau(j)) \\ =& \frac{1}{\binom{n}{2}(m+1)} \sum_{i\in I\cap J} \Big\{ \big[2\sigma'(i)-m-1\big] \\ & \qquad \times \big[2(\sigma(i)-\tau(i)) - k + |I\cap J|\big] \Big\}\,, \end{aligned}$$

where the last line concludes from basic deductive calculation. Similarly we have for case 3,

$$\begin{aligned} s_3 = \frac{1}{\binom{n}{2}(k+1)} \sum_{i\in I\cap J} & \Big\{ \big[2\sigma(i)-k-1\big] \\ & \times \big[2(\sigma'(i)-\tau'(i)) - m + |I\cap J|\big] \Big\}\,. \end{aligned}$$

For every item pair $\{i,j\}$ in case 4, we may assume without loss of generality that $i \in I \cap J, j \in (I \cup J)^{\complement}$. As j is absent from I (or J respectively), there are $(k+1)$ (or $(m+1)$ resp.) possible positions for j to interleave in some $\pi \in A_I$ (or $\pi' \in A_J$ resp.) and the number of positions with a lower relative order of j to i is $\sigma(i)$ (or $\sigma'(i)$ resp.). The times we get $(\pi(i)-\pi(j))(\pi'(i)-\pi'(j)) > 0$ for all possible interleaved positions of j in some $\pi \in A_I, \pi' \in A_J$ is in total $[\sigma(i)\sigma'(i) + (k+1-\sigma(i))(m+1-\sigma'(i))]$, and the times we get $(\pi(i)-\pi(j))(\pi'(i)-\pi'(j)) < 0$ is in total $[\sigma(i)(m+1-\sigma'(i)) + \sigma'(i)(k+1-\sigma(i))]$. Therefore we have

$$
\begin{aligned}
s_4 =& \sum_{\substack{i \in I \cap J \\ j \in (I \cup J)^{\complement}}} \frac{k!m!}{(n!)^2 \binom{n}{2}} \sum_{\substack{\pi \in A_I \\ \pi' \in A_J}} \operatorname{sign}(\pi(i)-\pi(j))\operatorname{sign}(\pi'(i)-\pi'(j)) \\
=& \sum_{i \in I \cap J} \frac{k!m!}{(n!)^2 \binom{n}{2}} \frac{(n!)^2}{(k+1)!(m+1)!} |(I \cup J)^{\complement}| \\
& \times \Big\{ \big[\sigma(i)\sigma'(i) + (k+1-\sigma(i))(m+1-\sigma'(i))\big] \\
& \quad - \big[\sigma(i)(m+1-\sigma'(i)) + \sigma'(i)(k+1-\sigma(i))\big] \Big\} \\
=& \frac{|(I \cup J)^{\complement}|}{\binom{n}{2}(k+1)(m+1)} \\
& \times \sum_{i \in I \cap J} \big[2\sigma(i) - k - 1\big]\big[2\sigma'(i) - m - 1\big] .
\end{aligned}
$$

For case 5, similar derivation (as case 4) with interleaving i in A_J and interleaving j in A_I leads to

$$
\begin{aligned}
s_5 =& \sum_{\substack{i \in I \setminus J \\ j \in J \setminus I}} \frac{k!m!}{(n!)^2 \binom{n}{2}} \sum_{\substack{\pi \in A_I \\ \pi' \in A_J}} \operatorname{sign}(\pi(i)-\pi(j))\operatorname{sign}(\pi'(i)-\pi'(j)) \\
=& \sum_{i \in I \setminus J} \sum_{j \in J \setminus I} \frac{k!m!}{(n!)^2 \binom{n}{2}} \frac{(n!)^2}{(k+1)!(m+1)!} \\
& \times \Big\{ \big[\sigma(i)(m+1-\sigma'(j)) + \sigma'(j)(k+1-\sigma(i))\big] \\
& \quad - \big[\sigma(i)\sigma'(j) + (k+1-\sigma(i))(m+1-\sigma'(j))\big] \Big\} \\
=& \frac{-1}{\binom{n}{2}(k+1)(m+1)} \\
& \times \sum_{i \in I \setminus J} \big[2\sigma(i) - k - 1\big] \sum_{j \in J \setminus I} \big[2\sigma'(j) - m - 1\big] .
\end{aligned}
$$

Finally $K_\tau(A_{i_1,\ldots,i_k}, A_{j_1,\ldots,j_m}) = s_1 + s_2 + s_3 + s_4 + s_5$ concludes the proof. The algorithms are summarized in Algorithm 3 for interleaving partial rankings and Algorithm 4 for top-k rankings. Note that in both algorithms, the first step is the computationally most intensive one, where we need to identify the total ranking restricted to the items present in the partial

rankings. This can be achieved by any sorting algorithm, leading the algorithms to a time complexity $O(k \log k + m \log m)$.

`R` implementation of both algorithms can be found at https://github.com/YunlongJiao/kernrank. □

Proof of Lemma 1. For any $\mathbf{x} \in \mathbb{R}^n$, note that $\|\Phi(\mathbf{x})\| \leq 1$. We can therefore apply [10, Example 6.3] to the random vector $X_j = \Phi(\tilde{\mathbf{x}}^j) - \Psi(\mathbf{x})$ that satisfies $\mathbb{E}X_j = 0$ and $\|X_j\| \leq 2$ a.s. to get, for any $u \geq 2/\sqrt{D}$,

$$\mathbb{P}\left(\|\Psi_D(\mathbf{x}) - \Psi(\mathbf{x})\| \geq u\right) \leq \exp\left(-\frac{\left(u\sqrt{D} - 2\right)^2}{8}\right).$$

We recover (a) by setting the right-hand side equal to δ and solving for u. (b) then follows by a simple union bound. □

Proof of Theorem 3. Let $\widehat{\mathbf{w}}$ be a solution to the original SVM optimization problem, and $\widehat{\mathbf{w}}_D$ a solution to the perturbed SVM, i.e., a solution of

$$\min_{\mathbf{w}} F_D(\mathbf{w}) = \frac{\lambda}{2}\|\mathbf{w}\|^2 + \widehat{R}_D(\mathbf{w}), \tag{25}$$

with $\widehat{R}_D(\mathbf{w}) = \frac{1}{m}\sum_{i=1}^m \ell(y_i \mathbf{w}^\top \Psi_D(\mathbf{x}_i))$. Since the hinge loss is 1-Lipschitz, i.e., $|\ell(a) - \ell(b)| \leq |a - b|$ for any $a, b \in \mathbb{R}$, we obtain that for any $\mathbf{u} \in \mathbb{R}^{\binom{n}{2}}$:

$$\begin{aligned}\left|\widehat{R}(\mathbf{u}) - \widehat{R}_D(\mathbf{u})\right| &\leq \frac{1}{m}\sum_{i=1}^m \left|\mathbf{u}^\top\left(\Psi(\mathbf{x}_i) - \Psi_D(\mathbf{x}_i)\right)\right| \\ &\leq \|\mathbf{u}\| \sup_{i=1,\ldots,m} \|\Psi_D(\mathbf{x}_i) - \Psi(\mathbf{x}_i)\| \,.\end{aligned} \tag{26}$$

Now, since $\widehat{\mathbf{w}}_D$ is a solution of (25), it satisfies

$$\|\widehat{\mathbf{w}}_D\| \leq \sqrt{\frac{2F_D(\widehat{\mathbf{w}}_D)}{\lambda}} \leq \sqrt{\frac{2F_D(0)}{\lambda}} = \sqrt{\frac{2}{\lambda}}\,,$$

and similarly $\|\widehat{\mathbf{w}}\| \leq \sqrt{2/\lambda}$ because $\widehat{\mathbf{w}}$ is a solution of the original SVM optimization problem. Using (26) and these bounds on $\|\widehat{\mathbf{w}}_D\|$ and $\|\widehat{\mathbf{w}}\|$, we get

$$\begin{aligned}&F(\widehat{\mathbf{w}}_D) - F(\widehat{\mathbf{w}}) \\ &\quad= F(\widehat{\mathbf{w}}_D) - F_D(\widehat{\mathbf{w}}_D) + F_D(\widehat{\mathbf{w}}_D) - F(\widehat{\mathbf{w}}) \\ &\quad\leq F(\widehat{\mathbf{w}}_D) - F_D(\widehat{\mathbf{w}}_D) + F_D(\widehat{\mathbf{w}}) - F(\widehat{\mathbf{w}}) \\ &\quad= \widehat{R}(\widehat{\mathbf{w}}_D) - \widehat{R}_D(\widehat{\mathbf{w}}_D) + \widehat{R}_D(\widehat{\mathbf{w}}) - \widehat{R}(\widehat{\mathbf{w}}) \\ &\quad\leq (\|\widehat{\mathbf{w}}_D\| + \|\widehat{\mathbf{w}}\|) \sup_{i=1,\ldots,m} \|\Psi_D(\mathbf{x}_i) - \Psi(\mathbf{x}_i)\| \\ &\quad\leq \sqrt{\frac{8}{\lambda}} \sup_{i=1,\ldots,m} \|\Psi_D(\mathbf{x}_i) - \Psi(\mathbf{x}_i)\| \,.\end{aligned}$$

Theorem 3 then follows from Lemma 1. □

Algorithm 3 Kendall kernel for two interleaving partial rankings.

Input: two partial rankings $A_{i_1,\ldots,i_k}, A_{j_1,\ldots,j_m} \subset \mathbb{S}_n$, corresponding to subsets of item indices $I := \{i_1, \ldots, i_k\}$ and $J := \{j_1, \ldots, j_m\}$.

1: Let $\sigma \in \mathbb{S}_k$ be the total ranking corresponding to the k observed items in $A_{i_1,\ldots,i_k}$, and $\sigma' \in \mathbb{S}_m$ be the total ranking corresponding to the m observed items in $A_{j_1,\ldots,j_m}$.

2: Let $\tau \in \mathbb{S}_{|I\cap J|}$ be the total ranking of the observed items indexed by $I \cap J$ in $A_{i_1,\ldots,i_k}$, and $\tau' \in \mathbb{S}_{|I\cap J|}$ the total ranking of the observed items indexed by $I \cap J$ in partial ranking $A_{j_1,\ldots,j_m}$.

3: Initialize $s_1 = s_2 = s_3 = s_4 = s_5 = 0$.

4: If $|I \cap J| \geq 2$, update

$$s_1 = \frac{\binom{|I\cap J|}{2}}{\binom{n}{2}} K_\tau(\tau, \tau').$$

5: If $|I \cap J| \geq 1$ and $|I \setminus J| \geq 1$, update

$$s_2 = \frac{1}{\binom{n}{2}(m+1)} \sum_{i\in I\cap J} \Big\{ \big[2\sigma'(i) - m - 1\big] \times \big[2(\sigma(i) - \tau(i)) - k + |I \cap J|\big] \Big\}.$$

6: If $|I \cap J| \geq 1$ and $|J \setminus I| \geq 1$, update

$$s_3 = \frac{1}{\binom{n}{2}(k+1)} \sum_{i\in I\cap J} \Big\{ \big[2\sigma(i) - k - 1\big] \times \big[2(\sigma'(i) - \tau'(i)) - m + |I \cap J|\big] \Big\}.$$

7: If $|I \cap J| \geq 1$ and $|(I \cup J)^{\complement}| \geq 1$, update

$$s_4 = \frac{|(I \cup J)^{\complement}|}{\binom{n}{2}(k+1)(m+1)} \times \sum_{i\in I\cap J} \big[2\sigma(i) - k - 1\big]\big[2\sigma'(i) - m - 1\big].$$

8: If $|I \setminus J| \geq 1$ and $|J \setminus I| \geq 1$, update

$$s_5 = \frac{-1}{\binom{n}{2}(k+1)(m+1)} \times \sum_{i\in I\setminus J} \big[2\sigma(i) - k - 1\big] \sum_{j\in J\setminus I} \big[2\sigma'(j) - m - 1\big].$$

Output: $K_\tau(A_{i_1,\ldots,i_k}, A_{j_1,\ldots,j_m}) = s_1 + s_2 + s_3 + s_4 + s_5$.

Algorithm 4 Kendall kernel for a top-k partial ranking and a top-m partial ranking.

Input: a top-k partial ranking and a top-m partial ranking $B_{i_1,\ldots,i_k}, B_{j_1,\ldots,j_m} \subset \mathbb{S}_n$, corresponding to subsets of item indices $I := \{i_1,\ldots,i_k\}$ and $J := \{j_1,\ldots,j_m\}$.

1: Let $\sigma \in \mathbb{S}_k$ be the total ranking corresponding to the k observed items in $B_{i_1,\ldots,i_k}$, and $\sigma' \in \mathbb{S}_m$ be the total ranking corresponding to the m observed items in $B_{j_1,\ldots,j_m}$.

2: Let $\tau \in \mathbb{S}_{|I\cap J|}$ be the total ranking of the observed items indexed by $I \cap J$ in $B_{i_1,\ldots,i_k}$, and $\tau' \in \mathbb{S}_{|I\cap J|}$ the total ranking of the observed items indexed by $I \cap J$ in partial ranking $B_{j_1,\ldots,j_m}$.

3: Initialize $s_1 = s_2 = s_3 = s_4 = s_5 = 0$.

4: If $|I \cap J| \geq 2$, update

$$s_1 = \frac{\binom{|I\cap J|}{2}}{\binom{n}{2}} K_\tau(\tau, \tau') \,.$$

5: If $|I \cap J| \geq 1$ and $|I \setminus J| \geq 1$, update

$$s_2 = \frac{1}{\binom{n}{2}} \sum_{i \in I\cap J} \left[2(\sigma(i) - \tau(i)) - k + |I \cap J|\right] .$$

6: If $|I \cap J| \geq 1$ and $|J \setminus I| \geq 1$, update

$$s_3 = \frac{1}{\binom{n}{2}} \sum_{i \in I\cap J} \left[2(\sigma'(i) - \tau'(i)) - m + |I \cap J|\right] .$$

7: If $|I \cap J| \geq 1$ and $|(I \cup J)^{\complement}| \geq 1$, update

$$s_4 = \frac{|I \cap J| \cdot |(I \cup J)^{\complement}|}{\binom{n}{2}} \,.$$

8: If $|I \setminus J| \geq 1$ and $|J \setminus I| \geq 1$, update

$$s_5 = \frac{-|I \setminus J| \cdot |J \setminus I|}{\binom{n}{2}} \,.$$

Output: $K_\tau(B_{i_1,\ldots,i_k}, B_{j_1,\ldots,j_m}) = s_1 + s_2 + s_3 + s_4 + s_5$.

B Stability study of k-means algorithms

Good clustering algorithms are supposed be robust to "perturbation" in data, in the sense that clusters formed by running an algorithm on bootstrap replicas of the original data should be similar. In other words, if we bootstrap the complete dataset twice and form a clustering with respect to each, the two clustering assignments should be close to each other. Note that in order to measure the similarity of two clustering assignments, we use the (adjusted) Rand index defined by the percentage of instance pairs falling in the same or in different clusters by the two assignments [33].

In Section 7.3 of the main paper, we performed clustering on the 1980 APA election data with k-means approaches including the proposed kernel k-means and several classic k-means algorithms. Under the same experimental setting, we now compare their stability performance. Specifically, for each fixed number of clusters, we repeatedly use a bootstrap replica of the dataset to search for centroids returned by running k-means algorithms, and partition the original dataset with these identified centroids. The Rand index for two such clustering assignments is computed and the computation is repeated for 100 times accounting for the random process of bootstrapping.

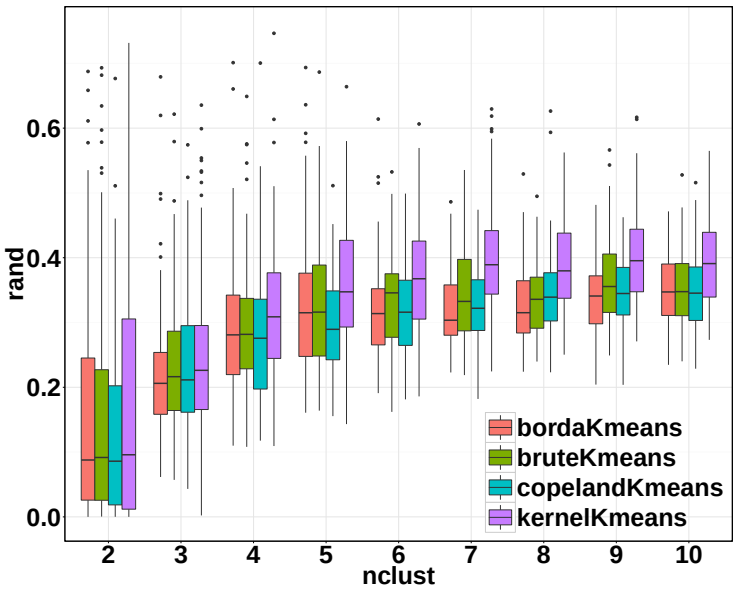


Figure 7: Across different number of clusters, Rand index between clustering assignments by running k-means algorithm on bootstrap replicas of the 1980 APA election data. For each fixed number of clusters, the boxplot represents the variance over 100 repeated runs.

Results are shown in Figure 7. We observe that, for each fixed number of clusters, kernel k-means has higher stability scores than the classic k-means algorithms in general. Notably, the discrepancy between kernel k-means and the others in terms of their stability performance is even sharper when the number of clusters becomes large. In conclusion, evidence advocates again the use of kernel k-means over classic k-means algorithms in clustering rank data.