Technical Report

eman ta zabal zazu

Universidad del País Vasco Euskal Herriko Unibertsitatea

UNIVERSITY OF THE BASQUE COUNTRY
Department of Computer Science and Artificial Intelligence

Sampling and learning the Mallows and Generalized Mallows models under the Cayley distance

Ekhine Irurozki, Borja Calvo, José A. Lozano

January, 2014

San Sebastian, Spain
www.ehu.es/ccia-kzaa
hdl.handle.net/10810/4562

Sampling and learning Mallows and Generalized Mallows models under the Cayley distance [*]

Ekhine Irurozki, Borja Calvo, Jose A. Lozano

Received: date / Accepted: date

Abstract

The Mallows and Generalized Mallows models are compact yet powerful and natural ways of representing a probability distribution over the space of permutations. In this paper we deal with the problems of sampling and learning (estimating) such distributions when the metric on permutations is the Cayley distance. We propose new methods for both operations, whose performance is shown through several experiments. We also introduce novel procedures to count and randomly generate permutations at a given Cayley distance both with and without certain structural restrictions. An application in the field of biology is given to motivate the interest of this model.

Index terms— Permutations Mallows Model Sampling Learning Cayley distance Cycle

1 Introduction

The presence of data in the form of permutations or rankings of items is ubiquitous in many real world scenarios, from the traditional applications of social and behavioral sciences [45] or card shuffling [3] to novel disciplines such as information retrieval [28], preference learning [29], bioinformatics [10] or identity management [27]. To deal with these new application fields, researchers have gone beyond classical probabilistic models over permutations [34], [11] and have proposed novel approaches. Most of these new approaches are based on extensions of Plackett-Luce [30], [40] and Mallows models [31]. In this paper we focus on the Mallows distribution and its most popular extension called Generalized Mallows Model.

The literature on Mallows and Generalized Mallows models ranges from theoretical discussion [15], [17], [19], [24], [42], [43] to more practical applications to multilabel classification [9], label ranking [8] or estimation of distribution algorithms [7].

The Mallows model can be considered as an exponential-location probability model for permutations based on distances. In this way it requires the definition of a "closeness" relation between permutations [13]. Although Mallows has been traditionally based on the Kendall distance [11], in this paper we

[*]Grants or other notes about the article that should go on the front page should be placed here. General acknowledgements should be placed at the end of the article.

consider an alternative metric, the Cayley distance. It counts the number of swaps (not necessarily adjacent) to transform a given permutation into another one. It is closely related with the cyclic structure of permutations. Therefore, while Kendall is a natural measure of dissimilarity in domains such as elections, Cayley is suited in problems dealing with swaps and the cyclic structure of permutations, which are relevant in areas such as cryptography, random number generators or computer vision [47].

In order for the Cayley based Mallows and Generalized Mallows models to become paradigms of interdisciplinary interest, the sampling and estimation (learning) operations must be carried out efficiently. Estimation of distribution algorithms [7] and label ranking [8] are good examples of applications for which a trade-off between time efficiency and accuracy in these two key operations is decisive. In this paper we focus on these two processes, sampling and learning, over both Mallows and Generalized Mallows models under Cayley distance.

In particular, the contributions of this paper include three new sampling algorithms. These sampling methods make use of novel procedures to both count the number of permutations and generate a permutation with or without certain structural restrictions, such as its Cayley distance to a given permutation. For the learning process we give an exact and a heuristic algorithm for both Mallows and Generalized Mallows model. We analyze both algorithms and show how, when the permutations in the sample are 'close enough', the exhaustive algorithm is very efficient due to its branching strategy. Moreover, the proposed heuristic algorithm is a quick alternative which reaches the optimal solution in almost every instance.

The rest of the paper is organized as follows. The next section introduces the Cayley metric as well as the Mallows and Generalized Mallows models for rankings. Section 3 describes the three proposed sampling algorithms. Section 4 deals with the problem of learning the maximum likelihood parameters in both an exact and heuristic manner. The results of the experiments designed to test the proposed methods are given in Section 5 and we conclude the paper in Section 6.

2 Cayley distance and Mallows model

Permutations are bijections of the set of integers $\{1, \ldots, n\}$ onto itself. They are usually denoted with the letters σ or π. From now on we will use the following notation, $\sigma(j) = i$ means that item i is in position j and represents the permutation σ as $\sigma = [\sigma(1)\sigma(2)\ldots\sigma(n)]$. A special permutation which is worth mentioning is the identity permutation, $e = [123\ldots n]$ which maps each item j to position j.

By composing two permutations σ and π of n items, we obtain a new permutation $\sigma\circ\pi$ such that $\sigma\circ\pi(j) = \sigma(\pi(j))$, which will be denoted as $\sigma\pi$. The result of composing a permutation σ and its inverse σ^{-1} is the identity, $\sigma\sigma^{-1} = e$ and the composition $\sigma e = e\sigma = \sigma$.

The Cayley distance $d(\sigma, \pi)$ between two permutations σ and π counts the minimum number of swaps (not necessarily adjacent) required to convert one permutation into the other. This value ranges between 0 and $n-1$ for permutations of n items. Its invariance property asserts that $d(\sigma, \pi) = d(\sigma\gamma, \pi\gamma)$ for every permutation γ. Particularly, taking $\gamma = \pi^{-1}$ and since $\pi\pi^{-1} = e$, one can write $d(\sigma, \pi) = d(\sigma\pi^{-1}, e) = d(\sigma\pi^{-1})$ which will simplify the notation.

An important concept closely related with the Cayley distance is that of a cycle of a permutation. A cycle $(i_1, i_2, \ldots, i_k)$ is a subset of the items of the permutation $\{i_1, i_2, \ldots, i_k\} \subseteq \{1, \ldots, n\}$ such that $\sigma(i_1) = i_2, \sigma(i_2) = i_3, \ldots, \sigma(i_{k-1}) = i_k, \sigma(i_k) = i_1$. As an example, consider the permutation $\pi = [3126547]$. In order to obtain the items in the first cycle, we look for the item in position 1, $\pi(1) = 3$. Then, we look for the item in position 3, $\pi(3) = 2$ and in the same way look for the next item by looking for the item in position 2, $\pi(2) = 1$. Since we have already visited position 1 then the first cycle is closed. By following this procedure, we obtain the set of four cycles that is (132), (46), (5), and (7).

The relation between the cycles of a permutation and the Cayley distance can be seen when converting a given permutation into the identity, e, with the minimum number of swaps. Note that in this case every swap involves two items that belong to the same cycle. Actually, the minimum number of swaps required to order the items of a cycle of k items is $k-1$. Let us show a minimal sequence of swaps to convert the previous example, $\pi = [3126547]$ to the identity:

Swap items 1 and 3	1326547
Swap items 2 and 3	1236547
Swap items 4 and 6	1234567

Therefore, the number of swaps to convert π into the identity permutation, and thus the Cayley distance $d(\pi)$, equals n minus the number of cycles of π. For π in the previous example the distance is $d(\pi) = 7 - 4 = 3$.

The Cayley distance $d(\pi)$ can be decomposed into a sum of $n-1$ terms, $d(\pi) = \sum_{j=1}^{n-1} X_j(\pi)$, by setting $X_j(\pi) = 0$ if j is the largest item in its cycle in π and $X_j(\pi) = 1$ otherwise. These binary variables $X_1(\pi), \ldots, X_{n-1}(\pi)$ are grouped in the vector $\mathbf{X}(\pi)$. Let us show this with an example. The Cayley distance between permutations [3421] and [1243] is computed as follows.

$$\begin{aligned} d([3421],[1243]) =& d([3421][1243]^{-1}) = d([3421][1243]) = d([3412]) \\ =& \sum_{j=1}^{4-1} X_j([3412]) = 1 + 1 + 0 = 2 \end{aligned}$$

The cycle decomposition of the permutation in this example is $\pi = [3412] = (13)(24)$. Since the largest item in the first cycle is 3, then $X_3 = 0$ and $X_1 = 1$. The variables are only defined for $1 \leq j < n$ because item n will always be the largest item in its cycle, so for the second cycle $X_2 = 1$.

It is worth noticing that although the distance $d(\pi)$ has a unique decomposition as a vector $\mathbf{X}(\pi)$, the opposite is not necessarily true. See Figure 1 for an example of two different permutations ([3412] and [2431]) that have the same decomposition.

$$\begin{aligned} \mathbf{X}([3412]) = (X_1, X_2, X_3) = (1, 1, 0) \\ \mathbf{X}([2431]) = (X_1, X_2, X_3) = (1, 1, 0) \end{aligned}$$

Figure 1: Two different permutations with the same distance decomposition

2.1 Feller coupling

The Feller coupling is used to generate permutations from the uniform distribution, [2], [21] (pag. 257). We detail the Feller coupling process before it is the basis for the later generation under MC and GMC.

This process, which is described as a sequence of n decisions, is clearly seen when the permutation is represented in cycle notation. Recall that every permutation can be written as the product of independent cycles and that if item i is placed in position j, $\sigma(j) = i$, then items i and j are in the same cycle in σ.

The Feller coupling procedure generates a permutation by constructing a collection of cycles. This generation process considers the initial cycle (1) and selects uniformly at random the item j to place at position 1 in σ, that is, select j such that $\sigma(1) = j$. If $j = 1$, then the cycle (1) is closed. Otherwise, $\sigma(1) = j$ and thus the previous cycle is updated to $(1j)$. Then, the item j' to place at $\sigma(j)$ is selected uniformly at random. If $j' = 1$, the cycle $(1j)$ is closed. Otherwise, this first cycle is now $(1jj')$. The process continues in this way until the cycle is closed, i.e. until we select item 1 to place at any position. Then, the smallest item among those that have not been inserted in the previous cycles is selected to construct another cycle in the same way, and the process is repeated until a permutation is generated, i.e., until every item $1 \leq i \leq n$ belongs to a cycle.

Note that the generative process is carried out in n separate stages. At stage s the decision can be seen as either closing the current cycle and opening a new one or either choosing one of the $n - s$ items to add to the current cycle. Note that exactly one of those items closes a cycle while the remaining $n - s$ items do not close a cycle.

In this process we can consider the previously defined $X_j(\sigma)$ binary variables as independent Bernoulli random variables of parameter $1/(n - j + 1)$.

2.2 Mallows and Generalized Mallows models

The Mallows model (MM) is an exponential-location probability model for permutations based on distances. It can be expressed as follows:

$$p(\sigma) = \frac{exp(-\theta d(\sigma, \sigma_0))}{\psi(\theta)} \tag{1}$$

where $\theta \in \mathbb{R}$ is a spread parameter, σ_0 is the location parameter called the central permutation, $d(\sigma, \sigma_0)$ represents a distance between σ and σ_0 and $\psi(\theta)$ is the normalization constant $\psi(\theta) = \sum_\sigma exp(-\theta d(\sigma, \sigma_0))$. Note that when the dispersion parameter θ is greater than 0, then σ_0 is the permutation with the largest probability mass (the mode), and the closer a permutation σ is to σ_0, the larger the $p(\sigma)$. On the other hand, with $\theta = 0$ we obtain the uniform distribution and when $\theta < 0$ then σ_0 is the anti mode, a situation not considered in this paper. We will denote the MM under the Cayley distance as MC.

Despite being realistic for many real world problems, some model characteristics of the MM are sometimes too severe to properly fit the data, such as the fact that every permutation at the same distance from σ_0 has the same probability. In this case, one can consider any of the multiple extensions of the model. Some of the most popular extensions are non-parametric models [33], infinite permutations [25], [36] and mixture models [12], [37], [39]. However, the best known is the Generalized Mallows model (GMM) [23]. Instead of a single spread parameter, it considers a vector $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_{n-1})$ of $n-1$ spread parameters, each affecting a particular position of the permutation. In order to base the GMM on a particular distance, this distance must be decomposable as the sum on $n-1$ terms, each related to a particular position of the permutation. If these terms (considered as random variables) happen to be independent under the uniform distribution, then the GMM based on such a distance can be factored into $n-1$ terms [23].

As we have already seen, the Cayley distance induces a decomposition in an $n-1$ length vector such that $d(\sigma, \sigma_0) = d(\sigma\sigma_0^{-1}) = \sum_{j=1}^{n-1} X_j(\sigma\sigma_0^{-1})$. Therefore, the GMM can be coupled with the Cayley distance and it can be written as a product of $n-1$ terms as follows:

$$p(\sigma) = \frac{\sum_{j=1}^{n-1} exp(-\theta_j X_j(\sigma\sigma_0^{-1}))}{\psi(\theta)} = \prod_{j=1}^{n-1} \frac{exp(-\theta_j X_j(\sigma\sigma_0^{-1}))}{\psi_j(\theta_j)}$$

where the normalization constants are given by the following expression:

$$\psi_j(\theta_j) = (n-j)exp(-\theta_j) + 1 \tag{2}$$

We would like to point out that these expressions are the correction of those given in [23] which includes typos. We will denote the GMM under the Cayley distance as GMC. The decomposition of the normalization constant, $\psi(\theta)$, is obtained by computing the moment generating function, as in [23], and taking into consideration that each $X_j(\pi)$ is an independent Bernoulli random variable for an u.a.r. drawn permutation π (see Section 2.1). Thus, the probability of each position j being the largest item in its cycle under the GMC is as follows:

$$p(X_j(\sigma\sigma_0^{-1}) = r) = \begin{cases} \dfrac{1}{(n-j)exp(-\theta_j)+1} & \text{if } r = 0 \text{ (the cycle is closed)} \\ \dfrac{(n-j)exp(-\theta_j)}{(n-j)exp(-\theta_j)+1} & \text{if } r = 1 \text{ (otherwise)} \end{cases} \tag{3}$$

One of the main advantages of this factorization is that the normalization constant, which was defined as a sum of $n!$ terms, can be posed as a product of $n-1$ terms. Note that since the MM is the special case of the GMM in which every θ_j has the same value, the normalization constant in Equation (2) and the decomposition in Equation (3) hold for both MC and GMC.

Although, both MM and GMM are unimodal distributions, the GMM breaks the MM constraint that two permutations at the same distance have the same probabilities. In the case of the GMM, two permutations σ and π have the same probability if they share the same decomposition vector, i.e. $\mathbf{X}(\sigma\sigma_0^{-1}) = \mathbf{X}(\pi\sigma_0^{-1})$.

3 Sampling

In this section we introduce two new sampling methods and an extension to the GMC of the Gibbs sampler proposed in [15] for the MC. The first one receives the name of Distances sampling method and can draw permutations only from the MC. It starts by drawing a distance from σ_0 using the MC and then builds a permutation at the given distance uniformly at random. The second one, called Multistage sampling algorithm, can be used to simulate both MC and GMC (recall that the MC is the particular case of the GMC in which every dispersion parameter has the same value). Based on Equation (3) it generates permutations from the GMC in an stepwise procedure. Finally we extend the Gibbs sampler proposed for the MC [15] to the GMC. We compare the algorithms in terms of computation time and accuracy.

3.1 Distances sampling method

The process of drawing a permutation σ from a given Mallows distribution is carried out by first, randomly selecting a distance between σ_0 and σ and secondly, generating σ at the given distance in an unbiased way (i.e. uniformly at random among the permutations at the given distance). The process to generate permutations at a given distance is also novel and can be useful on its own.

The key idea behind this sampling algorithm is that, under the MM, every permutation at the same distance from σ_0 is equally probable. Let us assume that the number of permutations at each possible distance d is known and denoted by c_d. Then, the probability of generating, under a MC, a permutation σ at distance d from the central permutation σ_0 is as follows:

$$p(\sigma | d(\sigma, \sigma_0) = d) \propto c_d exp(-\theta d) \tag{4}$$

Moreover, the normalization constant $\psi(\theta) = \sum_\sigma exp(-\theta d(\sigma\sigma_0^{-1}))$ can be expressed as the sum of n terms in the following way:

$$\psi(\theta) = \sum_{d=0}^{n-1} c_d exp(-\theta d) \tag{5}$$

Therefore the sampling process of σ is as follows: first, sample the distance d, where $d = d(\sigma, \sigma_0)$ using Equation (4). Secondly, pick uniformly at random a permutation π at distance d from the identity permutation e, i.e. $d(\pi) = d$. Finally, in case $\sigma_0 \neq e$ Cayley's invariance property lets us obtain $\sigma = \pi\sigma_0$, since $d = d(\pi) = d(\pi\sigma_0, \sigma_0) = d(\sigma, \sigma_0)$.

In order to be able to follow the previous process, we still need to fill two gaps: calculating the number of permutations of n items at a given distance d, c_d, and the uniform generation of a permutation at distance d from the identity permutation. The answers to both questions are related with Stirling numbers.

The Stirling number of the first kind, $S(n, k)$, counts the number of permutations of n items with k cycles. Note that, as seen before, the number of cycles is an alternative way of measuring the Cayley distance. Therefore, the number of permutations of n items at distance d, c_d, is the number of permutations of n items with $k = n - d$ cycles, i.e. $c_d = S(n, n - d)$. An intuitive explanation of the recursive method for obtaining the Stirling numbers of the first kind can be found in Section 4 of [46].

The second question, that of generating a permutation at a given distance from the identity, is closely related to that of counting the permutations at a given distance. Algorithm 1 shows a recursive procedure that generates uniformly at random one of the $S(n, k)$ permutations of n items with k cycles. This algorithm is an adaptation of that of counting permutations.

This algorithm is a recursive procedure which, in its base case, when $k = 1$, generates a permutation of n items and 1 cycle, so it uniformly at random generates one of the $(n-1)!$ possible distinct cycles of n items. In the general case, there are two options:

1. Recursively generate a permutation of $n-1$ items and $k-1$ cycles and then place item n alone in its own cycle ($\pi(n) = n$), obtaining the k-th cycle.

2. Generate a permutation of $n-1$ items and k cycles and then randomly insert item n in any of the cycles (such that $\pi(n) \neq n$).

The probability of selecting option 1 or 2 depends on the number of permutations that can be built in each of the two ways. Let us focus on the number of permutations generated in option 1, those of n items and k cycles in which item n is alone in its own cycle. Note that for each permutation of $n-1$ items and $k-1$ cycles there is exactly one way of generating a permutation of n items and k cycles, that which results from setting $\pi(n) = n$. Therefore, from $S(n,k)$ permutations of n items and k cycles, exactly $S(n-1,k-1)$ of them have $\pi(n) = n$. As a consequence, the probability of choosing the first option 1 equals $S(n-1,k-1)/S(n,k)$, while the probability of selecting option 2 equals $1 - [S(n-1,k-1)/S(n,k)] = (n-1)S(n-1,k)/S(n,k)$.

Algorithm 1: *generate_permu*(n, k)
This algorithm generates a permutation of n items with k cycles. Note that every permutation of n items with k cycles is equally probable.

```
Input: $n$, num. of items; $k$, num. of cycles
Output: $\pi$, permutation of $n$ items with $k$ cycles
if $k = 1$ then $\pi$=generate a cycle with the $n$ items; /* base case */
else
    $prob = S(n-1,k-1)/S(n,k)$;
    with probability $prob$ /* $n$ stands in a cycle alone */
        $\pi(1 \ldots n-1) = generate\_permu(n-1,k-1)$;
        $\pi(n) = n$;
    end
    otherwise /* $n$ is in a cycle with other items */
        $\pi(1 \ldots n-1) = generate\_permu(n-1,k)$;
        $ran$ = random number in the range $[1, n-1]$;
        $\pi(ran) = n$;
        $\pi(n) = ran$;
    end
end
return $\pi$;
```

Regarding the computational complexity of this algorithm, the process of sampling a distance is $O(n)$ given the Stirling numbers. Stirling numbers only need to be computed once and require time $O(n^2)$. The process of generating a random permutation given the distance is $O(n)$. Finally when $\sigma_0 \neq e$, the process of calculating $\sigma = \pi\sigma_0$ is also $O(n)$.

3.2 Multistage sampling algorithm

This novel sampling algorithm can generate permutations from both MC and GMC. Recall that Equation (3) gives the probability of position j being the largest position in a cycle in π, that is $p(X_j(\pi) = 0)$. Based on Equation (3) the Multistage sampling algorithm generates a permutation π from the GMC as a sequence of n decisions. Moreover, this process can be carried out in two different ways, by selecting the positions in increasing order or by selecting items in decreasing order. Both approaches, forwards or backwards are detailed in this section. The former generates a permutation starting from position 1 to position n. The latter, on the other hand, generates the cycles of the permutation. Each of its n iterations assigns an item to a cycle, and the items are chosen in decreasing order.

In this random generation of π from the GMC, the central permutation is the identity. If $\sigma_0 \neq e$ we obtain the final permutation σ centered around σ_0 by composing π with σ_0 as $\sigma = \pi\sigma_0 = \sigma\sigma_0^{-1}\sigma_0$.

The idea behind these processes can be adapted to count the number of distinct permutations with a given $\mathbf{X}(\pi)$ vector, see Appendix A.

Generating the permutation forwards The forwards generation of a permutation π can be done in n stages, in a similar way to the Feller coupling. At the first stage, the item to place in position 1 is selected, then the item at position 2, and so on. In general, by following this process, at stage j there is exactly one item that closes a cycle in π. The probability at stage j of that item that closes a cycle is $P(X_j(\pi) = 0)$ as given by Equation (3) while the probability of any other item is uniform, $P(X_j(\pi) = 1)/(n - j)$.

Generating the permutation backwards The backwards generation of a permutation π is performed in n stages by generating a partition of the set of n items into the disjoint cycles that form the permutation. The process iterates by selecting the items in decreasing order and placing each of them in a cycle.

The backwards generation of a permutation from a GMC can be seen as a Chinese restaurant process, [5]. The Chinese restaurant process is motivated by the sequential clustering of items. Imagine a Chinese restaurant in which there are infinitely many tables, each of infinite capacity. The tables are labeled with the set of natural numbers and ordered according to that number. The customers come sequentially. The first customer sits at table 1. The second customer either sits at the same table or goes to the first unoccupied table, i.e. table 2. In general, each customer can either sit at any of the occupied tables or sit at the first unoccupied table.

In the simile of the Chinese restaurant process and the backwards generation of a permutation randomly from a GMC, the tables are the cycles of the permutation. The customers are the items in the permutation. The process starts with item n in the first cycle. Then, with probability $p(X_{n-1}(\pi) = 0)$ item $n - 1$ is placed in a new cycle. Otherwise, with probability $p(X_{n-1}(\pi) = 1)$, it is placed in the same cycle as item n. The same operation is repeated with the next item, $n - 2$. With probability $p(X_{n-2}(\pi) = 0)$ it is placed in a new cycle, and with probability $p(X_{n-2}(\pi) = 1)$ it is placed in any of the previous cycles. In general, with probability $p(X_j(\pi) = 0)$ item j is placed alone in a new cycle. Otherwise, that is, with probability $p(X_j(\pi) = 1)$ it is placed in a cycle with the previous items. In the latter case, the cycle to place the item in must be chosen at random with probability proportional to the number of permutations that can be generated in any case. Following with the Chinese restaurant process simile, item j sits to the left of an uniformly at random chosen customer among those that are already sitting. In other words, the cycle in which item j will lay is chosen with probability proportional to the number of items in each cycle.

Intuition of the dispersion parameters Based on the backwards and forwards processes for generating permutations, these lines try to clarify the meaning of the dispersion parameters. Based on the backwards generative process it is easy to see that the larger θ_j the more likely to insert item j in a new cycle. In this way, the dispersion parameters θ_j can be considered as the measure of the similarity of item j with the items $i > j$, i.e., the larger θ_j, the more unlikely to group it in a cycle with items $i > j$.

Consider the generation of a permutation σ from a given GMC with central permutation σ_0 and dispersion parameters $\boldsymbol{\theta}$. Let θ_j have a large value with regard to the rest of the dispersion parameters. In this situation, it is likely that j is the largest item in its cycle in the generated $\sigma\sigma_0^{-1}$. This has a remarkable consequence: If we consider the minimum set of swaps to change σ_0 into σ, item j will very likely be swapped with item $j' < j$. In other words, θ_j can be seen as the weight of item j to change places with item $j' < j$. Depending on the problem, one may prefer to interpret the dispersion parameters as the strength of the item at position j to change places with the item at position $j' < j$. This can be easily carried out by inverting the permutations and working with the inverse of the permutations, σ^{-1} instead of dealing with σ.

3.3 Gibbs sampler

The Gibbs sampler is a classical algorithm based on sampling a Markov chain whose stationary distribution is the distribution of interest. It was proposed to sample the MC in [15], where the authors also prove a quick convergence to the stationary distribution. In this section we extend that algorithm to the case of GMC.

The designed process, which is detailed in Algorithm 2, proceeds as follows:

1. Generate uniformly at random a permutation σ.

2. From the current permutation σ, the Gibbs sampler generates a candidate permutation σ'. This permutation is equal to the previous one in all but two positions which have been swapped.

3. Let $\gamma = min\{1, p(\sigma')/p(\sigma)\}$. With probability γ, the algorithm accepts the candidate permutation moving the chain to the candidate permutation, $\sigma = \sigma'$, and goes back to 2. Otherwise, it discards σ' and goes back to step 2.

The initial samples are discarded (burn-in period) until the Markov chain approaches its stationary distribution and so samples from the chain are samples from the distribution of interest.

As explained, a new permutation σ' is built by swapping 2 items i and j of σ. If both items were part of the same cycle in σ, then after the swap the cycle has been split into two new cycles and each swapped item is in a different cycle in σ'. In this case the distance decreases by one unit and the chain moves to the new permutation σ'. Alternatively, if it happens that both items i and j were in different cycles in σ, then after the swap, both cycles are merged into a single one in σ'. In this case, the distance has been increased in one unit and the chain moves to σ' with probability $p(\sigma')/p(\sigma)$. Under the MC this ratio equals $exp(-\theta)$. Under the GMC, where the probability of a permutation is $p(\sigma) \propto -\sum_{j=1}^{n-1} \theta_j X_j(\sigma\sigma_0^{-1})$ the ratio equals $exp(-\theta_k)$, where k is the item such that $X_k(\sigma\sigma_0^{-1}) = 0$ and $X_k(\sigma'\sigma_0^{-1}) = 1$. Therefore, under the GMC it is not necessary to compute the entire $\mathbf{X}(\sigma'\sigma_0^{-1})$ vector, but just the $X_k(\sigma'\sigma_0^{-1})$ of the items k in the cycles of the swapped items. The computational complexity of generating each permutation is thus, $O(n)$.

Algorithm 2: Gibbs sampler algorithm

```
Input: $m$, number of samples; $n$, number of items
Output: $\mathcal{S}$, collection of permutations
$\sigma \leftarrow$ uniformly at random generate a permutation /* Knuth shuffle */
$\mathcal{S} \leftarrow \sigma$ ;
for $|\mathcal{S}| < m$ do
    for $k$=1 to $n$ do $\sigma'(k) = \sigma(k)$ ;
    $i, j \leftarrow$ random numbers in the range $\{1, \dots, n\}$, where $i \neq j$;
    $swap(\sigma'(i), \sigma'(j))$;
    $\gamma = min\{1, p(\sigma')/p(\sigma)\}$;
    with probability $\gamma$
        $\sigma = \sigma'$ ;
    end
    $\mathcal{S} \leftarrow \sigma$ ;
end
```

4 Learning

This section addresses the maximum likelihood estimation of the parameters of the distribution. For a sample of m i.i.d. permutations $\{\sigma_1, \sigma_2, \dots, \sigma_m\}$ the log likelihood of the GMM is given by

$$Ln\ \mathcal{L}(\{\sigma_1, \sigma_2, \dots, \sigma_m\}|\sigma_0, \boldsymbol{\theta}) = Ln \prod_{s=1}^{m} p(\sigma_s|\sigma_0, \boldsymbol{\theta})$$

Although the MM is a particular case of the GMM in which every θ_j has the same value, the calculation of the MLE parameters are different for each model. For this reason we will describe the MLE estimation for each model separately. Finally, we introduce two new algorithms for each model, a heuristic and an exhaustive one.

4.1 The Maximum likelihood parameter estimation

4.1.1 Mallows model

In the case of the MM, the likelihood expression is given by the following equation:

$$\begin{aligned} Ln\ \mathcal{L}(\{\sigma_1, \sigma_2, \ldots, \sigma_m\}|\sigma_0, \theta) &= Ln \prod_{s=1}^{m} \frac{exp(-\theta d(\sigma_s \sigma_0^{-1}))}{\psi(\theta)} \\ &= -\theta \sum_{s=1}^{m} d(\sigma_s \sigma_0^{-1}) - mLn\, \psi(\theta) \qquad (6) \end{aligned}$$

By looking at Equation (6), we can see that calculating the value of σ_0 that maximizes the equation is independent of θ. Therefore, the MLE estimation problem can be posed as a two-step process in which, first, the central permutation and then the dispersion parameter for the given $\hat{\sigma}_0$ are obtained. The MLE for the consensus permutation under the GMC is given by the following equation.

$$\hat{\sigma}_0 = \arg\max_{\sigma_0} \sum_{s=1}^{m} -d(\sigma_s \sigma_0^{-1}) = \arg\min_{\sigma_0} \sum_{s=1}^{m} d(\sigma_s \sigma_0^{-1}) = \arg\min_{\sigma_0} \sum_{j=1}^{n-1} \bar{X}_j$$

where $\bar{X}_j = \sum_{s=1}^{m} X_j(\sigma_s \sigma_0^{-1})/m$. Since the MLE of the central permutation is the one which minimizes the sum of the distances to the permutations in the sample, finding the MLE of σ_0 can be considered as a combinatorial optimization problem. In fact, this problem is known in the literature as the swap median problem [41]. Although it is believed to be an NP-complete problem, its computational classification is still an open question [41]. In [4] the parameterized complexity of the swap median problem is studied. The authors start by introducing the definition of *non-dirty items*, which are those items that appear in the same position (which is referred to as the dominating position) in more than half of the permutations in the sample. They also prove that a central permutation places the non-dirty items in their dominating positions. Moreover, they also give a bound on the number of dirty items for which the problem is solvable in polynomial time.

Suppose that the consensus permutation $\hat{\sigma}_0$ is known, the second and last stage of the learning process of an MM concerns the estimation of the spread parameter. The MLE for the dispersion parameter for MC is the θ that satisfies the following expression:

$$\sum_{j=1}^{n-1} \frac{j}{j + exp(\theta)} = \frac{\sum_{s=1}^{m} d(\sigma_s \hat{\sigma}_0^{-1})}{m}$$

This expression is obtained by deriving the likelihood in Equation (6) and making it equal to zero. Although there is no closed expression for θ, the solution to this equation can be easily calculated with numerical methods such as Newton-Raphson.

4.1.2 Generalized Mallows model

When a sample of permutations is to be modeled by a GMC, the likelihood expression can be given as follows:

$$
\begin{aligned}
Ln\ \mathcal{L}(\{\sigma_1,\sigma_2,\ldots,\sigma_m\}|\sigma_0,\boldsymbol{\theta}) &= Ln\prod_{i=1}^{m} p(\sigma_i) \\
&= \sum_{j=1}^{n-1}\sum_{i=1}^{m} -\theta_j X_j(\sigma_i\hat{\sigma}_0^{-1}) + \sum_{i=1}^{m}\sum_{j=1}^{n-1} Ln\ \psi_j(\theta_j) \\
&= \sum_{j=1}^{n-1}(\sum_{i=1}^{m} -\theta_j X_j(\sigma_i\hat{\sigma}_0^{-1}) + \sum_{i=1}^{m} Ln\ \psi_j(\theta_j)) \\
&= \sum_{j=1}^{n-1} -m(\theta_j\bar{X}_j + Ln\,\psi_j(\theta_j)) = \sum_{j=1}^{n-1}\mathcal{L}_j \qquad (7)
\end{aligned}
$$

where $\bar{X}_j = \sum_{i=1}^{m} X_j(\sigma_i\hat{\sigma}_0^{-1})/m$. It is worth noticing that the MLE for the central permutation may not be the same as that which minimizes the distance to the sample. Suppose that the MLE for σ_0 is known. Then, the MLE for the spread parameters are given by the following expression:

$$\hat{\theta}_j = Ln(n-j) - Ln(\bar{X}_j/(1-\bar{X}_j)) \qquad (8)$$

This expression is obtained by deriving the likelihood in Equation (7) and making it equal to zero.The GMM learning process cannot be divided into two different stages as in the MM case, and thus the MLE calculation must be carried out simultaneously for every parameter. However, Equation (7) decomposes the likelihood into $n-1$ sums, being each a function on a particular $\bar{X}_j$ and θ_j. Moreover, $\hat{\theta}_j$ is also a function of $\bar{X}_j$, see Equation (8), so the likelihood for each position can be expressed as follows:

$$\sum_{j=1}^{n-1}\mathcal{L}_j = \sum_{j=1}^{n-1} -\bar{X}_j Ln(n-j) + (n-j)^2 + \frac{(n-j)^2}{\bar{X}_j} + \bar{X}_j Ln\frac{\bar{X}_j}{1-\bar{X}_j} - m \qquad (9)$$

Thus, the GMC estimation problem is still a combinatorial optimization problem that can be written in the following way.

$$
\begin{aligned}
\hat{\sigma}_0 = \arg\max_{\sigma_0} Ln\ \mathcal{L}(\{\sigma_1,\sigma_2,\ldots,\sigma_m\}|\sigma_0,\boldsymbol{\theta}) = \arg\max_{\sigma_0}\sum_{j=1}^{n-1}\mathcal{L}_j = \\
\arg\max_{\sigma_0}\sum_{j=1}^{n-1}[-\bar{X}_j Ln(n-j) + (n-j)^2 + \frac{(n-j)^2}{\bar{X}_j} + \bar{X}_j Ln\frac{\bar{X}_j}{1-\bar{X}_j} - m]
\end{aligned}
\qquad (10)
$$

In order to give an efficient exhaustive algorithm that searches the space of permutations, it is of great importance to take into account the fact that each of these functions $\mathcal{L}_j$ is strictly increasing on X_j.

4.2 Algorithms for the MLE of the parameters

In this section, we introduce a heuristic and an exhaustive algorithm to find the MLE of the central permutation and the spread parameters for both MC and GMC. Note that the evaluation of any candidate solution σ_0 is carried out in terms of distance in the MC and in terms of likelihood in the GM,C and also that the optimal MLE for σ_0 may not be the same for both models. Both distance and likelihood can be obtained given the $\bar{\mathbf{X}}$ vector: The sum of the distances to the sample in the MC is $\sum_{s=1}^{m} d(\sigma_s\sigma_0^{-1}) = m\sum_{j=1}^{n-1}\bar{X}_j(\sigma_s\sigma_0^{-1})$ and the log likelihood expression of the GMC is given in Equation (9). In this way, these two algorithms optimize a function defined over the collection of $\bar{X}_j$ variables.

4.2.1 Heuristic search

The heuristic algorithm that we propose proceeds in two stages. First, it generates a solution in a greedy manner and then the initial solution is improved using a Variable Neighborhood Search (VNS) [38].

The greedy process, starting from an empty permutation, iterates in n steps adding at each step an item to a position in the following way.

1. Given the partial permutation of $0 \leq j < n$ items, σ_0, evaluate every candidate solution. The candidate solutions σ_0' are those partial permutations of $j+1$ items built by adding one more item to σ_0 in any of the $(n-j)^2$ possible ways.

2. Set as σ_0 the candidate solution that evaluates the best (ties are solved selecting one permutation at random).

This process is repeated until σ_0 is a complete permutation. The VNS performed afterwards is a heuristic algorithm that makes use of two separate neighborhood systems, namely the insert and the swap. The insert operator considers as neighbors all those permutations that result from inserting an item into another position. The swap operator considers as neighbors all those permutations that result from swapping two positions. The local search for both neighborhood systems selects at each iteration the best neighbor of the current one (or selects a solution uniformly at random among those with the best evaluation in case of ties). The VNS, detailed in Algorithm 3, applies alternatively each of the neighborhood systems until both are stuck at the same local optimal solution.

Algorithm 3: Variable Neighborhood Search (VNS)

```
Input: $\sigma_0$ solution obtained by the greedy process
Output: $\sigma_0$ local optima for the local search with both insert and swap
         neighborhood systems
$\sigma_0'' = \sigma_0$;
repeat
    $\sigma_0 = \sigma_0''$;
    $\sigma_0' \leftarrow$ local optimum found with the local search with the insert
    neighborhood starting from $\sigma_0$;
    $\sigma_0'' \leftarrow$ local optimum found with the local search with the swap
    neighborhood starting from $\sigma_0'$;
until $\sigma_0 == \sigma_0''$;
```

4.2.2 Exhaustive algorithm

The exhaustive algorithm implements a branch and bound search. It explores the space of partial permutations of the first j out of n items ($\sigma_0^{-1}(r) = i_r$ for $r \leq j$ and $\sigma_0^{-1}(r)$ is unknown for $r > j$). These partial permutations are generated following a deep-first search on the tree shown in Figure 2.

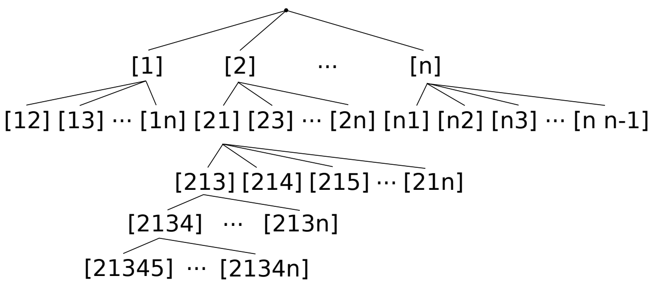

[1]
[2]
···
[n]
[12] [13] ··· [1n] [21] [23] ··· [2n] [n1] [n2] [n3] ··· [n n-1]
[213] [214] [215] ··· [21n]
[2134] ··· [213n]
[21345] ··· [2134n]

Figure 2: Permutation generating tree

The fact that the visited permutations are partial permutations of the first j-th indexes makes it possible to evaluate $\bar{X}_r$ for all $r \leq j$ and thus to partially evaluate a solution. In order to illustrate this point, let us start with $\bar{X}_1$. Given a sample permutation σ_s, $X_1(\sigma_s\sigma_0^{-1})$ can be calculated as follows:

$$X_1(\sigma_s\sigma_0^{-1}) = \begin{cases} 0 & \text{if } \sigma_s\sigma_0^{-1}(1) = 1 \\ 1 & \text{if } \sigma_s\sigma_0^{-1}(1) > 1 \end{cases}$$

In the case of $\bar{X}_2$, we can proceed similarly:

$$X_2(\sigma_s\sigma_0^{-1}) = \begin{cases} 0 & \text{if } \sigma_s\sigma_0^{-1}(2) = 2 \\ 1 & \text{if } \sigma_s\sigma_0^{-1}(2) > 2 \\ ?? & \text{if } \sigma_s\sigma_0^{-1}(2) < 2 \end{cases}$$

However, in this case if $\sigma_s\sigma_0^{-1}(2) < 2$, we need to resort to $\sigma_0^{-1}(1)$ in order to calculate that value, but because we construct the partial permutation ordered, that value has been assigned in the partial permutation. Particularly if $\sigma_0^{-1}(1) = 2$, then $X_2(\sigma_s\sigma_0^{-1}) = 0$, and if $\sigma_0^{-1}(1)$ is larger than 2, then $X_2(\sigma_s\sigma_0^{-1}) = 1$.

In general, in order to calculate $\bar{X}_j$ when all the previous indexes have been assigned, we have to proceed similarly:

$$X_j(\sigma_s\sigma_0^{-1}) = \begin{cases} 0 & \sigma_s\sigma_0^{-1}(j) = j \\ 1 & \sigma_s\sigma_0^{-1}(j) > j \\ ?? & \sigma_s\sigma_0^{-1}(j) < j \end{cases}$$

In the case of $\sigma_s\sigma_0^{-1}(j) = r < j$, we can recursively calculate $\sigma_s\sigma_0^{-1}(r)$ until we reach the first k such that $(\sigma_s\sigma_0^{-1})^k(r) = r' > j$.

Unfortunately, this simple method is extremely inefficient. However, we propose a method that, by exploring the same tree as in Figure 2, runs in $O(m)$ (where m is the number of permutations in the sample) to evaluate $\bar{X}_j$ at each node of depth j. This algorithm is based on modifying the sample of permutations at each evaluation. Briefly speaking, the evaluation at each node of depth j is carried out using the samples modified by its parent in the tree to obtain $\bar{X}_j$ and modify the samples before continuing the search. In particular, the evaluation is carried out as follows:

Evaluation The evaluation at each node requires the collection of samples modified by its parent in the tree as in Figure 2, together with the previous evaluation results $\bar{X}_r$ for $r < j$. Then, our proposed algorithm proceeds in two stages,

1. Calculate $\bar{X}_j$ as follows
$$\bar{X}_j = \sum_s X_j(\sigma_s \sigma_0^{-1})/m = \sum_s I(\sigma_s^{-1}(j) \neq \sigma_0^{-1}(j))/m \qquad (11)$$
where by $I(\cdot)$ we denote the indicator function.

2. For every permutation in the sample such that $\sigma_s^{-1}(j) \neq \sigma_0^{-1}(j)$, make the necessary swap in order to have $\sigma_0^{-1}(j) = \sigma_s^{-1}(j)$.

The fact that $X_j(\sigma_s \sigma_0^{-1}) = 0 \Leftrightarrow j$ is the largest item in its cycle in $\sigma_s \sigma_0^{-1}$ is proven in appendix **??**.

At this point, the $\bar{X}_r$ for $r \leq j$ are known and are not going to change for any node in its subtree. Although $\bar{X}_r$ for $r > j$ are not known, they can be lower bounded, as we explain in the following paragraphs. All these values of $\bar{X}_r$ for $1 \leq r < n$ are used to evaluate the current partial solution. Since $\bar{X}_r$ for $r > j$ are just lower bounds, we know that if the evaluation of the current partial solution is worse than the best solution found so far, then any complete permutation that is situated in its subtree will be even worse. In this case, the current branch can be bounded. On the other hand, if the current evaluation is better than the best evaluation found so far, continue the deep-first, branch and bound search in its subtree by passing $\bar{X}_r$ for $r \leq j$ and the modified samples to all its descendants.

Lower bound Let us now show how to obtain a lower bound for $\bar{X}_r$ for $r > j$ at a node of level j. At each node the evaluation of $\bar{X}_j$ is given as shown in Equation (11). Since for all $r < j$, $\sigma_0^{-1}(r) = \sigma_s^{-1}(r)$ then item $\sigma_0^{-1}(j)$ will be in position $r > j$ of σ_s^{-1}. Therefore, Equation (11) can be also written as follows:
$$\bar{X}_j = \sum_s \sum_{r>j} I(\sigma_s^{-1}(r) = \sigma_0^{-1}(j))/m$$
Intuitively, $\bar{X}_j$ can be seen as the proportion x/m where m is the number of permutations in the sample and x counts how many permutations σ_s^{-1} place the item in $\sigma_0^{-1}(j)$ to the right of j. Basically, this lower bound $\bar{X}_k^*$ for $k > j$ goes through the collection of samples σ_s^{-1}, counts the number of permutations that place item q (for any value of q) to the right of the k and select the minimum among them. Therefore, a simple lower bound for $\bar{X}_k$, denoted as $\bar{X}_k^*$, can be expressed as
$$\bar{X}_k^* = min \sum_s \sum_{r>k} I(\sigma_s^{-1}(r) = q)/m \quad \text{for any possible } q \qquad (12)$$
The computational complexity of evaluating this lower bound for every remaining $j < k$ is done in $O((n-k)m)$. Such an overload in the computational time is counterbalanced by the number of branches bounded by the algorithm, which results in an efficient algorithm, as will be shown in the experimental section.

In [4] it is shown that every consensus permutation places the non-dirty items in their dominating positions. This fact can be used when fitting an MC to reduce the search space since we will focus on just non-dirty items.

Example Let us illustrate our proposed exhaustive algorithm with an example. We will perform a sequence of evaluations of the branch of the tree as in Figure 2 corresponding to the set of partial permutations [2___], [24__], [241_]. In this process we will show how the X_j are computed, the samples modified and the lower bound obtained. Assume that the collection of samples is as follows:

$$\begin{array}{ll} \sigma_1 : 2134 & \sigma_1^{-1} : 2134 \\ \sigma_2 : 3241 & \sigma_2^{-1} : 4213 \\ \sigma_3 : 1342 & \sigma_3^{-1} : 1423 \end{array}$$

We first show the evaluation of the node at level 1 that corresponds to $\sigma_0^{-1} = [2\_\_\_]$. The first step is to compute $\bar{X}_1$ (the proportion of permutations in the sample such that $\sigma_s^{-1}(1) \neq \sigma_0^{-1}(1)$) as shown in Equation (11).

$$\bar{X}_1 = \sum_s I(\sigma_s^{-1}(1) \neq \sigma_0^{-1}(1))/3 = 2/3$$

Now, for the sake of efficiency of the following evaluations, we make the necessary swaps to have $\sigma_s^{-1}(1) = \sigma_0^{-1}(1)$ (remember that the current partial solution is $\sigma_0^{-1} = [2\_\_\_]$). The resulting collection, which will be used by every descendant of the current node, is as follows:

$$\begin{array}{ll} \sigma_1 : 2134 & \sigma_1^{-1} : 2134 \\ \sigma_2 : 3142 & \sigma_2^{-1} : 2413 \\ \sigma_3 : 3142 & \sigma_3^{-1} : 2413 \end{array}$$

Let us now show how the lower bound for every $r > 1$ is computed.

- The number of permutations in which item 1 is to the right of position 2 is 2.
- The number of permutations in which item 3 is to the right of position 2 is 3.
- The number of permutations in which item 4 is to the right of position 2 is 1.

Since the minimum is 1, then by using Equation (12) we get $\bar{X}_2^* = 1/3$. The same process is carried out for the lower bound $\bar{X}_3^*$.

- The number of permutations in which item 1 is to the right of position 3 is 0.
- The number of permutations in which item 3 is to the right of position 3 is 2.
- The number of permutations in which item 4 is to the right of position 3 is 1.

Since the minimum is 0, then $\bar{X}_2^* = 0$. Suppose that the evaluation of the current partial permutation $(2/3 + 1/3 + 0 = 1)$ is still better than that of the best solution found so far and so the algorithm continues the deep first search on the tree. Let the next node in level 2 be $\sigma_0^{-1} = [24\_\_]$. The evaluation of X_2 is

$$\bar{X}_2 = \sum_s I(\sigma_s^{-1}(2) \neq \sigma_0^{-1}(2))/3 = 1/3$$

and the set of samples

$\sigma_1 : 3142$	$\sigma_1^{-1} : 2413$
$\sigma_2 : 3142$	$\sigma_2^{-1} : 2413$
$\sigma_3 : 3142$	$\sigma_3^{-1} : 2413$

The computation of the lower bound is now as follows:

- The number of permutations in which item 1 is to the right of position 3 is 0.
- The number of permutations in which item 3 is to the right of position 3 is 3.

The lower bound is now $\bar{X}_3^* = 0$.

If the algorithm continues by evaluating $\sigma_0^{-1} = [241\_]$, it will result in the same collection of permutations as the one above.

Partial permutations In the context of rankings, partial permutations are generally used to consider preference information about a subset of the items. When dealing with problems relevant to the cyclic structure of arrangements the partialness can be also useful. In this sense, there can be information about certain assignments and no information about some others. Moreover, they can come as complete cycles of a subset of the items or as a partial cycles. Our proposed learning processes can be adapted to handle partial permutations by including an EM (expectation maximization) algorithm.

5 Experiments

In this section the empirical evaluation of the designed sampling and learning algorithms is presented. Moreover, the MC and GMC are statistically motivated with an experiment on real data.

5.1 Sampling MC and GMC

The next framework is designed to compare the three sampling algorithms in terms of accuracy and computational time. In this way, the three algorithms described in the previous sections are used to generate permutations of $n \in \{10, 50, 100, 150\}$ items and $\sigma_0 = [123...n]$, the identity permutation. The values of θ for MC are $\theta \in \{10^{-9}, 10^{-6}, 10^{-3}, 1\}$ while under the GMC each $\theta_j = \theta_{j-1}/2$, $j = 2, \ldots, n-1$ being $\theta_1 \in \{10^{-9}, 10^{-6}, 10^{-3}, 1\}$. For each parameter setting, each sampling process was repeated 10 times so the average results are given. Due to lack of space we only show in this paper the results of the representative selection $\theta \in \{10^{-9}, 1\}$ for MC and $\theta_1 \in \{10^{-9}, 1\}$ for GMC[1].

We show the results of generating samples of size $m \in \{100, 200, 300, \ldots, 1600\}$ permutations. Following the recommendation in [15], the burn-in process for the Gibbs algorithm discards the first $n\,log(n)$ permutations.

In order to measure the accuracy of each sampling algorithm, our first choice was to compare the likelihood of the samples. However, the differences between them were not fully appreciated due to their different sample sizes. Therefore, instead of the likelihood, the accuracy results measure the sum of the Kullback-Leibler divergences between the empirical distributions and the real distributions of the random binary variables $X_j(\sigma\sigma_0^{-1})$.

We have previously shown how to decompose the Cayley distance between two permutations σ and σ_0 into $n-1$ terms, organized in the vector $\mathbf{X}(\sigma\sigma_0^{-1})$. These binary terms $X_j(\sigma\sigma_0^{-1})$ are independent random variables whose probability distribution P_{X_j} is given by Equation (3). For every permutation σ in the generated sample, we obtain its $\mathbf{X}(\sigma\sigma_0^{-1})$ vector. The empirical distribution is $\hat{P}_{X_j}(X_j = 1) = \sum_{\sigma \in \mathcal{S}} X_j(\sigma\sigma_0^{-1})/m$ for each $1 \leq j < n$. The error of the generated samples of permutations is given as the sum of the Kullback-Leibler divergence between each empirical distribution of $X_j(\sigma\sigma_0^{-1})$ and the real one, $1 \leq j < n$, $\sum_{j=1}^{n-1} KLDiv(\hat{P}_{X_j}, P_{X_j})$.

Figures 3 and 4 summarize the time and error results of the samples generated under the MC and GMC with the previously introduced algorithms. Each figure shows the average time (x-axis) and error (y-axis) results of the simulation of a distribution having a particular n and θ (θ_1 in GMC). Each line corresponds to a particular algorithm while each marker in each line corresponds to the results of a sample of a particular size m, the results of m and $m + 100$ being joined by a line.

In Figure 3 we can see the results of the samples generated under the MC. It plots the results of sampling two distributions, one with $\theta = 10^{-9}$ and another with $\theta = 1$. It is worth noticing that the first distribution is almost uniform while there is very little consensus in the second distribution. As expected, as the sample size grows, so do the required time and the accuracy. The time consumed by the Gibbs sampler is always the smallest one and the error the largest one. The error of Gibbs with $m = 1600$ is worse than the error of the Multistage or the Distances processes with $m = 100$, despite its quick convergence to the stationary probability distribution [15]. Moreover, the difference in the accuracy increases as θ grows.

The Multistage model and the Distances method offer almost the same accuracy. Their differences are in terms of computational time, where the Distances method is faster. So, is it worth considering the Multistage sampling method? There are two reasons why the answer to this question is yes. First, the Distances algorithm uses the Stirling number of the first kind, which quickly increase as n rises, making it impossible to manage the count of permutations with standard programming libraries for values of n larger than 200. The second reason is that the Distances method can not be used to generate samples under the GMC, it can only simulate the MC, which brings us to Figure 4.

Figure 4 shows the time and accuracy results of the Gibbs sampler and the

[1]The complete results can be found on the web `http://www.sc.ehu.es/ccwbayes/members/ekhine/permus/sampling_full_results.pdf`

Multistage sampling method. Again, each figure contains data of a particular value of n and dispersion parameters $\boldsymbol{\theta}$. Each method is plotted with a particular marker symbol. A line joins the instances of size m and $m + 100$. Once more, the Gibbs sampler offers the best time results while the Multistage algorithm is the most accurate for every instance. Also, for larger values of $\boldsymbol{\theta}$ the difference between the accuracy of both methods increases.

5.2 Learning

In this section we test the performance of the algorithms for the estimation of the maximum likelihood parameters.

The parameter setting is as follows. The number of items of the permutations is $n \in \{5, \ldots, 13, 15, 20, 25\}$. The sample size is $m \in \{1000, 1500, 2000\}$. The dispersion parameter in MC is set as $\theta \in \{0.2, 0.4, 0.6, 0.8, 1, 1.5, 2, 3, 4, 5, 6, 7, 8, 9\}$. In the GMC case each θ_j can have different values. We give $\theta_1 \in \{0.2, 0.4, 0.6, 0.8,$
$1, 1.5, 2, 3, 4, 5, 6, 7, 8, 9\}$ as reference and set $\theta_j = \theta_1 - (\theta_1/2(n-2))(j-1)$ for $j > 1$, i.e. θ_1 is the largest parameter while the value of the rest decrease linearly to $\theta_{n-1} = \theta_1/2$. The central permutation is randomly chosen. For every configuration of the parameters, the experiments are repeated 10 times and their average results are shown. Due to lack of space, we only show in this paper the results of the representative selection $m = 1000$ for both MC and GMC[2]. We have not used the same parameter setting as in the previous section. Instead, this parameter setting is the same as that used in [32], where an efficient algorithm for the GMM under the Kendall's-τ distance is introduced.

In order to analyze the quality of the exhaustive algorithm, the number of partial solutions evaluated is given. Clearly, the larger the number of bounded branches, the smaller the number of solutions evaluated by the algorithm and the less time the exhaustive search will require. Moreover, in this way the results are independent of the machine. However, in order to give an intuition of the required time, we can say that the average number of nodes visited per second for every experiment is 4000. Some of the instances are not given (see for example that of $n = 20$ and $\theta = 1$ in the MC case, Figure 5a). Those instances were aborted for excessing the time limit, which was set at 15 hours for the MC and 48 hours for the GMC.

Figure 5 shows the number of evaluations when learning from each of the samples generated under different θ (X-axis) and for different values of n. Figure 5a concerns the MC learning case. Due to the efficiency of the bounding strategy, instances drawn from almost uniform distributions and large values of n can be solved. Moreover, the number of partial solutions evaluated quickly decreases as the dispersion parameter that generated the sample increases. For $\theta > 4$ all these samples were correctly solved evaluating just $n + 1$ partial permutations, although there are almost no dirty items in the samples.

Similar conclusions can be drawn when fitting a GMC (see Figure 5b). Once again, when there is consensus in the sample the exhaustive process is very efficient. For this particular setting of the θ_j parameters, one can learn the MLE for samples of permutations generated under $\theta_1 = 4$ of $n = 25$ items in less than

[2]The complete results can be found on the web `http://www.sc.ehu.es/ccwbayes/members/ekhine/permus/learning_full_results.pdf`.

5 hours. We conclude that the described algorithm happens to be an efficient learning procedure that can deal with samples of permutations up to $n = 25$ items if there is some consensus in the sample. For instances drawn from almost uniform distributions where this exhaustive algorithm is inefficient in terms of time, we propose the use of the heuristic algorithm introduced previously.

In Figure 6 the accuracy of the heuristic solutions are compared to that of the MLE for the central permutation. In the MC (Figure 6a) the quality of the heuristic solution is given as $\sum_{s=1}^{m} |d(\sigma_s, \sigma_0^+) - d(\sigma_s, \sigma_0^*)|$ where σ_0^+ and σ_0^* are the exact and heuristic solutions respectively. Conversely, in GMC (Figure 6b), the accuracy is the difference between the likelihood of σ_0^+ and σ_0^*. In both cases the heuristic algorithm is very accurate, being enough to have very little consensus (say $\theta = 1$ or $\theta_1 = 1$) to find the optimal solution. Moreover, the computational time was less than 300 ms for each of the 10 repetitions of the heuristic search.

These results show that the lower bounding technique for the branching algorithm is very efficient despite its naivety. Moreover, the computational overload is worth it due to the large number of branches bounded. For samples with no consensus, where the probability of each permutation is almost the same, and large n, the exhaustive algorithm requires a large amount of time. In such a situation we may want to sacrifice the accuracy for a quick and good solution with no guarantee of being optimal. In this situation we encourage the use of the heuristic algorithm.

5.3 Real dataset

In order to better understand the behavior of the Mallows model under the Cayley distance, we have used the model to fit a collection of permutations on the Metazoan mtDNA dataset, [6], [1]. This dataset has been used on genome rearrangements context for the construction of phylogenetic trees. By using the GMC we can obtain an interpretation on the partition and swapping structure of genes.

This dataset consists of 11 signed permutations of 36 items. Our aim is to study how genes swap and group. We are not interested in the genes turning, so we can ignore the signs. The experiment consists of fitting the GMC. As a result, we obtain the consensus permutation, σ_0 and the vector of dispersion parameters, $\boldsymbol{\theta}$. The consensus σ_0 is the permutation from which those permutations in the sample are supposed to have evolved. Figure 7 plots, for each $1 \leq j < n$ the value θ_j.

As can be noticed, there are five θ_j values that are larger than the average, those that are circled. We group those j with the largest θ_j values in the set $\mathcal{J} = \{1, 3, 6, 12, 18\}$. Several conclusions can be drawn from this fact.

Remember that the permutations in the sample, σ_s, are assumed to have evolved from σ_0 by swapping items. Subsets of the items in the permutation are cyclically swapped and items in $\mathcal{J}$ are likely to be the largest items in their cycles. This implies that if $j \in \mathcal{J}$, then the cycle of items in the generation of σ_s from σ_0 that include j is not likely to include item $j' > j$.

For every pair of items $i, j \in \mathcal{J}$ they are not likely to be part of the same cycle that transformed σ_0 in σ_s. This implies that items j and i are not likely to be swapped.

This last statement can be easily checked in this example. We consider every pair of items $1 \leq i < j \leq 36$ and count the number of times they are in the same cycle in $\sigma_s\sigma_0^{-1}$. Each pair happens to be part of the same cycle in 7.29% of the permutations. However, if we focus on the pairs of items in $\mathcal{J}$ only 1.09% of the permutations have both items in the same cycle.

The GMK (Generalized Mallows under the Kendall's-τ metric) is the most popular distribution among those related to GMC. As we have stated, GMC and GMK are not likely to fit correctly the same domains. The log likelihood is the most popular way of measuring the goodness of fit of two different probability models. We have computed the log likelihood for GMC and GMK and the result is -885.09 for the former and -961.947 for the latter. As a consequence, we can clearly state that the GMC fits this data better than the GMK.

6 Related models

In this section we explore the relations with other probabilistic models for permutations.

Mallows The original distance based model for permutations introduced by Mallows, [31], considered both Spearman's and Kendall's-τ distance. Diaconis [14] uses these Mallows models to motivate a general class of metric-based ranking models. The best known variant of these MM is the one that considers Kendall's-τ as the metric for permutations, also known as the Mallows-ϕ model. This is mainly for two reasons. First, the MM under Kendall's-τ metric has nice theoretical properties that allow the efficient computation of operations such as sampling and tractable approximations for the learning. Moreover, it can be posed as a model based on paired comparisons, [11]. The second reason for its popularity is that among the proposed metrics, it is the most natural for modeling distances in ranking domains. Note that the ranking domain is the application of permutations that has become the most popular due to the great number of commercial applications of the rankings, such as preference elicitation.

Nevertheless, in this paper we focus on the Cayley distance. The MM and GMM under Kendall's-τ and Cayley distances define very different landscapes. Their only common fact is that both are unimodal distributions centered around σ_0.

The most prominent source of difference for both models is the application domain. In Figure 8 we show the correlation for both Kendall's-τ and Cayley metrics. We consider every possible permutation $\sigma \in S_6$ (i.e., of six items). Each σ is plotted as the point (x, y) where x is the Cayley distance from σ to the identity and y is the Kendall distance from σ to the identity. Remember that the right invariant property implies that every permutation has an equal number of permutations at each possible distance. Noise has been introduced for each point for a clearer visualization of the results.

As the figure shows, Kendall's-τ and Cayley distance are not correlated. In fact, the information retained by the Kendall's-τ distance captures preference information while the Cayley distance deals with cyclic structure of the space of permutations. Therefore, a GMM under the Kendall's-τ distance will be suited for preference domains, since it is the natural metric for measuring discrepancies in voting problems. On the other hand, a GMC will be the model for dealing with problems related to the cyclic structure of permutations, which can be found in areas such as in Biology, scheduling, assignment problems, cryptography . . .

Non-null Feller coupling Recall that Fellers coupling (Section 2.1) generates a permutation as a sequence of independent cycles. This process is divided in n stages, each of which implies choosing an item that has not already been selected. At each stage s, one out of the $n-s+1$ possible items is uniformly at random selected. Exactly one of them closes a cycle. This process generates every possible permutation with equal probability.

Consider the parametric variation of the Feller coupling with parameter λ in which, at each step of the generation of σ, the item that closes the cycle and the rest do not have the same probability. In this way, we can write this generalization as follows:

$$p(X_j(\sigma) = r) = \begin{cases} \dfrac{\lambda}{\lambda + n - j} & \text{if } r = 0 \text{ (the cycle is closed)} \\ \dfrac{n-j}{\lambda + n - j} & \text{if } r = 1 \text{ (otherwise)} \end{cases} \tag{13}$$

It is clear that the non-null Feller coupling and the MC describe the same generative process.

Following this generative process, let $C^{(n)} = (C_1^{(n)}, C_2^{(n)}, \ldots, C_n^{(n)})$ be a vector where $C_i^{(n)}$ equals the number of cycles of length i in a given permutation of n items. Then, the distribution on $C^{(n)}$ under the Generalized Feller coupling is given by the Ewens Sampling Formula, $ESF(\lambda)$, for any $\mathbf{a} \in \mathbb{Z}_+^n$, [44].

$$P(C^{(n)} = a) = I(\sum_{i=1}^{n} i a_i = n) \frac{n!}{\lambda^{(n)}} \prod_{j=1}^{n} \frac{\lambda^{a_j}}{j^{a_j}} \frac{1}{a_j!}$$

where $I(\cdot)$ is the indicator function and $\lambda^{(n)} = \lambda(\lambda+1)\cdots(\lambda+n+1)$. The $ESF(\lambda)$ was introduced in [20] in the context of population genetics and arises also in areas such as Bayesian statistics. Among its many references we can highlight [2] and [18]. The asymptotic distribution of this formula and its generalizations have been largely studied [22].

Random Utility models The Latent scale model is a special case of the random utility models. This model considers the situation in which one is given a set of options $N = \{1, \ldots, n\}$ and the goal is to select a subset of those items. Therefore, the subset choice models define a probability distribution over every possible subset of the input items.

In the latent scale model, each item j has an associated probability, l_j. This probability is independent of the probabilities of the rest of the alternatives. In this way, the probability of selecting a particular subset is as follows:

$$P(X \subseteq N) = \prod_{x \in X} l_x \prod_{x \notin X} (1 - l_x)$$

Note that every possible probability distribution over the subsets that can be generated under the latent scale model can be generated by the GMC model and vice versa. Both are therefore equivalent models.

Partition models These models define a probability distribution on the space of partitions of n items. In particular, the limit of the uniform Dirichlet-multinomial partition model tends to the Ewens sampling formula, [35]. Recall that the generative process that gives rise to the Ewens sampling formula is equivalent to the MC, as stated in the previous lines.

On the other hand, the GMC can be used to define a partition model as an extension of the previous one. In this case each θ_j is related to the weight of each item to be on a group with previous seen items.

Bayesian statistics In [16] Bayesian versions of classical problems are studied, one of which is the matching problem. The goal is to study the expected number of fixed points under uniform and non-uniform priors. One of such non-null models is the MC.

In [26] a variation of the MM is introduced. This variation makes use of the cycle structure of the permutations. In particular, they define a prior distribution for the central permutation that uses an MM that assigns every permutation with the same cycle decomposition the same probability.

7 Conclusions

The present paper deals with probability distributions over the set of permutations of n different items. In particular, we consider both Mallows model (MM) and Generalized Mallows model (GMM). These models rely on a distance metric for permutations, which in our case is the Cayley distance.

Our contributions include three sampling algorithms. The first one is the Distances sampling method, which can simulate the MM. It makes use of the Stirling numbers to generate permutations directly from the distribution. It is therefore quick and accurate. We also introduce the Multistage sampling method for simulating MM and GMM. Since it samples directly from the distribution, it is as accurate as the Distances sampling algorithm. Moreover, as it is not based on the Stirling numbers it can simulate distributions over permutations of very large n. The last algorithm is an adaptation of a Gibbs sampler to the GMM. Since it does not sample directly from the distribution, it is not as accurate as the previous algorithms. However it is very fast and with a careful implementation it can simulate distributions of large n. Our results show that the MM and GMM under the Cayley distance can be efficiently sampled even for large values of n.

These algorithms make use of two novel procedures to generate permutations which can be useful by themselves. The first one generates uniformly at random permutations of n items at a given distance d. The second one generates uniformly at random permutations with a given $\mathbf{X}(\sigma)$ vector, i.e. the only available information is whether or not an item is the largest item in its respective cycle or not, but not the cycle structure. This can done by both generative processes detailed in the Multistage sampling process, the forwards and the backwards.

Moreover, we also introduce two learning algorithms, a heuristic and an exhaustive one, for both MM and GMM. Due to a branch and bound strategy, the exhaustive algorithm is very efficient, even for samples that have little consensus. The experimental section reports the results of the experiments for n up to 25. when there is no consensus in the sample or the number of items is very large, we propose a heuristic algorithm. This algorithm is able to learn the optimal parameters of the distribution when there is some consensus in the sample. When there is no consensus the obtained solution is very close to the optimal. Moreover, it is a very quick algorithm.

As future work, the authors plan working on the improvement on the bounds of the exhaustive learning algorithm as well as extending present techniques to different metrics and ranking models.

A Counting permutations consistent with X (see Section 3.2)

The process of generating a permutation uniformly at random and that of counting permutations are directly related. As we have already stated, in the forwards generation of permutations from the MC and GMC, there is exactly one way of closing a cycle at each stage j. See in Section 2.1 for the intuition and Section 3.2 for its application on the generation from a GMC. This means that the number of different permutations consistent with a given $X(\pi)$ vector is given by the following equation:

$$\text{Number of permutations } \pi \text{ consistent with } X_j(\pi) = \prod_{\forall j | X_j(\pi)=1} (n-j)$$

This work has been partially supported by the Saiotek and IT-609-13 programs (Basque Government), TIN2010-14931 (Spanish Ministry of Science and Innovation), COMBIOMED network in computational biomedicine (Carlos III Health Institute), 2011-CIEN-000060-01 (Diputación Foral de Gipuzkoa) and CRC-Biomarkers project (6-12-TK-2011-014, Diputación Foral de Bizkaia). Currently, Ekhine Irurozki holds a grant BES-2009-029143 from the Spanish Ministry of Science and Innovation.

References

[1] Raman Arora and Marina Meila. Consensus Ranking with Signed Permutations. In *AISTATS*, volume 31 of *JMLR Proceedings*, pages 117–125. JMLR.org, 2013.

[2] Richard Arratia, A D Barbour, and Simon Tavaré. *Logarithmic combinatorial structures: a probabilistic approach.* EMS Monographs in Mathematics. European Mathematical Society (EMS), Zürich, 2003.

[3] D Bayer and P Diaconis. Trailing the dovetail shuffle to its lair. *The Annals of Applied Probability*, 2(2):294–313, 1992.

[4] Nadja Betzler, Jiong Guo, Christian Komusiewicz, and Rolf Niedermeier. Average parameterization and partial kernelization for computing medians. *Journal of computer and system science*, 77(4):774–789, 2011.

[5] David M Blei, Thomas L Griffiths, and Michael I Jordan. The nested chinese restaurant process and bayesian nonparametric inference of topic hierarchies. *J. ACM*, 57(2), 2010.

[6] Guillaume Bourque and Pavel A Pevzner. Genome-scale evolution: Reconstructing gene orders in the ancestral species. *Genome Research*, 1(12):26–36, 2002.

[7] Josu Ceberio, Alexander Mendiburu, and Jose A Lozano. Introducing The Mallows Model on Estimation of Distribution Algorithms. In *International Conference on Neural Information Processing (ICONIP)*, number 23-25, Shangay, 2011.

[8] Weiwei Cheng and Eyke Hüllermeier. A New Instance-Based Label Ranking Approach Using the Mallows Model. In Wen Yu, Haibo He, and Nian Zhang, editors, *Advances in Neural Networks - ISNN*, volume 5551 of *Lecture Notes in Computer Science*, pages 707–716. Springer, 2009.

[9] Weiwei Cheng and Eyke Hullermeier. A Simple Instance-Based Approach to Multilabel Classification Using the Mallows Model. In *Workshop Proceedings of Learning from Multi-Label Data*, pages 28–38, Bled, Slovenia, 2009.

[10] Douglas Edward Critchlow. Ulam's metric. *In Encyclopedia of Statistical Sciences*, 9:379–380, 1988.

[11] Douglas Edward Critchlow, Michael A Fligner, and Joseph S Verducci. Probability Models on Rankings. *Journal of Mathematical Psychology*, 35:294–318, 1991.

[12] Angela D'Elia and Domenico Piccolo. A mixture model for preferences data analysis. *Computational Statistics &amp; Data Analysis*, 49(3):917–934, 2005.

[13] M Deza and T Huang. Metrics on permutations, a survey. *Journal of Combinatorics, Information and System Sciences*, 23:173–185, 1998.

[14] Persi Diaconis. *Group representations in probability and statistics.* Institute of Matematical Statistics, 1988.

[15] Persi Diaconis. The Markov chain Monte Carlo revolution. *Bulletin of the American Mathematical Society*, 46(2):179–205, November 2008.

[16] Persi Diaconis and Susan Holmes. A Bayesian Peek into Feller Volume I. *Sankhy: The Indian Journal of Statistics, Series A (1961-2002)*, 64(3), 2002.

[17] Persi Diaconis and Arun Ram. Analysis of systematic scan Metropolis algorithms using Iwahori-Hecke algebra techniques. *The Michigan Mathematical Journal*, 48(1):157–190, 2000.

[18] Peter Donnelly. Partition structures, {Polya} urns, the {Ewens} sampling formula, and the ages of alleles. *Theoretical Population Biology*, 30(2):271–288, 1986.

[19] Steven Neil Evans, Rudolf Grubel, and Anton Wakolbinger. Trickle-down processes and their boundaries. *Electronic Journal of Probability*, 17, January 2012.

[20] Warren J Ewens. The sampling theory of selectively neutral alleles. *Theoretical Population Biology*, 3(1):87–112, 1972.

[21] William Feller. *An Introduction to Probability Theory and Its Applications*, volume 1. Wiley, 1968.

[22] V Féray. Asymptotic behavior of some statistics in Ewens random permutations. *ArXiv e-prints*, 2012.

[23] Michael A Fligner and Joseph S Verducci. Distance based ranking models. *Journal of the Royal Statistical Society*, 48(3):359–369, 1986.

[24] Michael A Fligner and Joseph S Verducci. *Probability Models and Statistical Analyses for Ranking Data*. Springer, 1993.

[25] Alexander Gnedin and Grigori Olshanski. The two-sided infinite extension of the Mallows model for random permutations. *Advances in Applied Mathematics*, 48(5):615–639, 2012.

[26] Jayanti Gupta and Paul Damien. Conjugacy class prior distributions on metric-based ranking models. pages 433–445, 2002.

[27] Jonathan Huang, Carlos Guestrin, and Leonidas Guibas. Efficient Inference for Distributions on Permutations. *Representations*, pages 1–8.

[28] Guy Lebanon and John Lafferty. Cranking: Combining rankings using conditional probability models on permutations. In *International Conference on Machine Learning (ICML)*, pages 363–370, 2002.

[29] Tyler Lu and Craig Boutilier. Learning Mallows Models with Pairwise Preferences. *Learning*, 2011.

[30] Duncan Luce R. *Individual Choice Behavior*. Wiley, New York, 1959.

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

[32] Bhushan Mandhani and Marina Meila. Tractable Search for Learning Exponential Models of Rankings. *Journal of Machine Learning Research*, 5:392–399, 2009.

[33] Yi Mao and Guy Lebanon. Non-Parametric Modeling of Partially Ranked Data. *Journal of Machine Learning Research*, 9:2401–2429, 2008.

[34] John I Marden. *Analyzing and Modeling Rank Data*. Chapman & Hall, 1995.

[35] Peter McCullagh. Random Permutations and Partition Models. In Miodrag Lovric, editor, *International Encyclopedia of Statistical Science*, pages 1170–1177. Springer Berlin Heidelberg, 2011.

[36] Marina Meila and Le Bao. Estimation and Clustering with Infinite Rankings. In *Uncertainty in Artificial Intelligence (UAI)*, pages 393–402, Corvallis, Oregon, 2008. AUAI Press.

[37] Marina Meila and Harr Chen. Dirichlet Process Mixtures of Generalized Mallows Models. In *Uncertainty in Artificial Intelligence (UAI)*, pages 285–294, 2010.

[38] N Mladenović and P Hansen. Variable neighborhood search. *Comput. Oper. Res.*, 24(11):1097–1100, 1997.

[39] Thomas Brendan Murphy and Donal Martin. Mixtures of distance-based models for ranking data. *Computational Statistics &amp; Data Analysis*, 41(34):645–655, 2003.

[40] R L Plackett. The Analysis of Permutations. *Journal of the Royal Statistical Society*, 24(10):193–202, 1975.

[41] Vladimir Popov. Multiple genome rearrangement by swaps and by element duplications. *Theoretical computer science*, 385(1-3):115–126, 2007.

[42] Paige E. Rinker and Daniel N Rockmore. *A Mallows model for Coxeter groups and buildings.* PhD thesis, Datmouth college, Hanover, New Hampshire, 2011.

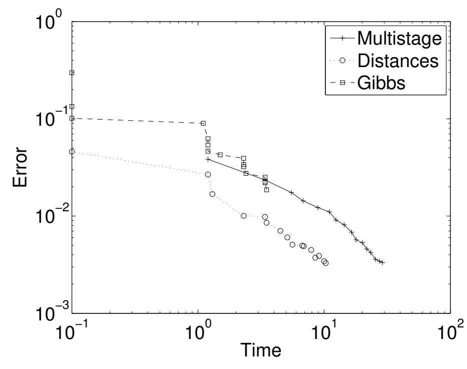
[43] Shannon Starr. Thermodynamic limit for the Mallows model on S_{n}. *Journal of Mathematical Physics*, 50(9):95208, 2009.

[44] Simon Tavaré and Warren J Ewens. *Multivariate Ewens distribution*, chapter 41, pages 232–246. Wiley, 1997.

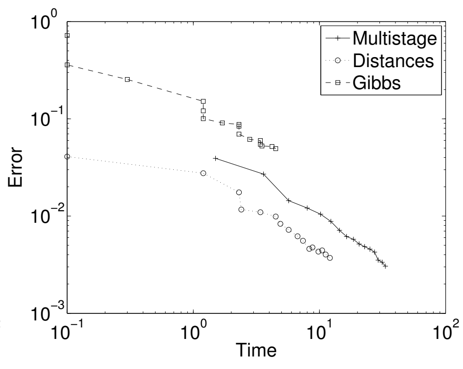
[45] L L Thurstone. A law of comparative judgment. *Psychological Review*, 34(4):273–286, 1927.

[46] Herbert S. Wilf. East Side, West Side . . . - an introduction to combinatorial families-with Maple programming. Technical report, 1999.

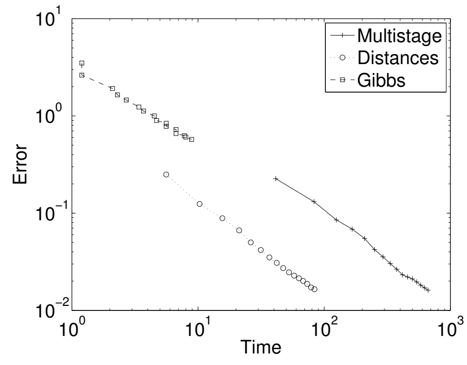
[47] Andrew Ziegler, Eric Christiansen, David Kriegman, and Serge Belongie. Locally Uniform Comparison Image Descriptor. In P Bartlett, F C N Pereira, C J C Burges, L Bottou, and K Q Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1–9. 2012.



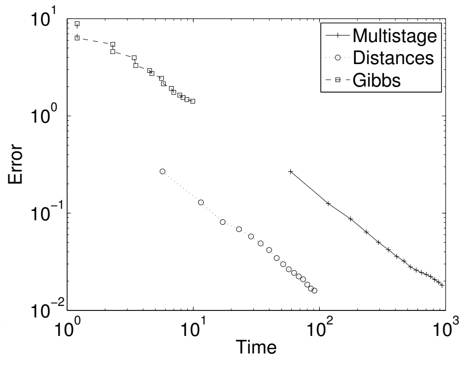
(a) $n = 10, \theta = 10^{-9}$



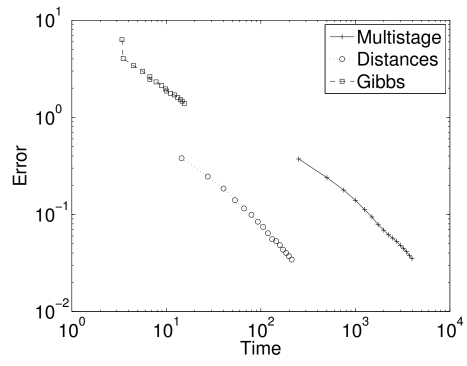
(b) $n = 10, \theta = 1$



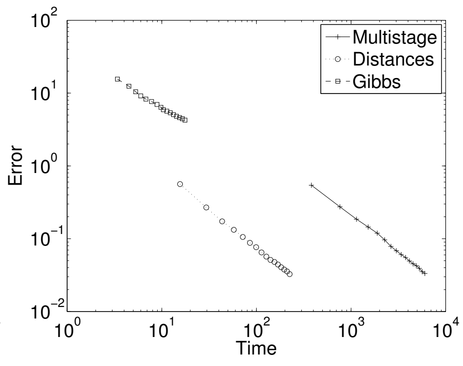
(c) $n = 50, \theta = 10^{-9}$



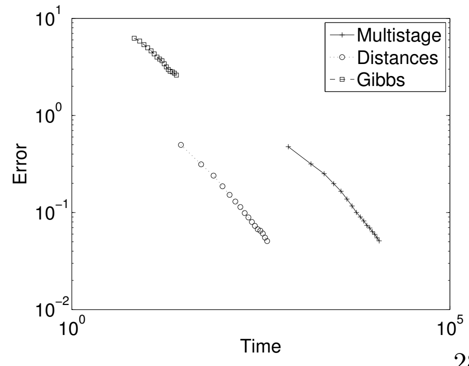
(d) $n = 50, \theta = 1$



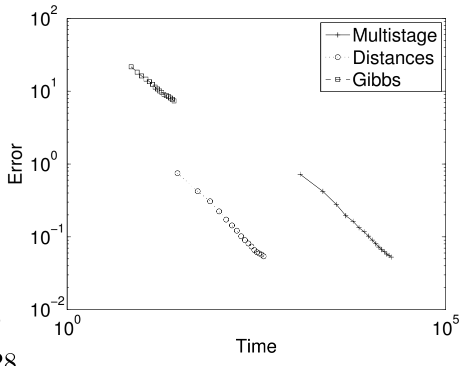
(e) $n = 100, \theta = 10^{-9}$



(f) $n = 100, \theta = 1$

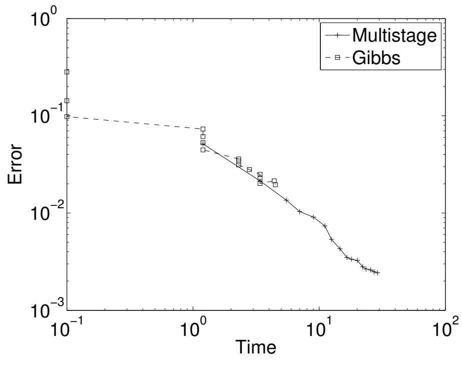


(g) $n = 150, \theta = 10^{-9}$

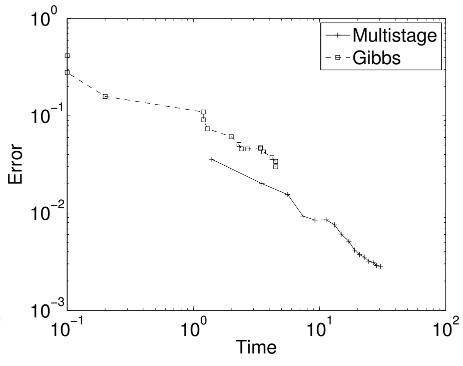


(h) $n = 150, \theta = 1$

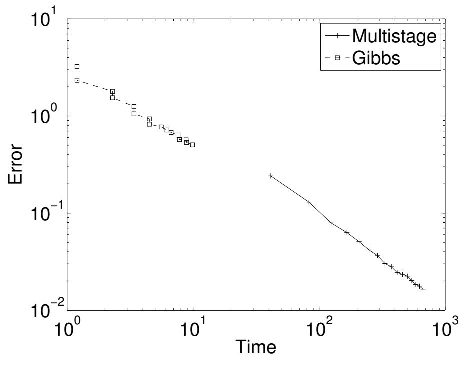
Figure 3: Time and Kullback-Leibler divergence between the generated sample and the MM for different n and θ.



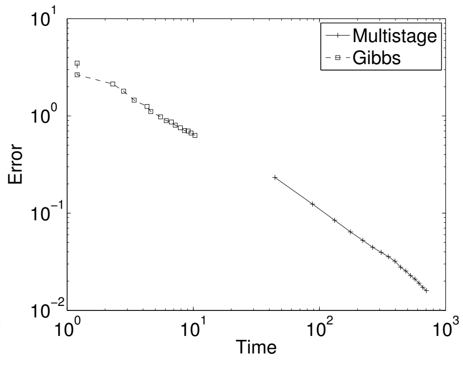
(a) $n = 10, \theta = 10^{-9}$



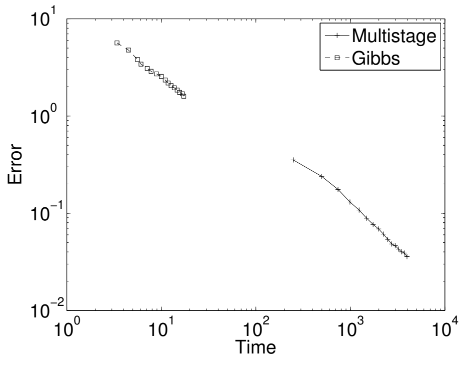
(b) $n = 10, \theta = 1$



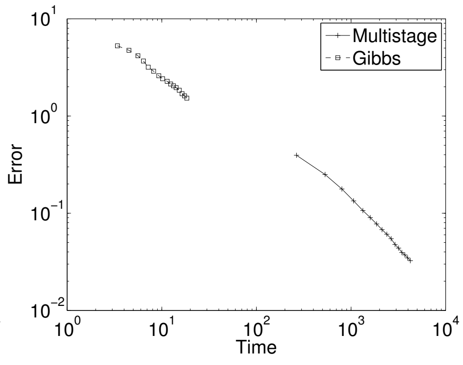
(c) $n = 50, \theta = 10^{-9}$



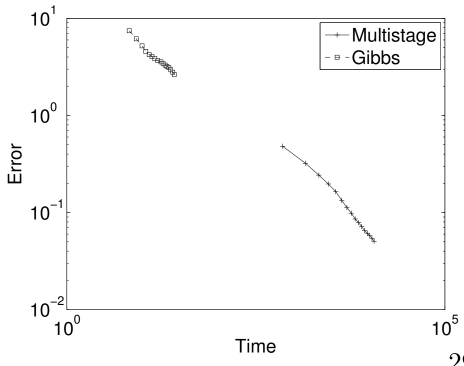
(d) $n = 50, \theta = 1$



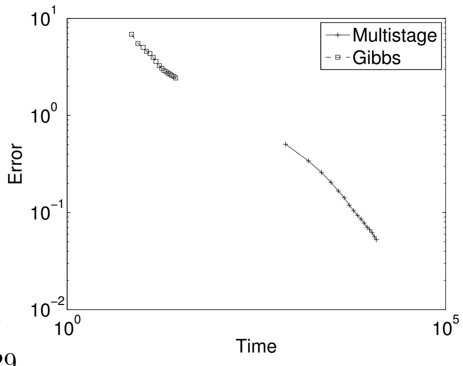
(e) $n = 100, \theta = 10^{-9}$



(f) $n = 100, \theta = 1$

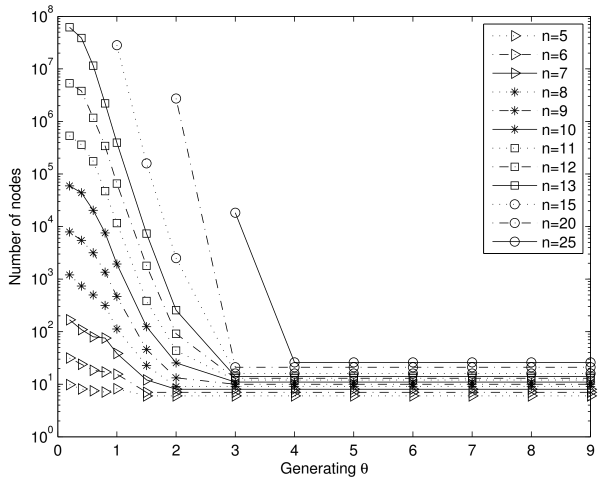


(g) $n = 150, \theta = 10^{-9}$

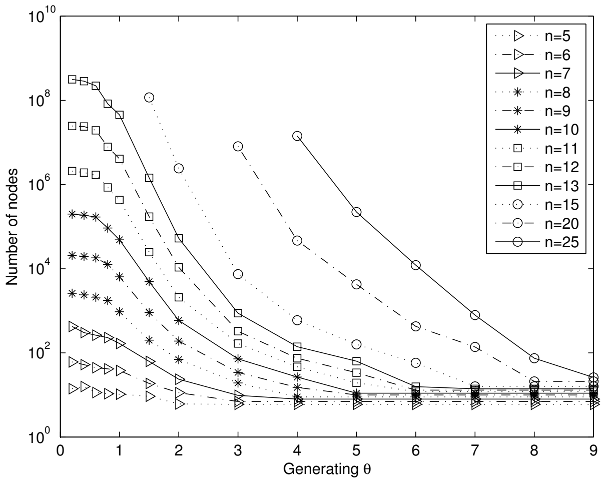


(h) $n = 150, \theta = 1$

Figure 4: Time and Kullback-Leibler divergence between the generated sample and the GMM for different n and θ_1 while $\theta_j = \theta_{j-1}/2, 2 \leq j \leq n-1$.

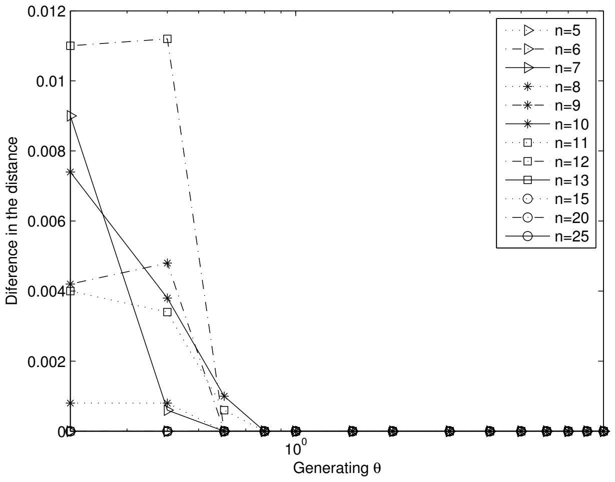


(a) MC

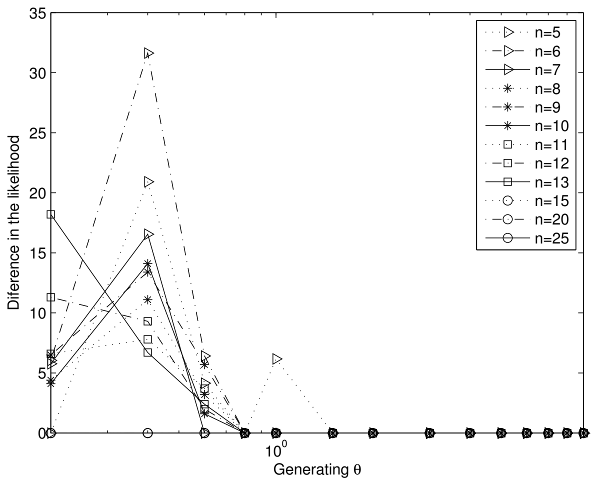


(b) GMC

Figure 5: Number of partial solutions evaluated by the exhaustive algorithm for a sample of 1000 permutations.



(a) MC, difference in the distances



(b) GMC, difference in the likelihood

Figure 6: Difference between the exhaustive and the heuristic solution.

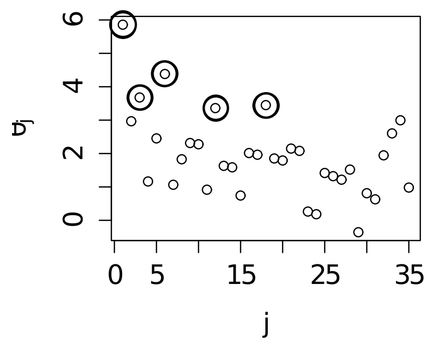


Figure 7: Dispersion parameter vector estimated for the mgr problem

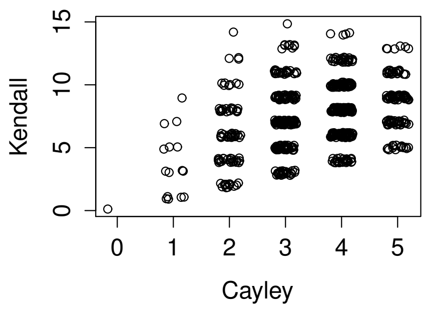


Figure 8: Correlation between Cayley and Kendall's-τ metrics fon S_6