Consensus Ranking with Signed Permutations

Marina Meila
University of Washington
Seattle WA 98195

Raman Arora
Toyota Technological Institute
Chicago, IL 60637

Abstract

Signed permutations (also known as the hyperoctahedral group) are used in modeling genome rearrangements. The algorithmic problems they raise are computationally demanding when not NP-hard. This paper presents a tractable algorithm for learning consensus ranking between signed permutations under the inversion distance. This can be extended to estimate a natural class of exponential models over the group of signed permutations. We investigate experimentally the efficiency of our algorithm for modeling data generated by random reversals.

1 Introduction

Our paper introduces the first family of statistical models over the group of signed permutations. The interest in signed permutations (SP) is intimately related to genetics; therefore, we start with the biological motivation of our work and the role SP's play in genetics.

1.1 Signed permutations in genetics

One of the key molecular evolutionary mechanisms is the rearrangement of gene order within chromosomes or genomes. In 1984, a seminal work by [Nadeau and Taylor, 1984] related humans and mice and estimated the number of reversals between human and mouse genomes. Even more remarkable was the discovery by [Palmer and Herbon, 1988] that the genomes of cabbage and turnip are almost identical in primary sequences of genes but different in the gene order which can be explained by as few as three reversals. It is also hypothesized that the rate of genome rearrangement in some species (certain plants and viruses) is much faster than the rate of point mutations of primary DNA sequences [Pevzner, 2000].

Since the pioneering works cited above, there has been a surge of interest in computational approaches for comparing gene orders between a *pair of genomes*. [Sankoff et al., 1990] formulated the problem as a combinatorial problem of sorting a permutation by reversals. In general, the problem of sorting by reversals is NP hard [Caprara, 1997] and much research has focused on approximate tractable algorithms and heuristics. However, owing to the directed structure (or orientation) of genes, the rearrangement of genomes is better modeled using a *signed permutation* and the problem of comparing gene orders of two genomes can be formulated as sorting a signed permutation [Bafna and Pevzner, 1996]. A polynomial time algorithm with good guarantees was given by Pevzner et. al. for sorting a signed permutation by reversal [Hannenhalli and Pevzner, 1999, Bafna and Pevzner, 1996].

While the problem of pairwise genome rearrangement is well understood, the problem of multiple genome rearrangement is still open. This problem can be described as inferring a phylogenetic tree which explains the rearrangements between multiple species. A key step in building such a tree is the median problem, i.e. finding a *consensus rearrangement* that best agrees with the given gene orders (a.k.a. signed permutations) of three genomes. The median problem is known to be NP hard [Caprara, 1997, 1999]. Consequently, most research has focused on building a binary phylogenetic tree [Bourque and Pevzner, 2002] using tractable algorithms for pairwise comparison of signed permutations. In this paper we address the original problem of consensus between multiple signed permutations, by giving a statistical formulation and deriving an estimation algorithm basedon this formulation. This allows one to have multiple splits at each node in a phylogenetic tree which is more plausible biologically. Furthermore, it allows us to build the tree in a top-down fashion.

Appearing in Proceedings of the 16^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2013, Scottsdale, AZ, USA. Volume 31 of JMLR: W&CP 31.

Another motivation for our work comes from the fact that genome rearrangements are common between different individuals of the same species. It has been observed that around 1 in 1000 individuals have genomic rearrangements that are asymptomatic [Pevzner, 2000]. At the same time, genome rearrangements have been associated with disorders like Down syndrome [Pevzner, 2000]. Therefore, inferring a typical rearrangement scenario within a certain population will serve as a biomarker.

1.2 The Consensus Ranking (Median) Problem

For comparing gene-orders and reconstructing genome rearrangements, it is convenient to model a genome as a signed permutation of integers $\{1, 2, \ldots, n\}$ where each gene in the genome is indexed by a unique integer and the orientation of each gene is encoded with the sign of the corresponding integer. Let $d(\sigma_1, \sigma_2)$ be a distance function that compares two signed permutations σ_1, σ_2. Then, the problem of consensus ranking with respect to the distance $d(\cdot, \cdot)$ is defined as finding a SP σ such that the sum-distance between σ and a set of given SP's $\{\sigma_1, \sigma_2, \ldots, \sigma_N\}$ is minimized: i.e. σ minimizes $\sum_{i=1}^{N} d(\sigma, \sigma_i)$ over all possible signed permutations.

1.3 Related Work

Since there is no natural metric on the signed permutation group, a lot of work has focused on defining a useful distance between two elements of the signed permutation group. Over the years, several distance functions including breakpoint distance, reversal distance, transposition distance and the double-cut-and-join (DCJ) distance have been proposed [Bader, 2011]. It has been shown that even the simplest form of the consensus ranking problem i.e. the median problem is NP-complete under all these distance functions [Bader, 2011]. Consequently, many heuristic as well as inexact algorithms have been proposed in recent years [Bourque and Pevzner, 2002, Caprara, 2003, Arndt and Tang, 2007, Swenson et al., 2009, Siepel and Moret, 2001]. However, all of these algorithms are based on pairwise distances and comparing two genomes at a time. For instance, given N genomes $\{G_1, G_2, \ldots, G_N\}$, these algorithms address the multiple genome rearrangement problem by building a binary phylogenetic tree based on all possible pairwise comparisons at each level [Tannier et al., 2009, Ma, 2011].

The problem of consensus ranking for permutations has also been studied in Information Retrieval and Natural Language Processing [A. Klementiev et al., 2008, O. Wu et al, 2011].

1.4 Our Approach

The main difference between our approach and existing methods is that we approach consensus ranking as a statistical estimation problem, in the exponential family. Thus we define a family of distributions over SP's, and formulate the median problem as Maximum Likelihood estimation in this family. We design an algorithm that can simultaneously solve the median problem (or the more general consensus ranking) for multiple signed permutations, under a surrogate distance called the *inversion distance*. We achieve this by efficiently aggregating sum-distance over a set of signed permutations into a sufficient statistic. Then we verify experimentally that the algorithm is able to solve the consensus ranking problem under the reversal distance, being thus relevant to genetics applications. The paper is organized as follows. We first present an algorithmic view of our distance function and a sufficient statistic matrix for distance between multiple genomes in Section 2. The algorithms for consensus sorting are discussed in Section 3. Section 4 discusses the general statistical model we propose, and Section 5 presents experimental results.

2 The inversion distance on W_n: an algorithmic view

Preliminaries The signed permutation group W_n, also known as the *hyper-octahedral group*, is the group of all permutations σ of $\{(-1)^k i | i = 1, \ldots, n;\ k = 1, 2\}$, such that $\sigma(-i) = -\sigma(i)$ for $i = 1, \ldots, n$. The order of the group W_n is $2^n n!$. It is the semi-direct product of the symmetric group S_n of n symbols and the cyclic group C_2 of order two. The signed permutation group W_n is generated by transpositions $r_i = (i, i+1)$, i.e. interchanging i^{th} and $(i+1)^{th}$ symbol, for $i = 1, \ldots, n-1$ and negation w_i that maps $i \mapsto -i$, for $i = 1, \ldots, n$.

Normal form for a signed permutation and bubble-sort algorithms: In this section we develop an algorithm for generating the *word* of a signed permutation in *normal form*. The normal form word for a SP π is a sequence of generators in $S = \{r_1, \ldots r_{n-1}, w_n\}$ that applied to the identity e produces π, and which has a particularly natural structure [Stanley, 1997]. Conversely, the inverse of this sequence, applied to π, will turn it into e, the identity element of the group. From our (algorithmic and statistical) standpoint we find the latter form more convenient. For permutations and signed permutations, one such normal form is obtained by iteratively bringing each item i in its correct place according to e. Algorithm NormalFormSort in Figure 1(a), inspired by [Stanley, 1997], does this.

Algorithm NormalFormSort(π)

For items $i = 1 : n$

1. if $\text{sign}(\pi(i)) = -$
   (a) move i right $n - \pi^{-1}(i)$ position to the end of the list
   (b) flip sign of i
2. move i left to rank i

Algorithm NormalFormSort2 (π^{ref})

For items $i = 1 : n$

1. move i left to rank i by adjacent transpositions
2. delete $\underline{i}$ from the list π^{ref}

Figure 1: Algorithms (a) NormalFormSort and (b) NormalFormSort2

The total number of steps taken by the NormalFormSort algorithm is the *inversion distance* between π and e, denoted $d(\pi|e)$. The distance d can be decomposed as $d = c_1 + c_2 + \dots c_n$, with c_i representing the number of steps needed to move item i in place. The vector $c(\pi) = [c_1, \dots c_n]$ uniquely defines the permutation π (which can be reconstructed from the identity by reversing the actions of algorithm NormalFormSort) and is called the *code* of π.

Our definition differs from the more common definition of the normal form word [Bjorner and Brenti, 2005] for the hyperoctahedral group. All the results of this paper can be immediately restated to agree with the [Bjorner and Brenti, 2005] definition. We prefer this form as it is more intuitive from the point of view of consensus ranking algorithms.

An alternative formulation of NormalFormSort will help us arrive at a consensus ranking algorithm. First, we introduce the *reflected form* for a signed permutation; e.g. $\pi = [4\,\underline{2}\,\underline{1}\,3]$ is given as $\pi^{ref} = [4\,\underline{2}\,\underline{1}\,3\,\underline{3}\,1\,2\,\underline{4}]$[1]. The reflected form may be interpreted as a permutation of $\mathcal{I} = [1, 2, \dots, n, -n, -n+1, \dots, -2, -1,]$ such that $\pi^{ref}_j = \pi_j$ and $\pi^{ref}_{j+n} = -\pi_j$ for $j = 1, 2, \dots, n$. Consequently, the identity permutation will correspond to $e^{ref} = [1 \dots n\,\underline{n} \dots \underline{1}]$. We will take this particular ordering imposed by e to be the "natural" ordering of the set $\mathcal{I}$ for the rest of this paper. Algorithm NormalFormSort2 in Figure 1(b) sorts the reflected form of a signed permutation. The result will be e (not e^{ref}). The proof is given in the Supplement.

Pairwise distance between signed permutations: A small modification of the above algorithms can compute the distance between two signed permutations π and π_0. Algorithm Distance described in Figure 2(a) computes the pairwise distance between signed permutations. It is easy to check that the proposed distance is right-invariant [Diaconis and Graham, 1977].

The code of π with respect to π_0, $c_j(\pi|\pi_0)$ can be defined as the number of adjacent transpositions to bring the element equal to $\pi_0(j)$ in π^{ref} in to the j'th position. Consider the example $\pi = [4\,\underline{2}\,\underline{1}\,3]$, $\pi_0 = [3\,1\,\underline{2}\,4]$.

		$\pi_0 =$	$[3\,1\,\underline{2}\,4]$	
j	$\pi_0(j)$	action	current π^{ref}	c_j
			$[4\,\underline{2}\,\underline{1}\,3 \shortmid \underline{3}\,1\,2\,\underline{4}]$	
1	3	move 3 left 3 steps, delete $\underline{3}$	$[3\,4\,\underline{2}\,\underline{1} \shortmid 1\,2\,\underline{4}]$	3
2	1	move 1 left 3 steps, delete $\underline{1}$	$[3\,1\,4\,\underline{2} \shortmid 2\,\underline{4}]$	3
3	$\underline{2}$	move $\underline{2}$ left 1 step, delete 2	$[3\,1\,\underline{2}\,4 \shortmid \underline{4}]$	1
4	4	4 already in place, delete $\underline{4}$	$[3\,1\,\underline{2}\,4]$	0

The matrix of sufficient statistics: The next and final step is to represent signed permutations, distances and codes w.r.t an arbitrary π_0 in an additive form. This is provided by the *inversion matrix* representation of a signed permutation. The inversion matrix C is a $2n \times 2n$ matrix, with rows and colums indexed by $\mathcal{I}$; $C_{ii} = 0$ for all i, and $C_{ii'} = 1$ if item i is before item i' in π^{ref} and zero otherwise. Note therefore that $C_{ii'} + C_{i'i} = 1$ for all $i \neq i'$, this case including $i' = \underline{i}$. In C, the sum of column i represents the number of items that precede i in π^{ref}. Hence, it follows that algorithm CDistance (C, π_0), in Figure 2(b), computes the code $c(\pi|\pi_0)$ and inversion distance $d(\pi|\pi_0)$.

It is easy to see now that to compute the sum of distances from a given π_0 to a set of permutations $\pi^{(1)}, \dots \pi^{(m)}$, one can perform algorithm CDistance on $\sum_{k=1:m} C(\pi^{(k)})$ where $C(\pi^{(k)})$ is the inversion matrix representation of the signed permutation $\pi^{(k)}$. Hence, consensus ranking is equivalent to minimizing the output of CDistance over all signed permutations π_0, i.e

$$min_{\pi_0 \in W_n} \text{CDistance}(\sum_{k=1}^{m} C(\pi^{(k)}), \pi_0). \qquad (1)$$

3 Algorithms for consensus sorting

Fortunately, the iterative form of algorithm CDistance is ideally suited for the optimization problem in Equation (1). For instance, one could replace Step 1 of CDistance, where $(\pi_0)_j$ is assumed to be known, with a search for i that minimizes the c_j

[1] We use the standard mathematical convention to represent the minus sign as an underline, e.g $-2 \equiv \underline{2}$.

Algorithm DISTANCE(π, π_0)
Represent π in reflected form π^{ref}
For $j = 1 : n$ ranks in π_0

1. let $i = (\pi_0)_j$ the rank j element of π_0
2. move i left in π^{ref} to rank j by adjacent transpositions
3. delete $\underline{i}$ from the list

Ouput: $d(\pi, \pi_0)$=the total number of adjacent transpositions

Algorithm CDISTANCE(C, π_0)
Input: Precedence matrix C, reference permutation π_0
For $j = 1 : n$ ranks in π_0

1. let $i = (\pi_0)_j$ the rank j element of π_0
2. set $c_j(\pi|\pi_0) = \sum_{i'} C_{i'i}$
3. delete rows and columns i and $\underline{i}$ from C

Ouput: $c(\pi|\pi_0) = (c_{1:n})$ and $d(\pi, \pi_0) = \sum_{j=1:n} c_j$

Figure 2: Algorithms (a) DISTANCE and (b) CDISTANCE

calculated in Step 2. This algorithm would greedily minimize the objective of (1) and therefore we will refer to it forthwith as GREEDY. Of course, there is no guarantee that GREEDY will find the globally optimal π_0. One can, however, obtain the optimal solution by performing an ASTAR type search on the matrix $C = \sum_{k=1}^{m} C(\pi^{(k)})$. Let

$$c_j(i_1\, i_2\, \ldots i_j) \;=\; \sum_{i' \notin \{\pm i_1, \pm i_2, \ldots \pm i_{j-1}, i_j\}} C_{i' i_j}. \qquad (2)$$

This represents the value of c_j in Step 2 of CDISTANCE for a π_0 that has $[i_1\, i_2\, \ldots i_j]$ as a prefix. The ASTAR search [Pearl, 1984] involves maintaining a priority queue Q, of all the partial solution paths $p = [i_1\, i_2\, \ldots i_j]$, sorted by their *estimated costs*. The estimated cost of a partial solution p is the sum of $c_j(p)$ and the *heuristic* $h(C, p)$ i.e an *optimistic* estimate of $\sum_{j'>j} c_{j'}$. The search proceeds by always exploring the most promising p in the priority queue. The first complete permutation π_0 that is found this way is a provably optimal solution for the optimization problem in (1)[2]. This algorithm is given in Figure 3(a).

An important observation is that, for each partial solution p, the remaining cost will depend only on the rows and columns of C indexed by $i \notin \pm p$ where $\pm p = \{\pm i_1, \ldots \pm i_j\}$. Conceptually, at each node in the search tree, a submatrix of C obtained by deleting the rows and columns corresponding to the elements in p and their negations, is passed down. Thus, the heuristic $h(C, p)$ needs only be in fact a function $h(C_{\setminus \pm p, \setminus \pm p})$. The trivial heuristic $h \equiv 0$ is always available, but now we describe a non-trivial simple heuristic that can be very effective.

3.1 An admissible heuristic

Any ASTAR type algorithm is exact, provided the heuristic h is a lower bound to the cost-to-go [Pearl, 1984]. However, the running time of the algorithm is proportional to the number of nodes expanded in the search. The total number of terminal nodes equals the number of elements in W_n. Therefore, it is prohibitively large for all but the smallest values of n to exhaustively search all nodes. The hope is that the heuristic h will help prune many partial solutions. In this sense, the trivial heuristic is the worst possible heuristic (although even with it, some partial solutions are pruned).

Here we propose the following heuristic. Assume w.l.o.g that $j = 1$; it is also convenient to consider ${\pi_0}^{ref}$ as a permutation over the whole set $\mathcal{I}$. For any two items i, i' with $i' \neq \pm i$ we have that exactly two of $C_{ii'}, C_{i'i}, C_{\underline{i}i'}, C_{\underline{i},\underline{i'}}, \ldots$ will contribute to the total cost CDISTANCE(π_0). For instance, if $\underline{i} \prec_{\pi_0} i'$, then the cost contains the terms $C_{i'\underline{i}}, C_{\underline{i'}i}$. We now consider the matrix $\tilde{C}$ defined by $\tilde{C}_{ii'} = \min(C_{ii'}, C_{i'i})$. We have trivially that $\tilde{C}_{ii'} \leq C_{i\underline{i'}}$ for all i, i', and that $\tilde{C}$ has several symmetries, i.e $\tilde{C}_{ii'} = \tilde{C}_{i'i} = \tilde{C}_{i'\underline{i}} = C_{\underline{i'}i}$, and consequently for any set $A \subseteq \mathcal{I}$ symmetric (i.e. $-A = A$), $\sum_{i \in A} \tilde{C}_{ii'} = \sum_{i \in A} \tilde{C}_{i\underline{i'}}$. From these observations, we have:

Theorem 1. *For any* $\pi_0 \in W_n$, CDISTANCE$(C, \pi_0) \geq h(C) = \sum_{i=1}^{n} \sum_{\underline{i} < i' < i} \tilde{C}_{i'i}$.

The function $h(C)$ defined above is the heuristic we propose. Theorem 1 implies that it is an admissible one and therefore the ASTAR algorithm will never find a suboptimal solution. As $\tilde{C}$ can be pre-computed, h is also very efficient to evaluate.

3.2 Computational issues

It may appear at first glance that creating a new node in ASTAR requires $\mathcal{O}(n)$ operations or more. Here we show that the ASTAR algorithm can be implemented with a constant number of operations per node, for all $j > 1$. Let $p = [i_1, \ldots, i_j]$, $p' = [i_1, \ldots, i_j, i]$ as above, and $p'' = [i_1, i_2, \ldots, i_{j-1}, i]$ a sibling of p, representing a path of length j. Then $c_{j+1}(p') = \sum_{i' \notin \mathcal{I} \setminus p \setminus \{i\}} C_{i'i} = \sum_{i' \notin \mathcal{I} \setminus p'' \setminus \{i_j\}} C_{i'i} = c_j(p'') - C_{i_j i} - C_{\underline{i_j} i}$. Since node p''

[2] Although it may not be the only one.

Algorithm ASTAR(C)

Input: Inversion matrix C, heuristic function $h(C,p)$

Initialize $p_\emptyset \leftarrow (j=0, c=0, h(C,\emptyset), l=h(C,\emptyset))$ the empty path. $Q \leftarrow p_\emptyset$

Extract $p=[i_1\, i_2\, \dots i_j]$ the top of Q
While $j<n$ do // *(Expand node p)*

- for $i \notin \{\pm i_1, \pm i_2, \dots \pm i_j\}$
  1. create node $p'=[i_1\, i_2\, \dots i_j\, i]$
  2. $c(p') \leftarrow c(p) + c_j(p')$
  3. calculate $h(C,p')$; $l(p') \leftarrow c(p') + h(C,p')$
  4. store node p' in Q

Output: $p \equiv \pi_0$, $c(p) \equiv$ CDISTANCE(C, π_0)

Algorithm ASTARGMM(C)

Input: Inversion matrix C, heuristic function $h(C,p)$

Initialize $p_\emptyset \leftarrow (j=0, c=0, cost=0, h(C,\emptyset), l=h(C,\emptyset))$ the empty path. $Q \leftarrow p_\emptyset$
Extract $p=[i_1\, i_2\, \dots i_j]$ the top of Q
While $j<n$ do // *(Expand node p)*

- for $i \notin \{\pm i_1, \pm i_2, \dots \pm i_j\}$
  1. create node $p'=[i_1\, i_2\, \dots i_j\, i]$
  2. using $c_j(p')$, estimate $\theta_j(p')$ and $cost_j(p') = \theta_j(p')c_j(p') + \ln Z_{n-j}(\theta_j(p'))$
  3. $cost(p') \leftarrow cost(p) + cost_j(p')$, calculate $h(C,p')$, $l(p') \leftarrow cost(p') + h(C,p')$
  4. store node p' in Q

Output: $p \equiv \pi_0$, $\bar\theta(p)$, $cost(p) \equiv L(\bar\theta, \pi_0)$

Figure 3: Algorithms (a) ASTAR and (b) ASTARGMM. For ASTAR, each node in Q stores the path p, the path length j, the cost $c(p)$ and the heuristic $h(C,p)$. The queue Q is prioritized by $l(p) = c(p) + h(C,p)$. For ASTARGMM, the log-likelihood of partial solution p is denoted by $cost(p)$ and is also stored at each node.

is created before p', it means that we can calculate the sum representing $c_{j+1}(p')$ in constant time, for all but the $j=1$ nodes. A similar strategy allows to calculate $h(C,p')$ in constant time, using values for previously generated nodes.

4 Generalization to GMM type models

Exponential models based on d. The consensus ranking π_0 can be seen as the "median" of the data $\pi^{(1)} \dots \pi^{(m)}$, and, as such, a "summary" of the data set. But there are other useful summaries of the data, like for instance the dispersion around π_0. Going further in this direction, one could summarize the data by fitting a *generative model*. There is one statistical model naturally associated with the inversion distance. This is the exponential model with *central permutation* π_0 and *concentration parameter* θ given by

$$P_{\pi_0,\theta}(\pi) \;=\; \frac{1}{Z(\theta)} e^{-\theta d(\pi|\pi_0)}, \quad \theta \geq 0. \tag{3}$$

In the above, $Z(\theta)$ is a normalization constant that will be discussed shortly. The model (3) is the analog for signed permutations of the Laplace distribution. Its mode is at $\pi=\pi_0$; P decays exponentially with the distance to π_0; $P_{\pi_0,0}$ is the uniform distribution over W_n. For larger θ, the distribution concentrates around π_0. A similar model was introduced by [Mallows, 1957] for unsigned permutations; therefore, distributions of this type over permutations are known as *Mallows Models* and we will extend this terminology to the newly defined model. The Mallows Model has a useful generalization, called by [Fligner and Verducci, 1986] the Generalized Mallows Model (GMM), which assigns a separate concentration parameter θ_j to each component $c_j(\pi|\pi_0)$ of the code of π. Since c_j is associated to the j-th element of π_0, this amounts to imposing different concentrations on the different ranks. For instance, the top ranks of π_0 may have large θ_j's, thus being affected by low levels of noise, while the bottom ranks, subject to more noise, could be assigned smaller θ_j's,

$$P_{\pi_0,\bar\theta}(\pi) = \frac{1}{Z(\theta)} e^{-\sum_{j=1}^n \theta_j c_j(\pi|\pi_0)}, \tag{4}$$

where $\bar\theta = [\theta_1 \dots \theta_n]$, $\theta_j \geq 0$, $Z(\theta) = \prod_{j=1}^n Z_{n-j}(\theta_j)$.

The normalization constant Z. In (4) it is easy to see that the normalization constant Z is computable in closed form. Indeed, Z_{n-j} is the sum of a finite geometric series, i.e. $Z_{n-j}(\theta_j) = \sum_{r=0}^{2(n-j)+1} e^{-\theta_j r} = \frac{1-e^{-\theta_j 2(n-j+1)}}{1-e^{-\theta_j}}$. This is a function of of a single variable, $n-j$, which motivates our notation. It immediately follows that for the single parameter model of (3), $Z(\theta) = \prod_{j=1}^n Z_{n-j}(\theta)$.

Estimation of the $\bar\theta$ or θ parameters. For a given π_0, (4) is an *exponential family* model with parameters $\bar\theta$. Thus, the log-likelihood given a data set $\{\pi^{(1)} \dots \pi^{(m)}\}$ will be a concave function in $\bar\theta$.

$$\begin{aligned} L(\bar\theta, \pi_0) \;&=\; \frac{1}{m}\sum_{k=1}^m \ln P_{\pi_0,\bar\theta}(\pi^{(k)}) \\ &=\; -\sum_{j=1}^n \Big[\theta_j \overbrace{\frac{1}{m}\sum_{k=1}^m c_j(\pi^{(k)}|\pi_0)}^{\bar c_j(\pi_0)} + \ln Z_{n-j}(\theta_j)\Big]. \end{aligned}$$

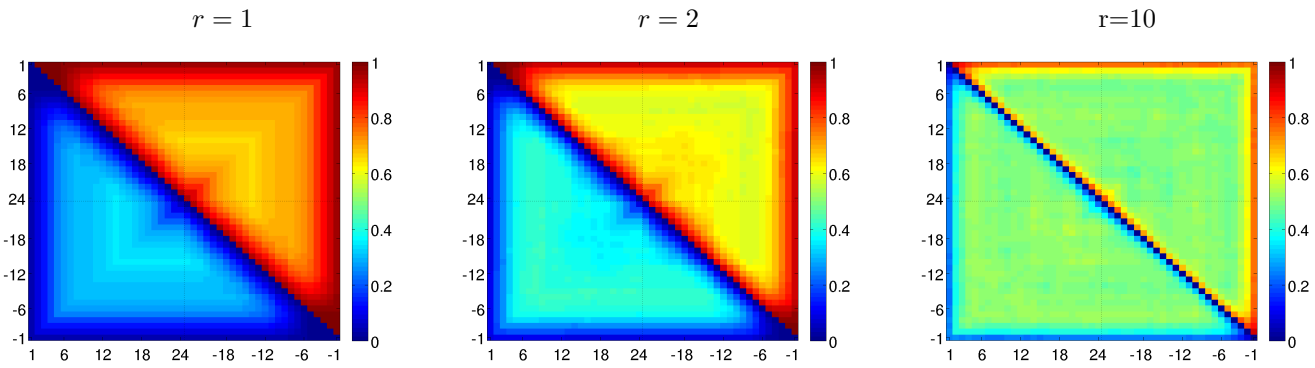

Figure 4: C matrices generated by random reversals with sample size $N = 1000$, for the permutation group of order $n = 24$; $C_{ii'} = Pr[i \prec i']$. The variable r represents the number of reversals.

Moreover, L decomposes into a sum of terms each depending on a single θ_j. Optimizing each term separately can be done by solving (numerically) the implicit equation $\bar{c}_j = g(n'_j, \theta_j)$ where $n'_j = 2(n-j)$ and $g(n'_j, \theta) = -\frac{\partial Z_{n-j}(\theta)}{\partial \theta} \frac{1}{Z_{n-j}(\theta)}$. This is typical of exponential family models. It also follows that estimating θ in the single parameter model amounts to solving for θ in $\sum_j c_j = \sum_j g(n'_j, \theta)$. The exact form of $g(n', \theta)$ is given in the Appendix.

Estimation of the central permutation π_0. We now turn to estimating the discrete parameter π_0 of the Mallows model. We start with the simpler case of the single parameter model. The log-likelihood is given by

$$L(\pi_0, \theta) \;=\; -\theta \frac{1}{m} \sum_{k=1}^{m} d(\pi^{(k)}|\pi_0) - \sum_{j=1}^{n} \ln Z_{n-j}(\theta). \quad (5)$$

The log-likelihood is maximized when the term $\frac{1}{m}\sum_{k=1}^{m} d(\pi^{(k)}|\pi_0)$ is minimized w.r.t π_0, in other words when π_0 is the solution of the consensus ranking problem. Thus, finding the Maximum Likelihood model $P_{\pi_0,\theta}$ for a sample can be done by first running the AStar algorithm to determine π_0, then estimating θ numerically as shown above.

For the GMM in (4), the estimation of π_0 and $\bar{\theta}$ does not decouple. However, we note that the θ_j estimate depends on π_0 only via the statistic $\bar{c}_j(\pi|\pi_0)$. As shown in Section 3, during the running of AStar, $\bar{c}_j(\pi|\pi_0)$ is computed at each node (i.e. partial solution) as a function of data available at that node. Thus, all we have to do is to perform the estimation of θ_j every time a node at depth j is created. The modified algorithm is AStarGMM in Figure 3(b). Since the numerical estimation of θ_j is a constant time operation, this adds only a constant time per node. Another change from the original AStar is in the cost function. The cost is now the log-likelihood, and not merely the $c_j(p)$ value. Again, this adds only a small constant time per node. However, there is a more subtle change: the admissible heuristic for ASTAR introduced in Section 3.1 is no longer guaranteed to be admissible for AStarGMM. Finding non-trivial admissible heuristics for this case is a matter of further research.

5 Experimental results

In this section we evaluate empirically the real time and memory requirements of the AStar algorithm, and we gauge its appropriateness for solving consensus problems in the more biologically relevant reversal distance.

Results on synthetic data: We generate the synthetic data not from the "true" underlying GMM model, but from the biologically motivated *random reversal* model. A *reversal* $r_{[a,b]}$ on π replaces $(\pi_a, \pi_{a+1}, \ldots, \pi_b)$ with $(\underline{\pi_b}, \ldots, \underline{\pi_{a+1}}, \underline{\pi_a})$. E.g. $r_{[3,5]}(e) = [1\,2\,\underline{5}\,\underline{4}\,\underline{3}\,6]$ in W_6. The *reversal distance* between π_1 and π_2 equals the minimum number of reversals needed to turn π_1 into π_2.

We sample SP's according to a $P^r_{\pi_0}(\pi)$, in which each π is generated by a sequence of r random reversals starting from e; the reversals have random length $l = b - a + 1$, with $l - 1 \sim$ (truncated) Poisson$(\sqrt{n})$, and the location is uniform given l. Figure 4 illustrates such distributions by presenting their inversion matrices C. Note that a single reversal can induce multiple transpositions.

The inversion matrices in Figure 4 show that, even though every individual π is far from e in inversion distance the true central permutation e is visibly the optimal π_0 w.r.t (1). This supports our conjecture that consensus ranking can be a viable alternative for studying biological data.

We simulated synthetic data, as described above, for

		$n = 24$						$n = 50$					
r	N	Objective			Distance			Objective			Distance		
		ASTAR	GREEDY	RAND	ASTAR	GREEDY	RAND	ASTAR	GREEDY	RAND	ASTAR	GREEDY	RAND
1	50	125.0	125.6	370.1	0	1.2	135	372.4	383.5	1684.3	0	17.1	612.8
1	100	120.8	129.0	370.1	0	16.5	134.7	363.4	414.0	1668.8	0	77.1	636.2
1	1000	125.5	125.5	365.4	0	0	140.7	370.3	370.3	1674.3	0	0	627.6
1	2000	119.1	129.9	362.0	0	25.2	136.9	382.8	455.1	1699.8	0	116.7	622.4
2	50	168.8	170.1	338.5	0	4.4	139.3	601.5	619.6	1565.4	0	39.5	619.0
2	100	175.4	186.1	336.7	0	43.3	153.4	613.0	676.3	1555.7	0	147.2	623.2
2	1000	174.5	175.0	337.7	0	1.5	146.4	601.5	613.5	1557.8	0	27	596.1
2	2000	171.4	182.5	340.2	9	47.3	149.4	595.0	666.6	1536.4	0	164	619.6
3	50	203.0	205.6	325.6	0	15.3	143.2	746.6	772.8	1480.8	0	76.6	608.4
3	100	198.1	206.4	330.1	21.1	57.1	135.7	739.5	798.8	1485.9	0	209.4	624.0
3	1000	202.9	205.3	326.0	0	14.3	125.5	748.2	768.8	1474.7	0	64.3	633.3
3	2000	201.1	210.7	324.7	49.4	94.5	132.6	744.2	806.1	1480.1	0	224.1	585.3

Table 1: Experimental results on artificial data generated by *random reversals*. For each algorithm, we give the objective value in (1) and the distance $\text{DISTANCE}(\hat{\pi}, \pi_0)$ between the estimated median permutation $\hat{\pi}$ and the true median permutation π_0. Both the objective and the distance are averaged over 10 runs.

N	100	1000	100	1000	100	1000
r	1	1	2	2	3	3
$n = 50$	3.5	3.5	3.4	3.4	3.4	3.4
$n = 24$	2.25	3	3	3	5	3

Table 2: We give the median of ratio of runtimes ASTAR /Greedy over 10 runs for each setting.

various sample sizes and various degrees of arrangements (the order of the permutation group and the number of reversals). We ran ASTAR and recorded the objective obtained and the quality of the median w.r.t. the true median in Table 1, and the running time in Table 2. For comparison, we also tested GREEDY and the "strawman" RAND, which choses the best out of 100 randomly sampled π_0's.

As Figure 4 shows, the difficulty of the problem increases with the order n of the permutation group, with the number of reversals r, and (not shown) with decreasing sample size N. This is reflected strongly in the running times of ASTAR, which are of the same order as GREEDY for large N, but can become large at small N. Remarkably, the metric $\text{DISTANCE}(\hat{\pi}, \pi_0)$ suggests that even for $n = 50$, the search is exact. The greedy algorithm, while appealing for its speed and simplicity, has significantly worse performance than the ASTAR search.

Results on real data: We tested algorithm ASTAR on the Metazoan mtDNA dataset [Bourque and Pevzner, 2002] comprising 11 genomes with 36 genes. We construct a phylogenetic tree (shown in Fig. 5) in a recursive top-down fashion by solving a consensus ranking problem at each node in the tree. Each internal node represents an unobserved or possibly extinct ancestor. The total cost associated with a tree is defined to be the sum of inversion distances between every pair of nodes in the tree; the objective is to find the tree with the smallest cost. Our consensus ranking algorithm allows multiple splits at each internal node compared to binary splits considered in previous work [Bourque and Pevzner, 2002]. The phylogenetic tree shown in Fig. 5 as a possible evolution scenario can be explained with 3111 elementary inversions compared to 4109 inversions required to explain the binary tree constructed by [Bourque and Pevzner, 2002]. Note that the reversal distances in the DNA data are much larger than in our synthetic experiments, and yet the phylogenetic tree reconstruction is efficient.

6 Discussion

This is, to our knowledge, the first paper offering a solution to the consensus ranking problem for signed permutations *under the inversion distance*.

We do this by formulating the problem in the broader context of statistical estimation; we introduce a new class of models over W_n that are intimately related to the combinatorial structure of signed permutations; we show that our model, the GMM, has sufficient statistics for both π_0 and $\bar{\theta}$, being thus in the exponential family; we also derive exact algorithms ASTAR, ASTARGMM to estimate these parameters from data.

We give very efficient implementations for each step of ASTAR, ASTARGMM; yet, the original problem being NP-hard, these algorithms will have an intractable number of steps in the worst case. In practice, however, we demonstrate that ASTAR is tractable on problems of relevant size. This result is not without theoretical support: indeed, for strongly modal data one can prove (proof omitted) that the algorithm will default to GREEDY.

Finally, we propose to use consensus under the inver-

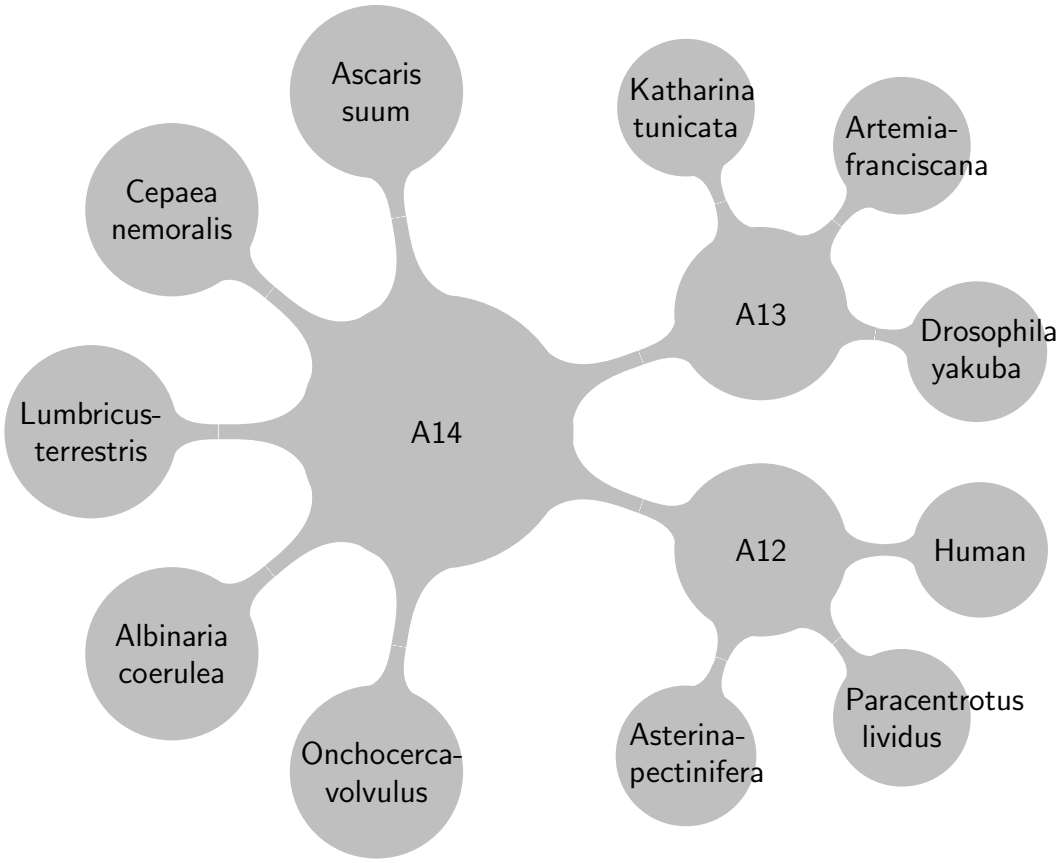

Figure 5: Reconstructed phylogenetic tree for Metazoan mtDNA dataset

sion distance as a surrogate for the biologically motivated consensus under reversal distance. We conjecture that due to the symmetry of W_n, the medians under the two distances will coincide under certain natural conditions, even though the distances themselves are very different. Our experiments with real and synthetic data explore this possibility with *very promising* results.

Prior work with signed permutations (Section 1.3) has focused on other distances, primarily the reversal distance. The techniques here are unrelated to that literature, but they have high-level similarities with the results and consensus ranking algorithm (for unsigned permutations) of [Meila et al., 2007] (which also expands prefixes). The main vehicle for similarity is the existence of the *code* for both S_n and W_n. This in turn is a consequence of a deeper fact relating to *reflection groups* [Bjorner and Brenti, 2005] of which both S_n and W_n are examples. An interesting future direction is to explore methods based on the irreducible representations of the hyperoctahedral group, similar to the Fourier-domain methods studied in [Kondor, 2008, Arora, 2009, Arora and Parthasarathy, 2010].

Appendix A.

Derivation of the estimation equation for θ_j The gradient of the log-partition function Z_{n-j} represents the expectation of the variable c_j given by $P(c_j) \propto e^{-\theta_j c_j}$. To estimate the parameter θ we have to equate $\frac{\partial L(\bar{\theta},\pi_0)}{\partial \theta_j} = 0$ where $L(\bar{\theta}, \pi_0)$ is given by (5) Setting the derivative to zero gives

$$\begin{aligned} \bar{c}_j &= -\frac{\partial \ln Z_{n-j}(\theta)}{\partial \theta} = g(n'_j, \theta) \\ &= \frac{e^{-\theta}}{1-e^{-\theta}}[1-(n'_j-1)e^{-\theta n'_j} + n'_j e^{-\theta(n'_j-1)}] \end{aligned}$$

where $n'_j = 2(n-j+1)$ as stated in the main text.

References

A. Klementiev et al. Unsupervised rank aggregation with distance-based models. In *ICML*, 2008.

W. Arndt and J. Tang. Improving inversion median computation using commuting reversals and cycle information. In *Comparative Genomics*, Vol. 4751 of LNCS, pages 30–44. Springer-Verlag, 2007.

Raman Arora. *Group theoretical methods in signal processing: learning similarities, transformations and invariants*. PhD thesis, University of Wisconsin-Madison, 2009.

Raman Arora and Harish Parthasarathy. Optimal estimation and detection in homogeneous spaces. *Signal Processing, IEEE Transactions on*, 58(5):2623–2635, 2010.

Martin Bader. The transposition median problem is np-complete. *Theor. Comput. Sci.*, 412:1099–1110, March 2011.

Vineet Bafna and Pavel A. Pevzner. Genome rearrangements and sorting by reversals. *SIAM J. Comput.*, 25(2):272–289, 1996.

Anders Bjorner and Francesco Brenti. *Combinatorics of Coxeter Groups*. Springer, 2005.

B. Bourque and P. Pevzner. Genome-scale evolution: Reconstructing gene orders in the ancestral species. *Genome Research*, 12(1):26–36, 2002.

A. Caprara. The reversal median problem. *INFORMS Journal on Computing*, 15(1):93–113, 2003.

Alberto Caprara. Sorting by reversals is difficult. In *Proceedings of the first annual international conference on Computational molecular biology*, RECOMB '97, pages 75–83. ACM, 1997.

Alberto Caprara. Sorting permutations by reversals and eulerian cycle decompositions. *SIAM J. Discrete Math.*, 12, 1999.

Persi Diaconis and R. L. Graham. Spearman's footrule as a measure of disarray. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(2): pp. 262–268, 1977. ISSN 00359246. URL `http://www.jstor.org/stable/2984804`.

M. A. Fligner and J. S. Verducci. Distance based ranking models. *Journal of the Royal Statistical Society B*, 48:359–369, 1986.

Sridhar Hannenhalli and Pavel Pevzner. Transforming cabbage into turnip: Polynomial algorithm for sorting signed permutations by reversals). *Journal of the ACM*, 46(1):178–189, January 1999.

Risi Kondor. *Group theoretical methods in machine learning*. PhD thesis, Columbia University, 2008.

Jian Ma. Reconstructing the history of large-scale genomic changes: Biological questions and computational challenges. *Journal of Computational Biology*, 18(7), 2011.

C. L. Mallows. Non-null ranking models. *Biometrika*, 44:114–130, 1957.

Marina Meila, Kapil Phadnis, Arthur Patterson, and Jeff Bilmes. Consensus ranking under the exponential model. In *In Proceedings of the 23rd Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, 2007.

J. H. Nadeau and B. A. Taylor. Lengths of chromosomonal segments conserved since divergence of man and mouse. *Proc. Nat. Acad. Sci.*, 81:814–818, 1984.

O. Wu et al. Learning to rank under multiple annotators. In *IJCAI*, 2011.

Jeffrey D. Palmer and Laura A. Herbon. Plant mitochondrial dna evolved rapidly in structure, but slowly in sequence. *Journal of Molecular Evolution*, 28:87–97, 1988.

Judea Pearl. *Heuristics: Intelligent Search Strategies for Computer Problem Solving*. Addison-Wesley, 1984.

Pavel A. Pevzner. *Computational Molecular Biology*. MIT Press, 2000.

David Sankoff, Robert Cedergren, and Yvon Abel. Genomic divergence through gene rearrangement. In *Methods in Enzymology*, volume 183, pages 428 – 438. Academic Press, 1990.

A. Siepel and B. Moret. Finding an optimal inversion median: Experimental results. In *Proc. 1st Workshop on Algorithms*, Vol. 2149 of Lecture Notes in Computer Science, pages 189–203. Springer-Verlag, 2001.

Richard P. Stanley. *Enumerative combinatorics*, volume 1. Cambridge University Press, Cambridge, England, 1997.

K. Swenson, Y. To, J. Tang, and B. Moret. Maximum independent sets of commuting and noninterfering inversions. *BMC Bioinformatics*, 10, 2009.

Eric Tannier, Chunfang Zheng, and David Sankoff. Multichromosomal median and halving problems under different genomic distances. *BMC Bioinformatics*, 10, 2009.