

Sequential Rank and Preference Learning with the Bayesian Mallows Model

Øystein Sørensen^{*}, Anja Stein[†], Waldir Leoncio Netto[§], and David S. Leslie[‡]

Abstract. The Bayesian Mallows model is a flexible tool for analyzing data in the form of complete or partial rankings, and transitive or intransitive pairwise preferences. In many potential applications of preference learning, data arrive sequentially and it is of practical interest to update posterior beliefs and predictions efficiently, based on the currently available data. Despite this, most algorithms proposed so far have focused on batch inference. In this paper we present an algorithm for sequentially estimating the posterior distributions of the Bayesian Mallows model using nested sequential Monte Carlo. The algorithm requires minimal user input in the form of tuning parameters, is straightforward to parallelize, and returns the marginal likelihood as a direct byproduct of estimation. We evaluate its performance in simulation experiments, and illustrate a real use case with sequential ranking of Formula 1 drivers throughout three seasons of races.

Keywords: Mallows mixtures, partial rankings, particle filter, preference learning, SMC2.

MSC2020 subject classifications: Primary 62L12; secondary 62F07.

1 Introduction

Data in the form of rankings and preferences arise naturally across a variety of domains. Examples include content recommendation based on click data (Liu et al., 2019b), algorithm comparison (Rojas-Delgado et al., 2022), consumer preferences (Kamishima, 2003), grant panel reviews (Pearce and Erosheva, 2022), genome-wide transcriptomic analyses (Eliseussen et al., 2022), analysis of mutual funds' preferences for governance structures (Yi, 2021), social hierarchies (Nicholls et al., 2022), and reinforcement learning from human feedback (Hwang et al., 2023). An early model for analyzing rankings (Thurstone, 1927) assumed a judge ranks m items by assigning a score to each item, and then ordering them according to the score. Further developments of this approach include the Plackett-Luce model (Luce, 1959; Plackett, 1975), the Babington Smith model (Babington Smith, 1950), and the Bradley-Terry model (Bradley and Terry, 1952), all of which are based on assigning real-valued utilities to each item, yielding a large number of parameters to be estimated.

Now consider a collection of items $\mathcal{A} = \{A_1, A_2, \dots, A_m\}$, and let ρ be a permutation of the integers $[m] := \{1, 2, \dots, m\}$ denoting the items' modal ranking in the

^{*}Department of Psychology, University of Oslo, oystein.sorensen@psykologi.uio.no

[†]School of Mathematical Sciences, Lancaster University, a.k.stein@outlook.com

[‡]School of Mathematical Sciences, Lancaster University, d.leslie@lancaster.ac.uk

[§]Oslo Centre for Biostatistics and Epidemiology, University of Oslo, w.l.netto@medisin.uio.no

population of interest, such that ρ_i denotes the modal ranking of item A_i . For a particular individual, let $A_i \succ A_j$ imply that the individual prefers A_i to A_j , and let \mathbf{r} be the permutation of $[m]$ encoding the individual’s rankings. Mallows (1957) showed that if the probability that an individual ranks a pair of items in agreement with their relative position in $\boldsymbol{\rho}$ is given by

$$P(A_i \succ A_j | \rho_i < \rho_j) = 0.5 + 0.5 \tanh\{(\rho_j - \rho_i) \log \theta + \log \phi\}$$

we obtain an exponential model $P(\mathbf{r} | \boldsymbol{\rho}) \propto \exp\{-\alpha d(\boldsymbol{\rho}, \mathbf{r})\}$ for the observed ranking \mathbf{r} . When $\phi = 1$, $d(\cdot, \cdot)$ is Spearman’s rank correlation and when $\theta = 1$, $d(\cdot, \cdot)$ is the Kendall distance. The precision parameter α quantifies how far observed rankings typically are from the modal ranking. Advantages of the Mallows model over utility-based models include a lower number of parameters (α and $\boldsymbol{\rho}$) and the fact that its support is defined on the space of rankings. The model has later been extended to incorporate additional distance functions (Diaconis, 1988) and to item-dependent precision parameters (Fligner and Verducci, 1986). We refer to the reviews by Liu et al. (2019a) and Yu et al. (2019) and the monograph by Marden (1995) for further details.

Vitelli et al. (2017) proposed a Bayesian Mallows model and a Markov chain Monte Carlo (MCMC) algorithm for its estimation. Compared to other approaches focusing on the Kendall or Cayley distances (Irurozki et al., 2018; Lu and Boutilier, 2014; Meila and Bao, 2010), Vitelli et al. (2017)’s algorithm works naturally with any of the distance functions proposed by Diaconis (1988) for the Mallows model, and it incorporates data in the form of partial rankings or pairwise preferences. Its fully Bayesian approach allows predicting users’ preferences of items they have not yet seen, allowing the model to be used as a probabilistic recommender system (Liu et al., 2019b).

The MCMC algorithm of Vitelli et al. (2017) has some drawbacks, however. The user has to set tuning parameters for the proposal distributions for precision parameters, modal rankings, and latent rankings. It hence may require several pilot runs for obtaining sufficient acceptance probabilities. Second, in settings where data arrive sequentially, updating the posteriors requires running the full algorithm from scratch. The goal of this paper is to alleviate these issues. To this end, we propose a nested sequential Monte Carlo (SMC²) algorithm (Chopin et al., 2013; Fulop and Li, 2013), which requires minimum user input and efficiently updates posterior distributions when new data arrive. The algorithm is straightforward to parallelize and is an extension of the work by Stein (2023), who proposed a sequential Monte Carlo (SMC) algorithm using a resample-move scheme (Berzuini and Gilks, 2001).

A motivating example is shown in Figure 1, showing race results from the Formula 1 seasons 2022–2024.¹ We included the 16 drivers who completed 50 or more of the total 68 races, and computed rankings from the results of each race. If we assume that the underlying skills of the drivers—as well as the performance of the cars and teams—are relatively stable across the three seasons, the results of each race can be viewed as a noisy assessment of the underlying true ranking. Using the SMC² algorithm proposed in this paper we can easily update the posterior distribution for the ranking of the drivers after each race. We will revisit these data in Section 5.

¹Data were downloaded from <https://github.com/toUpperCase78/formula1-datasets>

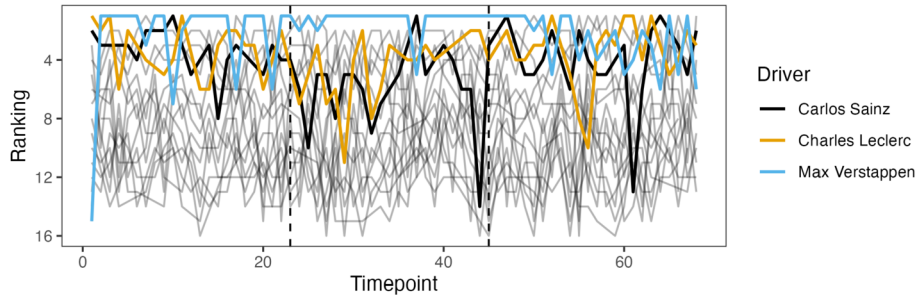


Figure 1: Rankings of drivers based on race results in Formula 1 seasons 2022, 2023, and 2024. Vertical dashed lines indicate the start of a season. Traces for the three drivers with best mean ranking across all races are shown in color.

The paper proceeds as follows. In Section 2 we provide necessary background on the Bayesian Mallows model. In Section 3 we propose an SMC² algorithm for the Bayesian Mallows model with partial rankings or pairwise preference data. In Section 4 we report the results of simulation experiments and in Section 5 we revisit the Formula 1 data. We conclude and discuss further developments in Section 6. Section 1 of the Supplementary Material (Sørensen et al., 2025) shows an extension of the proposed algorithms to sequential estimation of mixtures of Mallows models, and Section 2 of the Supplementary Material discusses in detail the generation of latent rankings. An R package providing an interface to our C++ implementation is available from GitHub² and R code for reproducing all the results in the paper is available from our OSF repository.³

2 Background and Model Setup

We now introduce the Bayesian Mallows model (Crispino et al., 2019; Vitelli et al., 2017).

2.1 Mallows' Model for Partial Rankings and Pairwise Preferences

Let \mathcal{P}_m denote the space of all permutations of $[m]$ and consider rankings $\mathbf{r} \in \mathcal{P}_m$ of a set of items \mathcal{A} distributed according to a Mallows model (Diaconis, 1988; Mallows, 1957; Vitelli et al., 2017)

$$p(\mathbf{r}|\theta) = Z(\alpha)^{-1} \exp\{-\alpha d(\mathbf{r}, \boldsymbol{\rho})\} 1\{\mathbf{r} \in \mathcal{P}_m\}, \quad (1)$$

where $\theta = \{\alpha, \boldsymbol{\rho}\}$, $1\{A\}$ is an indicator function for the event A , $\alpha \in \mathbb{R}_{\geq 0}$ is a precision parameter, and $\boldsymbol{\rho} \in \mathcal{P}_m$ is the modal ranking. $d(\cdot, \cdot)$ is a right-invariant distance function, and

$$Z(\alpha) = \sum_{\mathbf{r} \in \mathcal{P}_m} \exp\{-\alpha d(\mathbf{r}, \mathbf{e})\} \quad (2)$$

²<https://github.com/osorensen/BayesMallowsSMC2>

³<https://osf.io/pquk4/>

is the normalizing constant where $\mathbf{e} = (1, 2, \dots, m)'$.

A ranking \mathbf{r} is a latent variable, and observations \mathbf{y} are distributed according to $p(\mathbf{y}|\mathbf{r}, \theta)$. The marginal likelihood of N observations $\mathbf{y}_{1:N} = \{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_N\}$ is

$$p(\mathbf{y}_{1:N}|\theta) = \prod_{n=1}^N Z(\alpha)^{-1} \sum_{\mathbf{r}_n \in \mathcal{P}_m} \exp\{-\alpha d(\mathbf{r}_n, \boldsymbol{\rho})\} p_\epsilon(\mathbf{y}_n|\mathbf{r}_n) \quad (3)$$

where $p_\epsilon(\mathbf{y}_n|\mathbf{r}_n)$ is the sampling distribution of the observed rankings given the latent rankings where ϵ is an error parameter to be introduced later. Here and in the sequel, Greek letter subscripts imply conditioning.

Complete rankings correspond to $p_\epsilon(\mathbf{y}_n|\mathbf{r}_n) = 1\{\mathbf{y}_n = \mathbf{r}_n\}$. For top- k_n rankings with $k_n \in [m]$ we define the set of items ranked by user n as $\mathcal{A}_n = \{A_i \in \mathcal{A} : r_{ni} \leq k_n\}$ whereas for ranks missing completely at random we let \mathcal{A}_n define the set of ranked items. In both cases we have $p_\epsilon(y_{ni}|\mathbf{r}_n) = 1\{y_{ni} = r_{ni}\}$ for $i : A_i \in \mathcal{A}_n$. Defining the set of latent rankings consistent with the observations

$$\mathcal{S}_n = \{\mathbf{r} \in \mathcal{P}_m : (r_i = y_{ni} \ \forall i : A_i \in \mathcal{A}_n)\} \quad (4)$$

with the complete data case given by $\mathcal{S}_n = \{\mathbf{y}_n\}$, the marginal likelihood (3) reduces to

$$p(\mathbf{y}_{1:N}|\theta) = \prod_{n=1}^N Z(\alpha)^{-1} \sum_{\mathbf{r}_n \in \mathcal{S}_n} \exp\{-\alpha d(\mathbf{r}_n, \boldsymbol{\rho})\}. \quad (5)$$

When the data contain pairwise preferences for $p_n \leq \binom{m}{2}$ pairs of items, let y_{ni} denote the i 'th pairwise preference of user n . We define the function

$$g(y_{ni}, \mathbf{r}_n) = \begin{cases} 0 & \text{if } y_{ni} = (A_s \succ A_t) \text{ and } (r_{ns} < r_{nt}) \\ 1 & \text{if } y_{ni} = (A_s \succ A_t) \text{ and } (r_{ns} > r_{nt}) \end{cases}$$

indicating whether the pairwise preferences contradict the latent rankings. Assuming a true latent ranking exists, inconsistencies arise due to errors made by the users.⁴ If errors occur independently at rate $\epsilon \in [0, 1]$, the direction of the preference relation for the i 'th pair in \mathbf{y}_n has distribution

$$p_\epsilon(y_{ni}|\mathbf{r}_n) = \begin{cases} 1 - \epsilon & \text{if } g(y_{ni}, \mathbf{r}_n) = 0 \\ \epsilon & \text{if } g(y_{ni}, \mathbf{r}_n) = 1. \end{cases}$$

Setting $\epsilon > 0$ allows mutually incompatible preferences (Crispino et al., 2019), and the marginal likelihood (3) becomes

$$p(\mathbf{y}_{1:N}|\theta) = \prod_{n=1}^N Z(\alpha)^{-1} \sum_{\mathbf{r}_n \in \mathcal{P}_m} \exp\{-\alpha d(\mathbf{r}_n, \boldsymbol{\rho})\} \left(\frac{\epsilon}{1-\epsilon}\right)^{\sum_{i=1}^{p_n} g(y_{ni}, \mathbf{r}_n)} (1-\epsilon)^{p_n}$$

⁴This assumption is indeed a mathematical idealization, as empirical evidence suggest that human preferences are inherently non-transitive (Kahneman and Tversky, 1979).

where θ now also contains ϵ . To only allow mutually compatible pairwise preferences we set $\epsilon = 0$ and define the set of latent rankings consistent with \mathbf{y}_n as

$$\mathcal{S}_n = \{\mathbf{r} \in \mathcal{P}_m : (A_s \succ A_t) \in \text{tc}(\mathbf{y}_n) \Leftrightarrow r_s < r_t\}, \quad (6)$$

where $\text{tc}(\mathbf{y}_n)$ is the transitive closure of the directed acyclic graph induced by the pairwise preferences. In this case we can compute the marginal likelihood using (5), replacing \mathcal{S}_n from (4) with \mathcal{S}_n from (6).

2.2 Prior Distributions

For the precision parameter α we follow Crispino et al. (2019) and use a gamma prior with shape $\gamma > 0$ and rate $\lambda > 0$, $\pi(\alpha) = \lambda^\gamma \Gamma(\gamma)^{-1} \alpha^{\gamma-1} e^{-\lambda\alpha}$, where $\Gamma(\gamma) = \int_0^\infty t^{\gamma-1} e^{-t} dt$. Similar to Vitelli et al. (2017), for the modal ranking we use a uniform prior on \mathcal{P}_m , $\pi(\boldsymbol{\rho}) = (m!)^{-1} 1\{\boldsymbol{\rho} \in \mathcal{P}_m\}$.

With non-transitive pairwise preferences, we use the Bernoulli model of Crispino et al. (2019) with a truncated Beta prior on $[0, 0.5)$, $\pi(\epsilon) \propto \epsilon^{\kappa_1-1} (1-\epsilon)^{\kappa_2-1} 1\{\epsilon \in [0, 0.5)\}$. A model without non-transitive pairwise preferences corresponds to the limit $\kappa_1 \rightarrow \infty$ and $\kappa_2 \rightarrow 0$, which means that ϵ is fixed to 0 and can be ignored in the analyses. Crispino et al. (2019) also considered a logistic model, in which the logit of the error probability depends on the distance between the items in the latent ranking. While sequential inference with this logistic model is in principle straightforward, we do not consider it further in this paper for ease of presentation.

2.3 Distance Functions and Normalizing Constants

Consider two rankings $\mathbf{a}, \mathbf{b} \in \mathcal{P}_m$. Cayley distance measures the minimum number of pairwise swaps needed for converting \mathbf{a} into \mathbf{b} , two algorithms for which are given in Marden (1995, pp. 25-26). Ulam distance can be defined as m minus the length of the longest common subsequence of the item orderings corresponding to \mathbf{a} and \mathbf{b} . We further have Spearman distance $d(\mathbf{a}, \mathbf{b}) = \|\mathbf{a} - \mathbf{b}\|_2^2$, Kendall distance measuring the number of discordant pairs $d(\mathbf{a}, \mathbf{b}) = \sum_{i=1}^m \sum_{j=i+1}^m 1\{(a_i - a_j)(b_i - b_j) < 0\}$, the footrule $d(\mathbf{a}, \mathbf{b}) = \sum_{i=1}^m |a_i - b_i|$, and Hamming distance $d(\mathbf{a}, \mathbf{b}) = \sum_{i=1}^m 1\{a_i \neq b_i\}$.

The choice of distance in the Mallows model (1) is ultimately linked to the application at hand. For example, Hamming distance is not likely to work well under the ranking and preference applications considered in this paper but arises naturally when the Mallows model is used for matchings, e.g., when tracking a number of known objects using noisy sensors (Irurozki et al., 2019). Similarly, Cayley distance is suitable when total disorder is of interest (Crispino et al., 2019), whereas footrule, Kendall, and Spearman distance are most appropriate for preference data. Crispino et al. (2019) gives an example of two rankings $\mathbf{a} = (1, 2, 3, 4, 5)$ and $\mathbf{b} = (5, 2, 3, 4, 1)$. Their Cayley distance (normalized to be in $[0, 1]$) is 0.25, whereas the normalized footrule, Kendall, and Spearman distances are 2/3, 0.7, and 0.8, respectively. If \mathbf{a} and \mathbf{b} represent positions on a genome they can be seen as close, and the Cayley distance may be most appropriate.

On the other hand, if \mathbf{a} and \mathbf{b} represent preferences for five items, they are far apart and one of the latter three distances are better. An in-depth comparison of Cayley, Kendall and Ulam distances can be found in Ceberio et al. (2015) and further discussion in Diaconis (1988, Ch. 6).

Computing the normalizing constant $Z(\alpha)$ as in (2) requires summing over $|\mathcal{P}_m| = m!$ terms, but tractable exact expressions exist for Cayley, Kendall, and Hamming distances (Fligner and Verducci, 1986; Irurozki et al., 2018). Furthermore, since $d(\mathbf{r}, \mathbf{e})$ takes on a set of $l < m!$ values $\mathcal{D} = \{d_1, d_2, \dots, d_l\}$, we can define $L_i = \{\mathbf{r} \in \mathcal{P}_m : d(\mathbf{r}, \mathbf{e}) = d_i\}$ and write $Z(\alpha) = \sum_{i=1}^l |L_i| e^{-\alpha d_i}$. For the footrule, $l = \mathcal{O}(m^2)$, for Spearman distance $l = \mathcal{O}(m^3)$, and for Ulam distance $l = \mathcal{O}(m)$ (Crispino, 2018; Crispino et al., 2023; Irurozki, 2014; Irurozki et al., 2016). Unfortunately, while the set of distances \mathcal{D} is well known, finding the cardinalities $|L_i|$ is hard. The Online Encyclopedia of Integer Sequences (Sloane, 2023) contains $|L_i|$ up to $m = 50$ for the footrule, $m = 20$ for Spearman distance, and $m = 60$ for Ulam distance. Beyond these upper limits, asymptotic approximations exist for the footrule and Spearman distances (Crispino et al., 2023; Mukherjee, 2016), and an importance sampling scheme has been developed by Vitelli et al. (2017). The latter can in principle be run to arbitrary precision, and importantly, estimates of the partition function can be precomputed for a given number of items m over a grid of α values (Sørensen et al., 2020). Thus, in the rest of this paper we assume that $Z(\alpha)$ is available and that its Monte Carlo error (if any) is negligible compared to the Monte Carlo error of the proposed SMC² algorithm. In the simulations and application examples, the number of items m is always such that $Z(\alpha)$ is known exactly.

3 Sequential Inference in the Bayesian Mallows Model

Now assume data become available sequentially at timepoints $t = 1, 2, \dots, T$, and let $\mathbf{y}_{\mathcal{I}_t}$ contain partial rankings or pairwise preferences for new users entering the pool at time t , where $\mathcal{I}_t \subset \mathbb{N}$ is the set of user indices. Let $\mathcal{I}_{1:t} = \cup_{t=1}^T \mathcal{I}_t$ contain the indices of all users in the pool at time t . We assume throughout that a given user enters the pool only once, i.e., that $\mathcal{I}_s \cap \mathcal{I}_t = \emptyset$ if $s \neq t$. The target distribution is $\pi(\theta, \mathbf{r}_{\mathcal{I}_{1:t}} | \mathbf{y}_{\mathcal{I}_{1:t}})$, with static parameters $\theta = \{\alpha, \boldsymbol{\rho}, \epsilon\}$ and latent rankings $\mathbf{r}_{\mathcal{I}_{1:t}}$. The goal is to estimate the target distribution at all timepoints, in order to continuously perform inference based on the currently available evidence.

Sequential Monte Carlo (SMC) (Dai et al., 2022) typically scales better than MCMC for these types of problems, as the latter needs to be completely rerun at each new timepoint. Stein (2023) considered SMC for sequential inference in Bayesian Mallows models using a resample-move framework (Berzuini and Gilks, 2001). Unfortunately, such methods were designed for cases either with only static parameters or latent variables which can be easily integrated out. Integrating over the latent rankings is computationally demanding and we thus instead base our methodology on SMC² (Chopin et al., 2013; Fulop and Li, 2013) which uses particle marginal Metropolis-Hastings (Andrieu et al., 2010) in the rejuvenation step and was developed specifically for settings with challenging latent variable distributions. We extend the SMC² framework by incorporating the

hybrid particle MCMC sampler proposed by Mendes et al. (2020) to allow a combination of Gibbs sampling and Metropolis-Hastings steps.

To set the notation, assume we have R particles each containing static parameters $\theta^r = \{\alpha^r, \boldsymbol{\rho}^r, \epsilon^r\}$ and to each of these we attach S additional particles containing the latent rankings for the users entered up to timepoint t , $\mathbf{r}_{\mathcal{I}_{1:t}}^{s,r}$. Here and in what follows, superscripts r and s are assumed repeated for all $r = 1, 2, \dots, R$ and $s = 1, 2, \dots, S$. If the data consist of either partial rankings or consistent pairwise preferences we have $\epsilon^r = 0$ and this parameter can be ignored.

The building blocks of the algorithm are particle filters for latent rankings and cluster labels (Section 3.1), iterated batch importance sampling for the static parameters (Section 3.2), and a rejuvenation algorithm (Section 3.3).

3.1 Particle Filters for Latent Rankings

Define ancestor indices a_{t-1}^s indicating which particle at time $t-1$ is the ancestor of particle s at time t , and set the initial value $a_0^s = s$. At timepoint t , $\mathbf{r}_{n,t-1}^{a_{t-1}^s}$ denotes the latent rankings of user $n \in \mathcal{I}_{1:t-1}$ in particle s . The r superscripts linking latent rankings to static parameter particles are omitted for ease of notation. Algorithm 1 approximates $\pi_\theta(\mathbf{r}_{\mathcal{I}_{1:T}}) = p(\mathbf{r}_{\mathcal{I}_{1:T}} | \mathbf{y}_{\mathcal{I}_{1:T}}, \theta)$ for fixed θ .

Algorithm 1 Particle Filter.

- 1: **for** $t = 1$ to T **do**
- 2: **if** $t > 1$ **then**
- 3: Sample $a_{t-1}^s \in [S]$ with probabilities $W_{t-1,\theta}^{1:S}$.
- 4: Set $\mathbf{r}_{n,t}^s \leftarrow \mathbf{r}_{n,t-1}^{a_{t-1}^s}$ for all $n \in \mathcal{I}_{1:t-1}$.
- 5: **end if**
- 6: Sample $\mathbf{r}_{n,t}^s \sim q_\theta(\cdot | \mathcal{S}_n)$ for all $n \in \mathcal{I}_t$.
- 7: Compute weights

$$w_{t,\theta}^s = \prod_{n \in \mathcal{I}_t} \frac{\exp\{-\alpha d(\mathbf{r}_{n,t}^s, \boldsymbol{\rho})\}}{Z(\alpha) q_\theta(\mathbf{r}_{n,t}^s | \mathcal{S}_n)} \left(\frac{\epsilon}{1 - \epsilon} \right)^{\sum_{i=1}^{p_{n,t}} g(y_{ni}, \mathbf{r}_{n,t}^s)} (1 - \epsilon)^{p_{n,t}}. \quad (7)$$

- 8: Normalize weights

$$W_{t,\theta}^s = \frac{w_{t,\theta}^s}{\sum_{s=1}^S w_{t,\theta}^s}. \quad (8)$$

- 9: **end for**
-

In Algorithm 1 line 4 ensures that estimated latent rankings for all users are available at each timepoint, but can be omitted to reduce the memory cost. On line 6, \mathcal{S}_n is given by (4) in the case of partial rankings, (6) in the case of consistent pairwise preferences, and \mathcal{P}_m in the case of non-transitive pairwise preferences. We present the details of

these proposal distributions in Section 2 of the Supplementary Material (Sørensen et al., 2025).

The expression for the weights (7) is based on Step 2(c) of Chopin et al. (2013, Section 2.1) which in our notation becomes

$$w_{t,\theta}^s = \frac{f_\theta \left(\mathbf{r}_{\mathcal{I}_t}^s | \mathbf{r}_{\mathcal{I}_{1:t-1}}^{a_{t-1}^s} \right) p_\epsilon \left(\mathbf{y}_{\mathcal{I}_t} | \mathbf{r}_{\mathcal{I}_t}^s \right)}{q_{t,\theta} \left(\mathbf{r}_{\mathcal{I}_t}^s | \mathbf{r}_{\mathcal{I}_{t-1}}^{a_{t-1}^s} \right)}.$$

Because independent users arrive at each timepoint, $\mathbf{r}_{\mathcal{I}_t}^s$ and $\mathbf{r}_{\mathcal{I}_{1:t-1}}^{a_{t-1}^s}$ are independent given θ and we get

$$f_\theta \left(\mathbf{r}_{\mathcal{I}_t}^s | \mathbf{r}_{\mathcal{I}_{1:t-1}}^{a_{t-1}^s} \right) = f_\theta \left(\mathbf{r}_{\mathcal{I}_t}^s \right) = \prod_{n \in \mathcal{I}_t} \frac{1}{Z(\alpha)} \exp \left\{ -\alpha d \left(\mathbf{r}_{n,t}^s, \boldsymbol{\rho} \right) \right\}.$$

Next, it follows from Section 2.1 that

$$p_\epsilon \left(\mathbf{y}_{\mathcal{I}_t} | \mathbf{r}_{\mathcal{I}_t}^s \right) = \prod_{n \in \mathcal{I}_t} \left(\frac{\epsilon}{1 - \epsilon} \right)^{\sum_{i=1}^{p_{n,t}} g \left(\mathbf{y}_{ni}, \mathbf{r}_{n,t}^s \right)} (1 - \epsilon)^{p_{n,t}}$$

which simplifies to $p_\epsilon \left(\mathbf{y}_{\mathcal{I}_t} | \mathbf{r}_{\mathcal{I}_t}^s \right) = 1$ when $\epsilon = 0$ and where $p_{n,t}$ denotes the number of pairwise preferences in \mathbf{y}_n for some $n \in \mathcal{I}_t$. The proposal distribution is

$$q_{t,\theta} \left(\mathbf{r}_{\mathcal{I}_t}^s | \mathbf{r}_{\mathcal{I}_{t-1}}^{a_{t-1}^s} \right) = \prod_{n \in \mathcal{I}_t} q_\theta \left(\mathbf{r}_{n,t}^s | \mathcal{S}_n \right).$$

These three expressions combine to yield (7). For the special case $\epsilon = 0$ we recover the weight update formula in Algorithm 12 of Stein (2023).

In the resampling step on line 3 in Algorithm 1, as well as in the resampling steps of all subsequent algorithms, both multinomial resampling and the lower variance alternatives residual resampling (Liu and Chen, 1998), stratified resampling (Kitagawa, 1996), and systematic resampling (Kitagawa, 1996) can be used and are part of our implementation. We refer to Douc and Cappe (2005) and Hol et al. (2006) for details.

We also note the important fact that the quantity

$$\hat{Z}_t \left(\theta, \mathbf{r}_{\mathcal{I}_{1:t}}^{1:S}, a_{1:t-1}^{1:S} \right) = \frac{1}{S^t} \prod_{t'=1}^t \left\{ \sum_{s=1}^S w_{t',\theta}^s \right\} \quad (9)$$

is an unbiased estimator of the marginal likelihood $p(\mathbf{y}_{\mathcal{I}_{1:t}} | \theta)$ (Del Moral, 2004, Section 7.4.1).

Conditional Particle Filter

To allow particle Gibbs sampling in the rejuvenation step, we need a conditional particle filter (Andrieu et al., 2010) for which the full ancestral history of a given particle $\mathbf{r}_{\mathcal{I}_{1:T}}^k$ is fixed. This particle filter is shown in Algorithm 2 and yields samples approximately distributed according to $p(\mathbf{r}_{\mathcal{I}_{1:T}}^{-k} | \mathbf{y}_{\mathcal{I}_{1:T}}, \theta, \mathbf{r}_{\mathcal{I}_{1:T}}^k)$, where $k \in \{1, 2, \dots, S\}$ and $\mathbf{r}_{\mathcal{I}_{1:T}}^{-k}$ denotes the set of all particles except particle k .

Algorithm 2 Conditional Particle Filter.

- 1: Condition on a trajectory $\mathbf{r}_{\mathcal{I}_{1:T}}^k$ with ancestral lineage $b_T^k = k$ and $b_t^k = a_t^{b_{t+1}^k}$, $t = T - 1, \dots, 1$.
 - 2: **for** $t = 1$ to T **do**
 - 3: **if** $t > 1$ **then**
 - 4: For $s \neq b_t^k$ sample $a_{t-1}^s \in [S]$ with probabilities $W_{t-1, \theta}^{1:S}$.
 - 5: Set $\mathbf{r}_{n,t}^s \leftarrow \mathbf{r}_{n,t-1}^{a_{t-1}^s}$ for all $n \in \mathcal{I}_{1:t-1}$.
 - 6: **end if**
 - 7: For $s \neq b_t^k$ sample $\mathbf{r}_{n,t}^s \sim q_\theta(\cdot | \mathcal{S}_{n,t})$ for all $n \in \mathcal{I}_t$.
 - 8: Compute weights using (7) and normalize them using (8).
 - 9: **end for**
-

3.2 SMC² Algorithm

The top-level algorithm for sampling the static parameters is an extension of iterated batch importance sampling (Chopin, 2002) which uses the particle filters of the previous section to integrate out the latent variables, and is stated in Algorithm 3. Since the particle filters yield unbiased estimates of the marginal likelihood, Algorithm 3 targets the correct posterior distribution $\pi(\theta | \mathbf{y}_{\mathcal{I}_{1:t}})$ at each $t = 1, 2, \dots, T$ (Chopin et al., 2013).

Considering Algorithm 3, first note that equation (10) is an unbiased estimator of

$$p(\mathbf{y}_{\mathcal{I}_t} | \mathbf{y}_{\mathcal{I}_{1:t-1}}, \theta^r) = \prod_{n \in \mathcal{I}_t} Z(\alpha^r)^{-1} \sum_{\mathbf{r}_n \in \mathcal{S}_n} \exp\{-\alpha^r d(\mathbf{r}_n, \boldsymbol{\rho}^r)\}. \quad (13)$$

The marginal likelihood increments are given by

$$\hat{p}(\mathbf{y}_{\mathcal{I}_t} | \mathbf{y}_{\mathcal{I}_{1:t-1}}) = \sum_{r=1}^R \Omega^r \times \hat{p}(\mathbf{y}_{\mathcal{I}_t} | \mathbf{y}_{\mathcal{I}_{1:t-1}}, \theta^r) \quad (14)$$

and can be used to estimate the unconditional marginal likelihood

$$\hat{p}(\mathbf{y}_{\mathcal{I}_{1:t}}) = \prod_{t'=1}^t \hat{p}(\mathbf{y}_{\mathcal{I}_{t'}} | \mathbf{y}_{\mathcal{I}_{1:t'-1}}). \quad (15)$$

The rejuvenation threshold A can be set to $R/2$. As in Fulop and Li (2013), we iterate the rejuvenation algorithm at least once, and stop when the number of unique particles exceeds A or when some upper limit on the number of iterations is reached. If the acceptance rate in the rejuvenation step is below some threshold B , which we set to $B = 0.2$ here, the number of particle filters is doubled. The doubling on lines 10–13 implements the exchange importance sampling step of Chopin et al. (2013, Section 3.6.1). The components in (12) are readily available from the call to Algorithm 4 in the rejuvenation step.

Algorithm 3 SMC² Algorithm.

-
- 1: Sample $\theta^r = \{\alpha^r, \boldsymbol{\rho}^r, \epsilon^r\}$ from their priors and set $\omega^r \leftarrow 1$.
 - 2: **for** $t = 1$ to T **do**
 - 3: Perform iteration t of the particle filter in Algorithm 1 with $\theta = \theta^r$ and compute

$$\hat{p}(\mathbf{y}_{\mathcal{I}_t} | \mathbf{y}_{\mathcal{I}_{1:t-1}}, \theta^r) = \frac{1}{\tilde{S}} \sum_{s=1}^{\tilde{S}} w_{t,\theta}^s. \quad (10)$$

- 4: Update and normalize importance weights

$$\omega^r \leftarrow \omega^r \times \hat{p}(\mathbf{y}_{\mathcal{I}_t} | \mathbf{y}_{\mathcal{I}_{1:t-1}}, \theta^r), \quad \Omega^r = \frac{\omega^r}{\sum_{r=1}^R \omega^r}. \quad (11)$$

- 5: Compute the effective sample size $\text{ESS} = \{\sum_{r=1}^R (\Omega^r)^2\}^{-1}$.
- 6: **if** $\text{ESS} < A$ **then**
- 7: Sample $a_t^r \in [R]$ with probabilities Ω^r and set $(\theta^r, \omega^r) \leftarrow (\theta^{a_t^r}, 1)$.
- 8: Rejuvenate with Algorithm 4, letting ζ denote the acceptance rate of (17).
- 9: **if** $\zeta < B$ **then**
- 10: Set $\tilde{S} = 2S$ and sample $i^{\tilde{s}} \in [S]$ for $\tilde{s} = 1, \dots, \tilde{S}$ with probabilities W_{t,θ^r}^s .
- 11: Set $\{\tilde{\mathbf{r}}_{\mathcal{I}_{1:t}}^{1:\tilde{S}}, \tilde{\mathbf{a}}_{1:t-1}^{1:\tilde{S}}\} \leftarrow \{\mathbf{r}_{\mathcal{I}_{1:t}}^{i^{\tilde{s}}}, \mathbf{a}_{1:t-1}^{i^{\tilde{s}}}\}$ and $\tilde{w}_{1:t,\theta^r}^{1:\tilde{S}} \leftarrow w_{1:t,\theta^r}^{i^{\tilde{s}}}$.
- 12: Update the particle weight

$$\omega^r \leftarrow \omega^r \times (S/\tilde{S})^t \frac{\prod_{t'=1}^t \left\{ \sum_{s=1}^{\tilde{S}} \tilde{w}_{t',\theta^r}^s \right\}}{\prod_{t'=1}^t \left\{ \sum_{s=1}^S w_{t',\theta^r}^s \right\}}. \quad (12)$$

- 13: Update $S \leftarrow \tilde{S}$.
 - 14: **end if**
 - 15: **end if**
 - 16: **end for**
-

Parallelization

To reduce the amount of communication between nodes, it seems most sensible to parallelize the top-level Algorithm 3 rather than the particle filters. There are two main approaches to this in the literature (Dai et al., 2022; Naesseth et al., 2019). Jun et al. (2012) and Murray et al. (2016) use the fact that the weight updates in equations (10)-(11) can be done independently for each of the R particles. However, computing effective sample size and subsequently resampling requires communication between the nodes, and hence makes its implementation complicated.

A more straightforward approach, which we use in this paper, is what Naesseth et al. (2019) call importance weighted SMC samplers. In this case the full algorithm with R particles is run independently on P different nodes. Let $\theta^{r,p}$ denote the r th particle of the SMC² algorithm run on the p th compute node and $\Omega^{r,p}$ its weight, for $r = 1, \dots, R$

and $p = 1, \dots, P$. Also let $\hat{p}(\mathbf{y}_{\mathcal{I}_{1:T}})^p$ denote the marginal likelihood estimate (15) from the p 'th node. The combined set of particles $\{\theta^{r,p}\}$ with weights

$$\Omega^{r,p} \times \frac{\hat{p}(\mathbf{y}_{\mathcal{I}_{1:T}})^p}{\sum_{p'=1}^P \hat{p}(\mathbf{y}_{\mathcal{I}_{1:T}})^{p'}} \quad (16)$$

now yield a consistent estimate of the target distribution as $P \rightarrow \infty$ for any R (Naesseth et al., 2019, Section 4.4.1). The combined estimate of the marginal likelihood itself can be obtained by direct averaging, $\hat{p}(\mathbf{y}_{\mathcal{I}_{1:T}}) = \sum_{p'=1}^P \hat{p}(\mathbf{y}_{\mathcal{I}_{1:T}})^{p'} / P$.

Prediction of Latent Rankings

Predicting the latent rankings $\mathbf{r}_{\mathcal{I}_t}$ of the users entering at time t is a filtering problem, and we can obtain R samples, weighted by Ω^r , from $p(\mathbf{r}_{\mathcal{I}_t} | \theta, \mathbf{y}_{\mathcal{I}_{1:t}})$ by drawing an index $s \sim \mathcal{M}(W_{t,\theta}^{r,s})$ for each particle r (Chopin et al., 2013, Section 3.3), where $\mathcal{M}(\cdot)$ denotes a multinomial distribution.

Sampling from the posterior $P(\mathbf{r}_{\mathcal{I}_{1:t}} | \theta, \mathbf{y}_{\mathcal{I}_{1:t}})$ of the latent rankings of all users entered until time t can be done identically, but requires storing the full path for each particle. That is, if we draw a particle with index s we need to trace its latent variables according to its genealogy back until time 1. As noted by Chopin et al. (2013), this storage requirement can be avoided by triggering the particle doubling step in Algorithm 3 whenever a complete trajectory of the latent variables are needed, and then sampling an index $s \sim \mathcal{M}(W_{t,\theta}^{r,s})$ for each $r \in [R]$.

3.3 Rejuvenation

The rejuvenation step prevents degeneracy by moving each particle independently with an MCMC kernel. The original SMC² rejuvenation algorithms of Chopin et al. (2013) and Fulop and Li (2013) used particle marginal Metropolis-Hastings, but we instead use the algorithm proposed in Mendes et al. (2020) which combines particle marginal Metropolis-Hastings with particle Gibbs. This is useful in the present case because the error probability ϵ as well as the cluster probabilities discussed in Section 1 of the Supplementary Material (Sørensen et al., 2025) can be sampled conditionally, whereas the dispersion parameter α and the modal ranking $\boldsymbol{\rho}$ require a Metropolis-Hastings algorithm. For ease of notation, now let T denote the current value t of SMC² at the moment the rejuvenation algorithm is called. Also define $a \wedge b = \min\{a, b\}$.

Algorithm 4 starts by computing the variance of α , using unweighted formulas because we always resample before rejuvenating. These variances are used for tuning the random walk proposal on line 4. Next, on line 2, we sample a complete particle history $\mathbf{r}_{\mathcal{I}_{1:T}}^k$ from the particle filters previously run with the parameter value θ . On line 4, $LS(\cdot)$ denotes the leap-and-shift proposal defined in Algorithm S6 in Section 3 of the Supplementary Material (Sørensen et al., 2025). To avoid introducing another tuning parameter, and to keep the proposal symmetric, we set leap size to 1. The extension to larger leap sizes is straightforward, and we refer to Vitelli et al. (2017, Section 2.4) for details.

Algorithm 4 Rejuvenation Algorithm.

-
- 1: Compute $\hat{\sigma}_\alpha^2 = \frac{1}{R} \sum_{r=1}^R (\alpha^r - \hat{\alpha})^2$ where $\hat{\alpha} = \frac{1}{R} \sum_{r=1}^R \alpha^r$.
 - 2: Sample $k \in [S]$ with probabilities $W_{T, \theta^r}^{1:S}$.
 - 3: **while** stopping criterion not met **do**
 - 4: Sample $\alpha' \sim \log \mathcal{N}(\log \alpha^r, \hat{\sigma}_\alpha^2)$ and $\boldsymbol{\rho}' \sim LS(\boldsymbol{\rho}^r)$ and set $\theta' \leftarrow \{\alpha', \boldsymbol{\rho}', \epsilon\}$.
 - 5: Run a particle filter (Algorithm 1) for $t = 1, 2, \dots, T$ and compute

$$\hat{Z}_T(\theta', \mathbf{r}_{\mathcal{I}_{1:T}}^{1:S}, a_{1:T-1}^{1:S}) = \prod_{t=1}^T \left\{ \frac{1}{S} \sum_{s=1}^S w_{t, \theta'}^s \right\}.$$

- 6: Sample $k' \in [S]$ with probabilities $W_{T, \theta'}^{1:S}$ (from particle filter).
- 7: Set $(\theta^r, k) \leftarrow (\theta', k')$ with probability

$$1 \wedge \frac{\hat{Z}_T(\theta', \mathbf{r}_{\mathcal{I}_{1:T}}^{1:S}, a_{1:T-1}^{1:S})}{\hat{Z}_T(\theta^r, \mathbf{r}_{\mathcal{I}_{1:T}}^{1:S}, a_{1:T-1}^{1:S})} \left(\frac{\alpha'}{\alpha^r} \right)^\gamma \exp\{-\lambda(\alpha' - \alpha^r)\}. \quad (17)$$

- 8: Compute $a = \sum_{n \in \mathcal{I}_{1:t}} \sum_{i=1}^{p_n} g(y_{ni}, \mathbf{r}_n^k)$, $b = \sum_{n \in \mathcal{I}_{1:t}} \sum_{i=1}^{p_n} [1 - g(y_{ni}, \mathbf{r}_n^k)]$, and sample

$$\epsilon' \sim f(\epsilon) \propto \epsilon^{\kappa_1 - 1 + a} (1 - \epsilon)^{\kappa_2 - 1 + b} \mathbf{1}\{\epsilon \in [0, 0.5]\}.$$

- 9: Set $\theta^r \leftarrow \{\alpha^r, \boldsymbol{\rho}^r, \epsilon'\}$.
 - 10: Run Algorithm 2, conditional on $\mathbf{r}_{\mathcal{I}_{1:T}}^k = \{\mathbf{r}_n^k\}_{n \in \mathcal{I}_{1:t}}$, for $t = 1, 2, \dots, T$.
 - 11: Sample $k \in [S]$ with probabilities $W_{T, \theta^r}^{1:S}$ (from conditional particle filter).
 - 12: **end while**
-

On line 5 a new particle filter is run in order to compute the marginal likelihood of the proposed parameters. On line 6 we sample a proposal k' for a new particle history to condition on, using the weights from the particle filter run on line 5. The proposals θ' and k' are accepted with probability given by the Metropolis-Hastings ratio (17) in which the last two terms follow directly from the gamma prior.

On line 8 we sample the error probability from its conditional posterior (Crispino et al., 2019, p. 504). Lines 10 and 11, which consist of running a conditional particle filter and sampling a new $k \in [S]$ are necessary for computing the denominator in (17) in the next iteration.

A reasonable stopping criterion which is computationally easy to check is that the number of unique parameters exceeds some threshold, say $R/2$. However, the Gibbs sampler is guaranteed to produce new values of ϵ^r when we have non-transitive pairwise preferences, and hence we are guaranteed to have R unique particles after a single iteration of the algorithm. If this is sufficient, lines 10-11 can be skipped and no conditional particle filter needs to be run. On the other hand, this may lead to degeneracy in α and $\boldsymbol{\rho}$, so in this case it might be more useful to monitor to the number of unique values of α^r , and stop the rejuvenation when this number exceeds $R/2$.

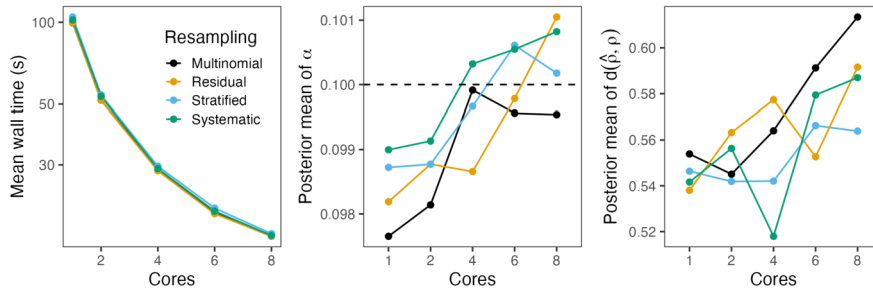


Figure 2: Average wall times (left), average posterior mean of α (center), and average posterior mean of the footrule distance to the true ranking (right), for simulations with complete rankings.

4 Simulation Experiments

We here report results of simulation experiments aimed at testing the proposed algorithms. The R packages Rcpp (Eddelbuettel and François, 2013) and RcppArmadillo (Eddelbuettel and Sanderson, 2014) were used as interfaces to the C++ implementation of our algorithms, which made heavy use of the Armadillo library (Sanderson and Curtin, 2016). Our parallelization of SMC² used the futures framework (Bengtsson, 2021), through the furr package (Vaughan and Dancho, 2021). Pre- and post-processing of data, as well as visualization, was done mainly with the set of R packages provided by the tidyverse (Wickham et al., 2019).

4.1 Complete Rankings

To study the performance of SMC² in a sequential inference case with complete rankings, we generated 100 datasets with complete rankings from the Mallows model with the footrule distance using the sampling algorithm of Vitelli et al. (2017, Appendix C). In all simulations there were $m = 10$ items and $N = 1000$ users and the scale parameter was $\alpha = 0.1$. The users were assumed to enter one at a time, yielding 1,000 timepoints. For all simulated datasets the algorithm was run in parallel on P cores, with $P \in \{1, 2, 4, 6, 8\}$ and $R = 5000/P$ particles, using multinomial, residual, stratified, and systematic resampling. The gamma prior for α had shape $\gamma = 1$ and rate $\lambda = 0.5$ and the resampling threshold was $A = R/2$. Since the likelihood increments are analytically given by (13), the number of particle filters was fixed to 1 and the doubling threshold B in Algorithm 3 set to 0. Simulations were run on a MacBook Pro with a 32 GB Apple M1 Max chip.

The simulation results are summarized in Figure 2. The left plot illustrates how the computing time depends on the resampling scheme and parallelization. Regarding the former, all four resampling schemes performed equally fast. Furthermore, the plot shows a clear benefit of parallelization, although with slightly diminishing returns: doubling the number of cores from 1 to 2, 2 to 4, and 4 to 8, respectively, led to factors 1.93, 1.82,

and 1.76 speed-up, respectively. The average time with eight cores was 16.5 seconds. For comparison, batch estimation with MCMC with a burnin-in period of 1,000 iterations and 5,000 post-burnin iterations took on average 3.2 seconds.

The center plot in Figure 2 shows the posterior means of α for different numbers of cores. MCMC batch estimation had average posterior mean at 0.100 with 95% Monte Carlo interval (0.099, 0.101), suggesting that the posterior mean on average was very close to the data generating value. SMC² had a slight negative bias⁵ when using a single core, with the upper limit of 95% Monte Carlo intervals lower than 0.100. With eight cores there was a tendency towards positive bias, but in this case all the Monte Carlo intervals covered the true value. Finally, the rightmost plot in Figure 2 shows the average posterior mean footrule distance to the true ranking. The differences between cores and resampling methods for the posterior mean of $d(\hat{\rho}, \rho)$ are well within 95% Monte Carlo confidence intervals, which are not included in the plots for ease of visualization.

4.2 Top- k Rankings

We next considered the case in which users provide top-3 rankings of $m = 10$ items. As in Section 4.1, the users were assumed to arrive sequentially, one at each timepoint. A total of 80 random datasets were simulated, with $N = 200$ users, scale parameter $\alpha = 0.3$, and footrule distance. Since each user had only ranked three out of ten items, the particle filter in Algorithm 1 now had to be run to integrate over the remaining seven items for each user. The total number of particles was set to $R = 3000$ which were processed in parallel on 20 cores and combined using (16). The resampling threshold was $A = R/2$ and the threshold for particle filter doubling was set to an average acceptance rate $B = 0.2$ in the rejuvenation step. The initial number of particle filters per core was set to $S = 20$. Both uniform and pseudolikelihood proposals were used for latent rankings, cf. Section 2 of the Supplementary Material (Sørensen et al., 2025). Simulations reported here and in Section 4.3 were run on the high performance computing cluster Fox provided by the University of Oslo.

Table 1 shows simulation results after the final timepoint. The third column shows that the final estimate of α was close to the true value in all cases. With pseudolikelihood proposal, residual resampling gave the lowest posterior mean distance to the true ranking, closely followed by multinomial resampling. With both proposals, systematic resampling gave the highest posterior mean distance to the true ranking. Overall, the statistical performance of uniform and pseudolikelihood proposal seems comparable. Processing a single dataset on eight cores took on average between 14.7 and 18.0 minutes when using uniform proposal, and between 257.3 and 269.2 minutes with pseudolikelihood proposal. In contrast, obtaining 10,000 posterior draws using batch estimation with MCMC took on average five seconds.

Figure 3 shows the number of particle filters per core as a function of time (left) and the resampling probability per core as a function of time (right). The plots for pseudolikelihood proposal were almost identical, and are not shown. The curves were obtained

⁵With bias we mean systematic deviation from the average posterior mean, here computed using MCMC batch estimation with a sufficient number of post burn-in samples. This is not necessarily equal to the data generating value $\alpha = 0.1$, although it was very close in this case.

Proposal	Resampler	α	$d(\boldsymbol{\rho}, \hat{\boldsymbol{\rho}})$	Time (minutes)
Pseudolikelihood	Multinomial	0.302 (4e-04)	3.22 (0.019)	269.2 (1.9)
	Residual	0.304 (3e-04)	3.20 (0.018)	262.3 (1.7)
	Stratified	0.300 (3e-04)	3.42 (0.018)	261.6 (1.7)
	Systematic	0.303 (3e-04)	3.54 (0.020)	257.3 (1.7)
Uniform	Multinomial	0.293 (3e-04)	3.40 (0.017)	15.0 (0.1)
	Residual	0.295 (3e-04)	3.40 (0.017)	14.7 (0.1)
	Stratified	0.296 (3e-04)	3.38 (0.020)	16.0 (0.1)
	Systematic	0.295 (3e-04)	3.62 (0.021)	18.0 (0.2)

Table 1: Results of simulations with sequentially arriving top-3 rankings. Values α and $d(\boldsymbol{\rho}, \hat{\boldsymbol{\rho}})$ are Monte Carlo averages of the posterior means, with standard errors in parentheses.

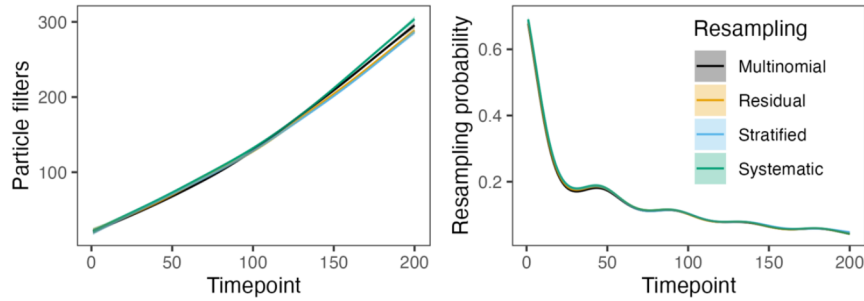


Figure 3: Number of particle filters per core and resampling probability for simulations with sequentially arriving top-3 rankings. Shaded regions are 95% confidence bands.

by fitting generalized additive models (GAMs) with ten thin-plate regression splines (Wood, 2003) as basis functions to the number of particle filters and a binary resampling indicator, respectively. A unit link function was used for the number of particles and a logit link for the resampling indicator, and smoothing was done with restricted maximum likelihood. The left plot shows that the number of particles grows close to linearly. This is expected from Theorem 1 of Andrieu et al. (2010), which implies that the number of particle filters must grow linearly with the number of observations for the acceptance rate to stay constant. The right plot shows that the resampling probability decreases with the number of timepoints. Since each resampling step is followed by one or more rejuvenation steps, this means that as rejuvenation becomes more computationally demanding due to more observations, it also becomes less frequent. This behavior agrees with what is expected from theory; in particular, Theorem 1 and Section 4.3 of Chopin (2002) predicts that the time interval between each time when resampling is needed should increase geometrically in the total number of observations, which would produce an exponentially decaying curve (see also Proposition 17.1 in Chopin and Papaspiliopoulos, 2020).

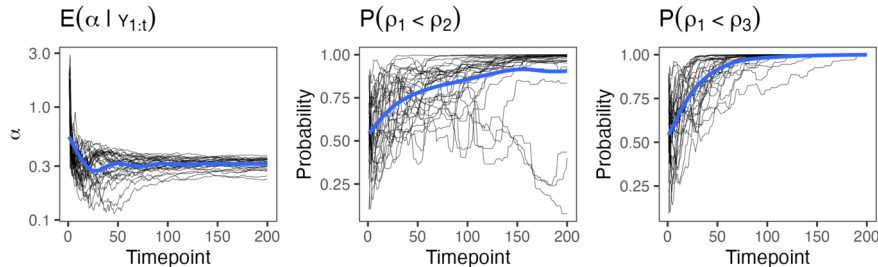


Figure 4: Trace plots of posterior means. Thin black lines shows the posterior probability for a single dataset and thick blue lines are GAM fits.

4.3 Pairwise Preferences

We next simulated consistent pairwise preference data by first generating complete rankings of $m = 5$ items with $\rho = (1, 2, \dots, 5)'$ and $\alpha = 0.3$ for 200 users, and then randomly selecting four implied pairwise preferences for each user. The users were assumed to arrive sequentially, one at each timepoint. The process was repeated 30 times, and each dataset was processed in parallel on 30 central processing units (CPUs), each with $S = 100$ particles. The initial number of particle filters was 20, residual resampling was used, and all other parameters were as described in the previous sections.

Across the 30 datasets, the average posterior mean of α at the final timepoint was 0.329, with 95% Monte Carlo interval (0.327, 0.331). For comparison, the average posterior mean of α obtained by MCMC estimation with 20,000 post-burnin iterations on the same datasets was 0.308 with 95% Monte Carlo interval (0.306, 0.310), suggesting that SMC² was slightly positively biased in this case. The average wall time for computing the posterior for a simulated dataset was 2.0 minutes, compared to 7.3 seconds for batch estimation with MCMC.

Figure 4 (left) shows how the posterior expectation of α evolved as more data became available, with initial rapid fluctuations followed by a stabilization close to the true value of 0.3 after about 100 timepoints. The center plot in Figure 4 shows the posterior probability that item 1 is preferred to item 2 in the modal ranking, and the right plot show the posterior probability that item 1 is preferred to item 3. As expected, the latter converges more quickly to one than the former. In particular, for three of the simulated datasets $P(\rho_1 < \rho_2)$ was below 0.5 even after the final timepoint, implying that items 1 and 2 were “flipped” in the posterior distribution, while $P(\rho_1 < \rho_3)$ was close to 1 for all datasets.

4.4 Timing Comparisons for Sequential Estimation

To compare the relative speed of SMC² and MCMC for sequential estimation, we performed an additional set of experiments in which a set of users were assumed to arrive one at a time, and the posterior had to be updated after the arrival of each new user.

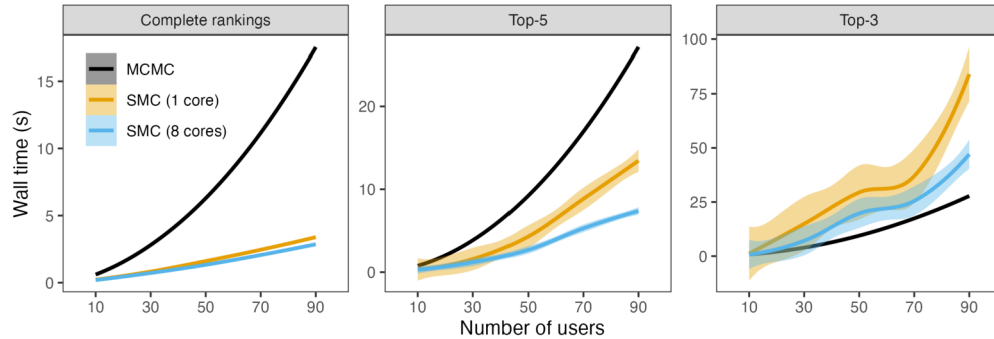


Figure 5: Wall time for sequential estimation with complete rankings (left), top-5 rankings (center), and top-3 rankings (right). Shaded areas are 95% confidence intervals.

For MCMC, this requires rerunning the whole algorithm at each timepoint, whereas for SMC² a single iteration of Algorithm 3 is sufficient.

For a total number of timepoints equal to 10, 30, 50, 70, and 90 we generated 100 random datasets with $m = 10$ items and $\alpha = 0.1$. In addition to the complete rankings case, we also did experiments in which only the top-5 or the top-3 items were retained. For each simulated dataset, 5,000 samples from the posterior distribution were obtained. For MCMC we used a burn-in of 1,000 and no thinning, so that a total of 6,000 iterations were run each time a new observation arrives. We confirmed that the burn-in was appropriate for the data in multiple pilot runs, where we also set the standard deviation of the proposal distribution for α to 0.01 and the leap size of the leap-and-shift proposal for ρ to 1 as these values gave acceptance rates of at least 20% in the pilot runs. For SMC², the number of particles was set to 5,000 which were either processed on a single core or in parallel on eight cores. The initial number of particle filters was set to 1, and the thresholds for resampling and particle filter doubling were set identically to Section 4.2. We emphasize that it is hard to ensure that the effective posterior sample size obtained from SMC² and MCMC are identical and hence obtain decisive evidence of which algorithm is faster. Our goal here was rather to understand the scaling behavior of the respective algorithms for sequential estimation, and how this depends on the amount of missing data.

The results are shown in Figure 5. In all cases, MCMC scaled approximately quadratically. On the other hand, SMC² on a single core and on eight cores scaled close to linearly with complete rankings. In the case with missing rankings, SMC² scaled quadratically, but remained faster than MCMC in the top-5 case. In the top-3 case, however, the computing time for SMC² grew more quickly with the number of users than MCMC. Closer examination of the data revealed that this was due to the particle filter doubling required to properly integrate over the latent variables in these cases.

5 Sequential Analysis of Formula 1 Data

We now analyze the Formula 1 data introduced in Figure 1. As mentioned in the introduction, the rankings were derived from the results of each race. Whenever a driver either was disqualified, did not finish a race, or did not start, the ranking for the particular race was set to missing and assumed to have a higher value than all the ranked drivers. This yielded a missingness proportion of 12.7%. Each race $n \in \{1, \dots, 68\}$ was hence modeled as an assessor yielding a top- k_n ranking, with k_n equal to the number of drivers completing the race.

We computed the posterior distribution of the Bayesian Mallows model with footrule distance sequentially using the SMC² algorithm with $R = 10^5$ particles and $S = 10$ initial particle filters. We used multinomial resampling, uniform latent rank proposal, resampling threshold $A = R/2$, and threshold $B = 0.2$ for particle filter doubling. We confirmed that the number of particle filters was sufficiently large by running the model multiple times with different random number seeds and checking that the posterior quantities of interest remained essentially the same. At the final timepoint, the posterior mean of the scale parameter α was 0.170 with 95% posterior interval (0.165, 0.178).

At each timepoint we computed the cumulative probability (CP) consensus based on the current posterior distribution. The CP consensus (Vitelli et al., 2017, Section 5.1) was computed by first selecting the driver with the highest posterior probability of being ranked first. Then among the remaining drivers we found the driver with the highest posterior probability of being ranked first or second, and so on. The resulting trace plots are shown in Figure 6, in which the solid black lines indicate the CP consensus ranking of each driver and the color scale indicates the posterior probability. Note that by construction, the posterior probability for the driver whose CP consensus equals 16 – the number of items – always has probability 1, since all drivers must be ranked 16 or better. Also note that the true ranking ρ is assumed to be constant, and that Figure 6 shows how our posterior belief about this parameter develops as we obtain more data.

Some interesting features can be seen from Figure 6. First we note that the CP consensus changes quickly during the first ten races, as more data is incorporated into the posterior and the effect of the uniform prior fades. Correspondingly, the probability for the drivers ranked as best after the first rounds is relatively low. For example Charles Leclerc is ranked first in the first eight timepoints, with a probability between 0.16 and 0.76 and Carlos Sainz is ranked second during the first three timepoints, with a probability between 0.29 and 0.65. After the first ten races the consensus stabilizes. For example, Max Verstappen is ranked first for the first time after the eighth race with a probability of 0.91 and retains this rank until the last timepoint with a probability exceeding 0.99 from the thirteenth timepoint and onward. However, there is also movement in the later timepoints; for example, Charles Leclerc climbs from rank 8 at timepoint 54 to rank 4 at the final timepoint.

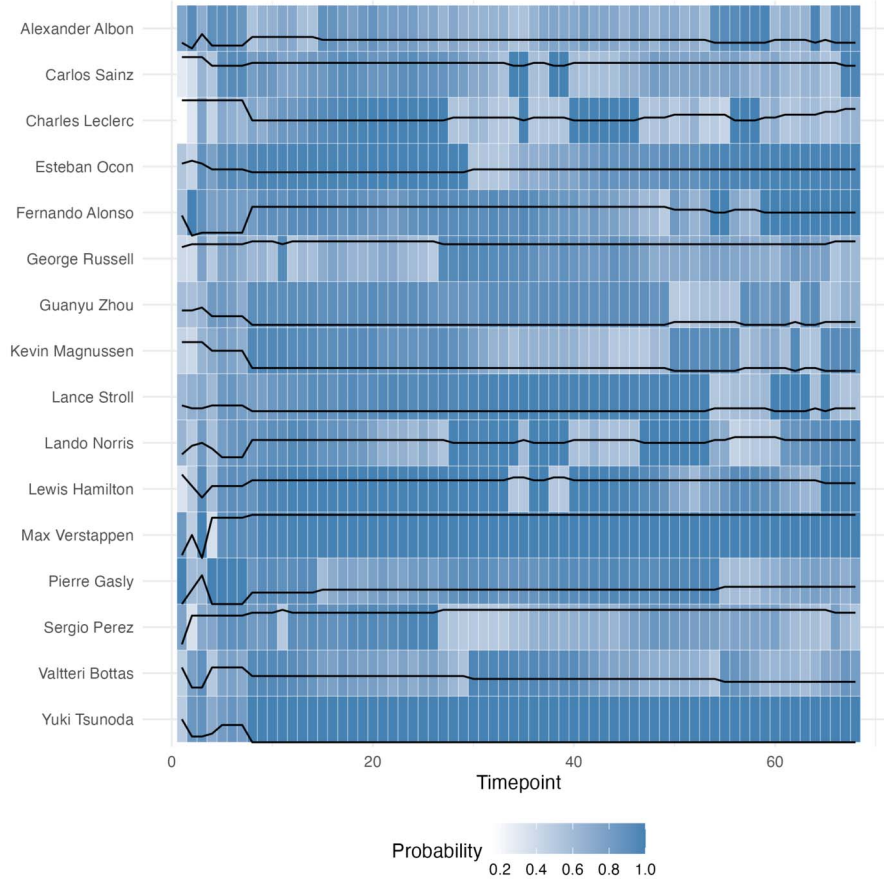


Figure 6: CP consensus over time for Formula 1 drivers. The black lines show the CP consensus ranking, and the color scale shows the posterior probability of having the given ranking or higher.

6 Discussion

We have proposed an SMC² algorithm for sequential estimation of the Bayesian Mallows model. The algorithm naturally incorporates data in the form of partial rankings and both transitive and non-transitive pairwise preferences, and is straightforward to parallelize. Compared to MCMC, the algorithm is competitive in use cases where data arrive sequentially and the posteriors of interest need to be recomputed for each new data batch. In batch estimation problems, on the other hand, MCMC has been faster in all cases considered in this paper.

A number of future extensions are possible. First, conditioning on the full particle history in the conditional particle filter may lead to a high degree of degeneracy (Whiteley, 2010), with the consequence that a large number of particle filters is required to

obtain a sufficiently accurate approximation of the conditional posterior. Backward sampling has been shown to considerably decrease this degeneracy (Lindsten and Schön, 2012), and would be interesting to consider in the current setting. Next, a challenge with SMC² is that its computational complexity grows quadratically with time, as the plot for top-3 rankings in Figure 5 (right) illustrates. A related nested particle filter algorithm which scales linearly with time has been proposed by Crisan and Míguez (2018). Since their algorithm requires the parameters to be real-valued, it is not directly applicable to the Mallows model, but creating such an extension is an interesting problem for future research. Another extension of practical interest is to allow users to provide updated rankings, e.g., comparisons of previously unseen items. Stein (2023, Ch. 6) proposed an algorithm for this in the case of partial rankings, in which users whose new data contradicted their current latent rankings were removed from the pool and then reentered. An extension of this approach to SMC² would likely require a particle filter which performs this type of correction while still yielding unbiased estimates of the marginal likelihood $p(\mathbf{y}_{\mathcal{I},t}|\theta)$, as equation (9) does in the current case where new users arrive at each timepoint. Finally, it may be of interest to let some or all of the parameters in θ depend on time, e.g., to monitor how preferences in a population evolve. An MCMC algorithm for time-varying modal ranking ρ has been proposed by Asfaw et al. (2017), but sequential estimation is likely a good alternative in this case. More recently Piancastelli and Barreto-Souza (2025) have proposed a model for timeseries of rankings, based on the Mallows model. In this model, complete or partial rankings of a set of items are assumed to be observed on a relatively large number of timepoints, and rather than estimating the consensus ranking the focus is on timeseries parameters describing the dynamics by which the modal ranking changes over time.

Another interesting possibility is to use the proposed algorithm in a sequential experimental design framework. For example, at a given timepoint, the items to be ranked or compared by the next user could be determined by a utility function seeking to maximize the information about some posterior quantity of interest, e.g., whether an item A_1 is preferred to another item A_2 in the modal ranking. Examples of similar uses of SMC include estimation of generalized (non-)linear models (Drovandi et al., 2013), model selection (Drovandi et al., 2014), and hierarchical models (McGree et al., 2016).

Acknowledgments

The authors thank Arnaldo Frigessi for discussions and encouragement. Ø.S. thanks Marta Crispino for fruitful discussions about rank modeling.

Supplementary Material

Generalization to Mixtures of Mallows Models and Proposals for Latent Rankings (DOI: [10.1214/25-BA1564SUPP](https://doi.org/10.1214/25-BA1564SUPP); .pdf). The supplementary material shows the extension of the algorithms to mixtures of Mallows models, and discussed in detail proposal distributions for latent rankings.

References

- Andrieu, C., Doucet, A., and Holenstein, R. (2010). “Particle Markov Chain Monte Carlo Methods.” *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 72(3): 269–342. MR2758115. doi: <https://doi.org/10.1111/j.1467-9868.2009.00736.x>. 6, 8, 15
- Asfaw, D., Vitelli, V., Sørensen, Ø., Arjas, E., and Frigessi, A. (2017). “Time-Varying Rankings with the Bayesian Mallows Model.” *Stat*, 6(1): 14–30. MR3599889. doi: <https://doi.org/10.1002/sta4.132>. 20
- Babington Smith, B. (1950). “Discussion of Professor Ross’s Paper.” *Journal of the Royal Statistical Society B*, 12(1): 41–59. 1
- Bengtsson, H. (2021). “A Unifying Framework for Parallel and Distributed Processing in R Using Futures.” *R Journal*. 13
- Berzuini, C. and Gilks, W. (2001). “RESAMPLE-MOVE Filtering with Cross-Model Jumps.” In Doucet, A., de Freitas, N., and Gordon, N. (eds.), *Sequential Monte Carlo Methods in Practice*, Statistics for Engineering and Information Science, 117–138. New York, NY: Springer. MR1847789. 2, 6
- Bradley, R. A. and Terry, M. E. (1952). “Rank Analysis of Incomplete Block Designs: I. The Method of Paired Comparisons.” *Biometrika*, 39(3/4): 324–345. MR0070925. doi: <https://doi.org/10.2307/2334029>. 1
- Ceberio, J., Irurozki, E., Mendiburu, A., and Lozano, J. A. (2015). “A Review of Distances for the Mallows and Generalized Mallows Estimation of Distribution Algorithms.” *Computational Optimization and Applications*, 62(2): 545–564. MR3406686. doi: <https://doi.org/10.1007/s10589-015-9740-x>. 6
- Chopin, N. (2002). “A Sequential Particle Filter Method for Static Models.” *Biometrika*, 89(3): 539–552. MR1929161. doi: <https://doi.org/10.1093/biomet/89.3.539>. 9, 15
- Chopin, N., Jacob, P. E., and Papaspiliopoulos, O. (2013). “SMC2: An Efficient Algorithm for Sequential Analysis of State Space Models.” *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 75(3): 397–426. MR3065473. doi: <https://doi.org/10.1111/j.1467-9868.2012.01046.x>. 2, 6, 8, 9, 11
- Chopin, N. and Papaspiliopoulos, O. (2020). *An Introduction to Sequential Monte Carlo*. Springer Series in Statistics. Cham: Springer International Publishing. MR4215639. doi: <https://doi.org/10.1007/978-3-030-47845-2>. 15
- Crisan, D. and Míguez, J. (2018). “Nested particle filters for online parameter estimation in discrete-time state-space Markov models.” *Bernoulli*, 24(4A): 3039–3086. Publisher: Bernoulli Society for Mathematical Statistics and Probability. URL <https://projecteuclid.org/journals/bernoulli/volume-24/issue-4A/Nested-particle-filters-for-online-parameter-estimation-in-discrete-time/10.3150/17-BEJ954.full> MR3779710. doi: <https://doi.org/10.3150/17-BEJ954>. 20

- Crispino, M. (2018). “Bayesian Learning with the Mallows Rank Model.” Ph.D. thesis, Bocconi University. 6
- Crispino, M., Arjas, E., Vitelli, V., Barrett, N., and Frigessi, A. (2019). “A Bayesian Mallows Approach to Nontransitive Pair Comparison Data: How Human Are Sounds?” *The Annals of Applied Statistics*, 13(1): 492–519. MR3937438. doi: <https://doi.org/10.1214/18-AOAS1203>. 3, 4, 5, 12
- Crispino, M., Mollica, C., Astuti, V., and Tardella, L. (2023). “Efficient and Accurate Inference for Mixtures of Mallows Models with Spearman Distance.” *Statistics and Computing*, 33(5): 98. MR4613471. doi: <https://doi.org/10.1007/s11222-023-10266-8>. 6
- Dai, C., Heng, J., Jacob, P. E., and Whiteley, N. (2022). “An Invitation to Sequential Monte Carlo Samplers.” *Journal of the American Statistical Association*, 117(539): 1587–1600. MR4480734. doi: <https://doi.org/10.1080/01621459.2022.2087659>. 6, 10
- Del Moral, P. (2004). *Feynman-Kac Formulae*. Probability and Its Applications. New York, NY: Springer. MR2044973. doi: <https://doi.org/10.1007/978-1-4684-9393-1>. 8
- Diaconis, P. (1988). *Group Representations in Probability and Statistics*. SPIE. MR0964069. 2, 3, 6
- Douc, R. and Cappe, O. (2005). “Comparison of Resampling Schemes for Particle Filtering.” In *ISPA 2005. Proceedings of the 4th International Symposium on Image and Signal Processing and Analysis, 2005.*, 64–69. 8
- Drovandi, C. C., McGree, J. M., and Pettitt, A. N. (2013). “Sequential Monte Carlo for Bayesian Sequentially Designed Experiments for Discrete Data.” *Computational Statistics & Data Analysis*, 57(1): 320–335. MR2981091. doi: <https://doi.org/10.1016/j.csda.2012.05.014>. 20
- Drovandi, C. C., McGree, J. M., and Pettitt, A. N. (2014). “A Sequential Monte Carlo Algorithm to Incorporate Model Uncertainty in Bayesian Sequential Design.” *Journal of Computational and Graphical Statistics*, 23(1): 3–24. MR3173758. doi: <https://doi.org/10.1080/10618600.2012.730083>. 20
- Eddelbuettel, D. and François, R. (2013). *Seamless R and C++ Integration with Rcpp*. New York, NY: Springer, 1 edition. 13
- Eddelbuettel, D. and Sanderson, C. (2014). “RcppArmadillo: Accelerating R with High-Performance C++ Linear Algebra.” *Computational Statistics & Data Analysis*, 71: 1054–1063. MR3132026. doi: <https://doi.org/10.1016/j.csda.2013.02.005>. 13
- Eliseussen, E., Fleischer, T., and Vitelli, V. (2022). “Rank-Based Bayesian Variable Selection for Genome-Wide Transcriptomic Analyses.” *Statistics in Medicine*, 41(23): 4532–4553. MR4488321. doi: <https://doi.org/10.1002/sim.9524>. 1
- Fligner, M. A. and Verducci, J. S. (1986). “Distance Based Ranking Models.” *Journal of*

- the Royal Statistical Society. Series B (Methodological)*, 48(3): 359–369. MR0876847. 2, 6
- Fulop, A. and Li, J. (2013). “Efficient Learning via Simulation: A Marginalized Resample-Move Approach.” *Journal of Econometrics*, 176(2): 146–161. MR3084050. doi: <https://doi.org/10.1016/j.jeconom.2013.05.002>. 2, 6, 9, 11
- Hol, J. D., Schon, T. B., and Gustafsson, F. (2006). “On Resampling Algorithms for Particle Filters.” In *2006 IEEE Nonlinear Statistical Signal Processing Workshop*, 79–82. 8
- Hwang, M., Lee, G., Kee, H., Kim, C. W., Lee, K., and Oh, S. (2023). “Sequential Preference Ranking for Efficient Reinforcement Learning from Human Feedback.” *Advances in Neural Information Processing Systems*, 36: 49088–49099. 1
- Irurozki, E. (2014). “Sampling and Learning Distance-Based Probability Models for Permutation Spaces.” Ph.D. thesis, Department of Computer Science and Artificial Intelligence of the University of the Basque Country. 6
- Irurozki, E., Calvo, B., and Lozano, J. A. (2016). “PerMallows: An R Package for Mallows and Generalized Mallows Models.” *Journal of Statistical Software*, 71: 1–30. 6
- Irurozki, E., Calvo, B., and Lozano, J. A. (2018). “Sampling and Learning Mallows and Generalized Mallows Models Under the Cayley Distance.” *Methodology and Computing in Applied Probability*, 20(1): 1–35. MR3760337. doi: <https://doi.org/10.1007/s11009-016-9506-7>. 2, 6
- Irurozki, E., Calvo, B., and Lozano, J. A. (2019). “Mallows and Generalized Mallows Model for Matchings.” *Bernoulli*, 25(2): 1160–1188. MR3920369. doi: <https://doi.org/10.3150/17-bej1017>. 5
- Jun, S.-h., Wang, L., and Bouchard-côté, A. (2012). “Entangled Monte Carlo.” In *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc. 10
- Kahneman, D. and Tversky, A. (1979). “Prospect Theory: An Analysis of Decision under Risk.” *Econometrica*, 47(2): 263–291. MR3618580. doi: <https://doi.org/10.2307/1914185>. 4
- Kamishima, T. (2003). “Nantonac Collaborative Filtering: Recommendation Based on Order Responses.” In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '03, 583–588. New York, NY, USA: Association for Computing Machinery. 1
- Kitagawa, G. (1996). “Monte Carlo Filter and Smoother for Non-Gaussian Nonlinear State Space Models.” *Journal of Computational and Graphical Statistics*, 5(1): 1–25. MR1380850. doi: <https://doi.org/10.2307/1390750>. 8
- Lindsten, F. and Schön, T. B. (2012). “On the Use of Backward Simulation in the Particle Gibbs Sampler.” In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 3845–3848. 20

- Liu, J. S. and Chen, R. (1998). “Sequential Monte Carlo Methods for Dynamic Systems.” *Journal of the American Statistical Association*, 93(443): 1032–1044. [MR1649198](#). doi: <https://doi.org/10.2307/2669847>. 8
- Liu, Q., Crispino, M., Scheel, I., Vitelli, V., and Frigessi, A. (2019a). “Model-Based Learning from Preference Data.” *Annual Review of Statistics and Its Application*, 6(1): 329–354. [MR3939524](#). doi: <https://doi.org/10.1146/annurev-statistics-031017-100213>. 2
- Liu, Q., Reiner, A. H., Frigessi, A., and Scheel, I. (2019b). “Diverse Personalized Recommendations with Uncertainty from Implicit Preference Data with the Bayesian Mallows Model.” *Knowledge-Based Systems*, 186: 104960. 1, 2
- Lu, T. and Boutilier, C. (2014). “Effective Sampling and Learning for Mallows Models with Pairwise-Preference Data.” *Journal of Machine Learning Research*, 15(117): 3963–4009. [MR3317212](#). 2
- Luce, R. D. (1959). *Individual Choice Behavior*. Individual Choice Behavior. Oxford, England: John Wiley. [MR0108411](#). 1
- Mallows, C. L. (1957). “Non-Null Ranking Models. I.” *Biometrika*, 44(1/2): 114–130. [MR0087267](#). doi: <https://doi.org/10.1093/biomet/44.1-2.114>. 2, 3
- Marden, J. I. (1995). *Analyzing and Modeling Rank Data*, volume 64 of *Monographs on Statistics and Applied Probability*. Cambridge, MA, USA: Chapman & Hall. [MR1346107](#). 2, 5
- McGree, J. M., Drovandi, C. C., White, G., and Pettitt, A. N. (2016). “A Pseudo-Marginal Sequential Monte Carlo Algorithm for Random Effects Models in Bayesian Sequential Design.” *Statistics and Computing*, 26(5): 1121–1136. [MR3535048](#). doi: <https://doi.org/10.1007/s11222-015-9596-z>. 20
- Meila, M. and Bao, L. (2010). “An Exponential Model for Infinite Rankings.” *Journal of Machine Learning Research*, 11(113): 3481–3518. [MR2756191](#). 2
- Mendes, E. F., Carter, C. K., Gunawan, D., and Kohn, R. (2020). “A Flexible Particle Markov Chain Monte Carlo Method.” *Statistics and Computing*, 30(4): 783–798. [MR4108677](#). doi: <https://doi.org/10.1007/s11222-019-09916-7>. 7, 11
- Mukherjee, S. (2016). “Estimation in Exponential Families on Permutations.” *The Annals of Statistics*, 44(2): 853–875. [MR3476619](#). doi: <https://doi.org/10.1214/15-AOS1389>. 6
- Murray, L. M., Lee, A., and Jacob, P. E. (2016). “Parallel Resampling in the Particle Filter.” *Journal of Computational and Graphical Statistics*, 25(3): 789–805. [MR3533638](#). doi: <https://doi.org/10.1080/10618600.2015.1062015>. 10
- Naesseth, C. A., Lindsten, F., and Schön, T. B. (2019). “Elements of Sequential Monte Carlo.” *Foundations and Trends® in Machine Learning*, 12(3): 307–392. 10, 11
- Nicholls, G. K., Lee, J. E., Karn, N., Johnson, D., Huang, R., and Muir-Watt, A. (2022). “Bayesian Inference for Partial Orders from Random Linear Extensions:

- Power Relations from 12th Century Royal Acta.” <https://arxiv.org/abs/2212.05524v2>. MR4912950. doi: <https://doi.org/10.1214/24-aos2002>. 1
- Pearce, M. and Erosheva, E. A. (2022). “A Unified Statistical Learning Model for Rankings and Scores with Application to Grant Panel Review.” *Journal of Machine Learning Research*, 23(210): 1–33. MR4577163. 1
- Piancastelli, L. and Barreto-Souza, W. (2025). “Time Series Analysis of Rankings: A GARCH-Type Approach.” [arXiv:2502.05102](https://arxiv.org/abs/2502.05102) [stat]. URL <http://arxiv.org/abs/2502.05102> 20
- Plackett, R. L. (1975). “The Analysis of Permutations.” *Journal of the Royal Statistical Society Series C: Applied Statistics*, 24(2): 193–202. MR0391338. doi: <https://doi.org/10.2307/2346567>. 1
- Rojas-Delgado, J., Ceberio, J., Calvo, B., and Lozano, J. A. (2022). “Bayesian Performance Analysis for Algorithm Ranking Comparison.” *IEEE Transactions on Evolutionary Computation*, 26(6): 1281–1292. 1
- Sanderson, C. and Curtin, R. (2016). “Armadillo: A Template-Based C++ Library for Linear Algebra.” *Journal of Open Source Software*, 1(2): 26. 13
- Sloane, N. J. A. (2023). “The Encyclopedia of Integer Sequences.” 6
- Sørensen, Ø., Crispino, M., Liu, Q., and Vitelli, V. (2020). “BayesMallows: An R Package for the Bayesian Mallows Model.” *The R Journal*, 12(1): 324–342. 6
- Sørensen, Ø., Stein, A., Netto, W. L., and Leslie, D. S. (2025). “Supplement to “Sequential Rank and Preference Learning with the Bayesian Mallows Model”.” *Bayesian Analysis*. doi: <https://doi.org/10.1214/25-BA1564SUPP>. 3, 8, 11, 14
- Stein, A. (2023). “Sequential Inference with the Mallows Model.” Ph.D. thesis, Lancaster University. MR4691108. 2, 6, 8, 20
- Thurstone, L. L. (1927). “A Law of Comparative Judgment.” *Psychological Review*, 34(4): 273–286. 1
- Vaughan, D. and Dancho, M. (2021). *furrr: Apply Mapping Functions in Parallel Using Futures*. 13
- Vitelli, V., Sørensen, Ø., Crispino, M., Frigessi, A., and Arjas, E. (2017). “Probabilistic Preference Learning with the Mallows Rank Model.” *The Journal of Machine Learning Research*, 18(1): 5796–5844. MR3813807. 2, 3, 5, 6, 11, 13, 18
- Whiteley, N. (2010). “Discussion on Particle Markov Chain Monte Carlo Methods.” *Journal of the Royal Statistical Society: Series B*, 72(3): 306–307. 19
- Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L. D., François, R., Grolemond, G., Hayes, A., Henry, L., Hester, J., Kuhn, M., Pedersen, T. L., Miller, E., Bache, S. M., Müller, K., Ooms, J., Robinson, D., Seidel, D. P., Spinu, V., Takahashi, K., Vaughan, D., Wilke, C., Woo, K., and Yutani, H. (2019). “Welcome to the Tidyverse.” *Journal of Open Source Software*, 4(43): 1686. 13

- Wood, S. N. (2003). “Thin Plate Regression Splines.” *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 65(1): 95–114. MR1959095. doi: <https://doi.org/10.1111/1467-9868.00374>. 15
- Yi, I. (2021). “Which Firms Require More Governance? Evidence from Mutual Funds’ Revealed Preferences.” 1
- Yu, P. L. H., Gu, J., and Xu, H. (2019). “Analysis of Ranking Data.” *WIREs Computational Statistics*, 11(6): e1483. MR4021451. doi: <https://doi.org/10.1002/wics.1483>. 2