Sequential Inference with the Mallows Model

Anja Katrina Stein, B.Sc.(Hons.), M.Res

Lancaster University

Submitted for the degree of Doctor of Philosophy at Lancaster University.

May 2023

STOR-i
excellence with impact

Abstract

The Mallows model is a widely used probabilistic model for analysing rank data. It assumes that a collection of n items can be ranked by each assessor and then summarised as a permutation of size n. The associated probability distribution is defined on the permutation space of these items. A hierarchical Bayesian framework for the Mallows model, named the Bayesian Mallows model, has been developed recently to perform inference and to provide uncertainty estimates of the model parameters. This framework typically uses Markov chain Monte Carlo (MCMC) methods to simulate from the target posterior distribution. However, MCMC can be considerably slow when additional computational effort is presented in the form of new ranking data. It can therefore be difficult to update the Bayesian Mallows model in real time.

This thesis extends the Bayesian Mallows model to allow for sequential updates of its posterior estimates each time a collection of new preference data is observed. The posterior is updated over a sequence of discrete time steps with fixed computational complexity. This can be achieved using Sequential Monte Carlo (SMC) methods. SMC offers a standard alternative to MCMC by constructing a sequence of posterior distributions using a set of weighted samples. The samples are propagated via a combination of importance sampling, resampling and moving steps.

We propose an SMC framework that can perform sequential updates for the posterior distribution for both a single Mallows model and a Mallows mixture each time we observe new full rankings in an online setting. We also construct a framework to conduct SMC with partial rankings for a single Mallows model. We propose an alternative proposal distribution for data augmentation in partial rankings that incorporates the current posterior estimates of the Mallows model parameters in each SMC iteration. We also extend the framework to consider how the posterior is updated when known assessors provide additional information in their partial ranking. We show how these corrections in the latent information are performed to account for the changes in the posterior.

Acknowledgements

First, I want to thank everyone at STOR-i, past and present. I feel very privileged to have been able to complete my thesis in a warm and friendly environment and to have been a student at this CDT. Special thanks go to the STOR-i admin team: Jen, Kim, Nicky and Wendy. Your hard work doesn't go unnoticed and STOR-i would be a mess without you.

Before starting the STOR-i programme, my final year as an undergraduate student was a testing time, to say the least, and full of anxiety about how to get to where I wanted to be. I'd like to thank my group project supervisor Ben for advocating for me and encouraging me to pursue research. Thank you to those who told me to take the leap of faith when I had doubts about accepting my place on the programme. I'm glad I did.

I'd like to thank my STOR-i cohort for providing a lot of fun and laughter as we dealt with the pressure and heavy workload in the MRes year and for the support we had for each other as we all embarked on our PhD journeys. You all made my time at Lancaster a wonderful experience and I hope that we will continue to keep in touch.

I'm grateful to my two supervisors David and Arnoldo for their support and guidance throughout my PhD. I'll remember to not run before I can walk when tackling any research problem. Thank you to Dan, Øystein and Waldir for the collaboration that we conducted in partnership with Lancaster and Oslo University. It's a real shame that I didn't get to meet you in person, Øystein and Waldir, but it's been a real pleasure working with you. I'm also grateful for the financial support provided by the EPSRC for my PhD project.

I'd also like to thank Louise, Lucy, Sam and Xiaoyun from the Women's writing group at Lancaster University. We held each other accountable as we wrote our theses remotely from our homes and we were there for each other to share the highs, the lows and the challenges that life can throw at you. I am amazed by the strength and motivation that each of you possess.

Thank you to my friends and family for their understanding and patience over the years, particularly during the writing-up phase, and for reminding me that there is a life outside of the PhD.

I'm grateful to Holly and Srshti for their encouragement, for being my confidants and for being by my side, both physically in the office and in spirit. Your friendships are invaluable.

Finally, thank you, Rob, for your love, patience and reassurance when I thought I'd never complete the PhD. You're alright as well.

Declaration

I declare that the work in this thesis has been done by myself and has not been submitted elsewhere for the award of any other degree. The word count of this thesis is approximately 52,000 words.

Anja Katrina Stein

Contents

Chapter 1

Introduction

Preference data, or ranking data, occur when an assessor judges a collection of items based on their personal preference or concerning some preference criterion. For example, we may rank a group of candidates in an election in terms of suitability for a role (Diaconis, 1988; Murphy and Martin, 2003). When handling lafrge volumes of preference data, which are typically incomplete, we may be interested in summarising the collected data in a meaningful way. This includes: finding the shared consensus on the preference for a set of items amongst a group of individuals; predicting how an individual may rate a particular item; or grouping a large group of individuals with different preferences into several smaller groups who are like-minded with similar preferences. These tasks come under the field of preference learning. The challenge in preference learning is to be able to achieve one or more of these aims given the nature and availability of the preference information observed from a group of individuals. As a result, many probabilistic models have been proposed to model the different types of preference data. These can be grouped into several classes: order statistics models (Thurstone, 1927); paired comparison models (Smith, 1950); distance-based models (Diaconis, 1988) and multistage model (Luce, 1959; Plackett, 1975).

This thesis focuses on one particular example in the distance-based model class called the Mallows model (Mallows, 1957). The Mallows model assumes that a set of n distinct items can be compared to each other by an assessor as an overall set and then summarised as a permutation of size n called a ranking. Its probabilistic density is defined on the permutation space of n items and assumes that the probability of observing a ranking decays as the distance between the ranking and the consensus ranking increases.

Several methodological developments, both frequentist and Bayesian, have been made to the Mallows model to perform inference using different types of preference data. However, a lot of the literature surrounding the Mallows model poses similar limitations. One of the main issues is that the proposed methods lack the flexibility to handle different kinds of preference data and heterogeneous data. This key issue has been mitigated using a hierarchical Bayesian model, named the Bayesian Mallows model (Vitelli et al., 2018), which uses a Markov chain Monte Carlo scheme to sample and learn the posterior distribution. Several extensions of this particular model have also been proposed to consider different scenarios with preference data, including clicking data in a recommender system style setting (Crispino et al., 2016), time-varying rankings (Asfaw et al., 2017) and intransitive pairwise comparisons (Liu et al., 2019b).

The extension we propose for the Bayesian Mallows Model addresses the following research question: can we update the posterior distribution sequentially each time we receive new information effectively? If we have a large number of assessors and/or items, we want to be able to update the posterior distribution sequentially each time we receive new information quickly, otherwise the posterior may not model the existing preference data correctly as we perform inference. A possible option is to rerun the MCMC algorithm from the beginning for every new piece of preference information observed each time. However, this process is time-consuming if it is conducted frequently with a high volume of data; MCMC has the typical setback of not knowing how long to run the algorithm to achieve convergence. This motivates the need to reduce computational cost. In this thesis, we attempt to develop a framework for performing sequential posterior updates for the Mallows model using sequential Monte Carlo (SMC) methods that will reduce the computational cost. No previous attempts have been made to develop an SMC framework for the Bayesian Mallows model as a viable alternative to MCMC sampling techniques.

This thesis is organised as follows. Chapter 2 reviews some of the attempts that have been made to create preference learning models in each class. These models are created based on the assumptions of how the preference data is created. We also see the links that are shared between the preference learning model classes. Chapter 3 studies Monte Carlo methods to perform Bayesian inference and the classical clustering problem. We will see how SMC methods have evolved over nearly three decades. Chapter 4 introduces the proposed SMC framework for approximating the Bayesian Mallows model. We consider the simplest case of the main research question in this thesis; when presented with a new collection of full rankings for a set of items over time, can we perform sequential updates on the posterior estimate in an effective way compared to using typical MCMC methods? Chapter 5 builds upon Chapter 4 by considering the same problem to handle new partial rankings over time. We propose an alternative method for performing data augmentation on the missing components of each partial ranking by considering the current parameter estimates of the posterior. We investigate if this informed approach to data augmentation performs better than augmenting the missing data via uniform sampling. We also consider performing sequential updates of the posterior estimate each time we receive an updated partial ranking from an existing assessor in Chapter 6. We discuss the theory for how to perform these updates when we need to account for the potential corrections that need to be made. Chapter 7 extends the SMC framework to clustering full rankings and we see if we can perform sequential clustering for a mixture model with a known number of components. The SMC framework described in Chapters 4-7 is then assessed using simulated data and real data sets against the MCMC algorithm for the Bayesian Mallows model of Vitelli et al. (2018).

Chapter 2

Probabilistic Models for Preference Data

2.1 Introduction

Preference data are a type of qualitative data, that occur when an individual is asked to judge a collection of items based on their personal preference or with respect to some preference criterion. These data can be presented in a variety of formats; they can be observed implicitly and explicitly and they are ubiquitous in modern society. Several examples of preference data can be found in: education, where prospective Irish college students are requested to select up to ten courses in order of preference in their application (Gormley and Murphy, 2006); elections, such as ranking five candidates for the American Psychological Association (Diaconis, 1988; Murphy and Martin, 2003); bioinformatics, where we might be interested in finding a consensus list from several independent studies which measure the level of differential gene expression between two conditions for a set of genes tested (Vitelli et al., 2018); sports tournaments, such as NASCAR stock car racing in the United States where drivers participate in multiple races and their finishing position in each race may be considered a ranking (Hunter, 2004; Caron and Doucet, 2012); music, where we might be asked to listen to a series of electronically synthesised sounds in pairs and select the sound that was perceived as more human (Crispino et al., 2019). The recent development of social media and online recommender systems means that preference data can now be collected in abundance. This means that typical ranking problems now face the additional challenge of handling many individuals' preferences for a large number of items, creating computationally difficult problems.

2.2 Aims of preference learning

The challenges we face in handling large volumes of preference data require methods from the field of preference learning. Preference learning aims to create a predictive model by utilising the data in a meaningful way such that we can summarise a group of individuals' preferences. Some of the aims of preference learning, as described in Vitelli et al. (2018) and Liu et al. (2019a), include:

- **Rank aggregation**: this involves summarising a group of individuals' preferences, assuming that the group is homogeneous where the preferences made by each individual are similar amongst the group, by providing a shared consensus preference of the items among the group.

- **Individual preference learning**: if an individual's preferences are incomplete, that is, they specify their preferences for a subset of items, then we may be interested in inferring the missing information for that individual.

- **Clustering**: a group of individuals may not share a common preference for a set of items, even if each individual provides incomplete preferences. Instead, we may find that preferences amongst the group are heterogeneous, where there are several different levels of similarity among the group's preferences. This motivates the need to divide the group's preferences into several smaller groups of homogeneous preferences. The challenge is to determine the number of components and then estimate the model parameters for each component. If appropriate, we might also want to perform individual preference learning.

2.3 Preference data

The challenge in preference learning is to be able to achieve one or more of these aims given the nature and availability of the preference information observed from a group of individuals. Typically, preference information can be collected in one of two ways; it can be provided directly from each individual, or it can be inferred through user interactions if we are collecting data through websites. We note that preference data can appear in a variety of formats, but the challenges we face with preference learning remain the same. We now introduce the notation for some forms of preference data; in this thesis will consider preference data in the form of rankings.

2.3.1 Implicit data

Implicit data are a form of feedback collected from an individual without having to ask them directly to specify a preference. Examples of implicit data are purchase history, watch times of media, and the number of views of a particular item. This feedback can be represented as binary variables; usually, the value 1 indicates that a user liked an item, while a value of 0 indicates that a user disliked an item. Instead of using binary variables, we can collect the frequencies of user interactions, for example, by counting the number of views for a particular item. This type of information is easy to collect in vast quantities and can help companies to gain a better understanding of their customers and to improve their recommendations to existing and future customers. However, this type of information does provide a level of uncertainty when modelling as the perceived preference the individual has for an item might not necessarily reflect the magnitude of the individual's preference.

2.3.2 Explicit data

Explicit data occur when an assessor gives a clear indication of how much they prefer an item or a collection of items by using a numerical value to indicate the magnitude of preference. Two common examples are providing a rating for a film out of 5 stars or assigning a score for a competitor's performance in a sport such as gymnastics.

2.3.3 Pairwise comparisons

An individual does not necessarily have to provide a rating for a single item to indicate a preference; they can also compare items as pairwise comparisons. This occurs when an individual is presented with a pair of distinct items and is asked to select the item in the pair that they prefer based on some attribute when making the comparison. This is a binary choice, and no pair of items can be considered equally preferred. We can express a pairwise comparison formally using the following notation. Given two distinct items A_i and A_j $(i \neq j)$, a pairwise comparison can be expressed as $A_i \prec A_j$, where $\prec$ means "is preferred over". Typically, we assume that the pairwise comparisons we obtain from a group of individuals are transitive, i.e., that each individual does not contradict themselves when specifying their preferences. For example, for three distinct items $\{A_i, A_j, A_k\}$ $(i \neq j \neq k)$, if $A_i \prec A_j$ and $A_j \prec A_k$, then it must follow that $A_i \prec A_k$.

2.3.4 Rankings

We can also compare items against each other in a set, rather than in pairs. If we are given a set of n distinct items, $\mathcal{A} = \{A_1, \ldots, A_n\}$, we can express a preference order for them concerning some attribute to create a ranking. A ranking, represented formally as a permutation $\mathbf{R} = (R_1, \ldots, R_n)$, is defined as a mapping $\mathbf{R} : \mathcal{A} \to \mathcal{P}_n$, where $\mathcal{P}_n$ is the space of permutations of dimension n of ranks $\{1, \ldots, n\}$. Each $R_i \in \{1, \ldots, n\}$ corresponds to the rank of an item $A_i \in \mathcal{A}$. There are $|\mathcal{P}_n| = n!$ possible permutations for ranking n items in $\mathcal{A}$. Conventionally, the item most preferred in

$\mathcal{A}$ has rank 1, while the least preferred item has rank n. For example, given a set $\mathcal{A} = \{A_1, A_2, A_3, A_4, A_5\}$ consisting of $n = 5$ items, then one possible ranking is,

$$\mathbf{R} = (2, 4, 5, 1, 3), \tag{2.3.1}$$

where $\mathbf{R}(A_2) = R_2 = 4$ denotes that item A_2 has rank 4.

Pairwise comparisons share a relationship with full rankings. Given a complete ranking $\mathbf{R} \in \mathcal{P}_n$, we can deduce all possible pairings between items, of which there are $\binom{n}{2} = \frac{n(n-1)}{2}$ in total, by the following rule:

$$(A_i \prec A_j) \quad \Longleftrightarrow \quad R_i < R_j, \quad i, j \in \{1, \ldots, n\}, \quad i \neq j.$$

In other words, an item with a higher rank (indicated by a low integer value for the rank) is preferred to items with lower ranks.

We can also interpret preference information for a set of items as an ordering. An ordering is the inverse of a ranking, that is, $\mathbf{X} = \mathbf{R}^{-1} = (X_1, \ldots, X_n)$ where each X_i represents the item assigned the rank i in the order. Each ordering belongs to the set of permutations of the labels in $\mathcal{A}$, denoted as $\mathcal{X}$. So the ranking given in (2.3.1) has the corresponding ordering

$$\mathbf{X} = \mathbf{R}^{-1} = (4, 1, 5, 2, 3),$$

where $\mathbf{X}(2) = X_2 = \mathbf{R}^{-1}(2) = 1$ means that the item of rank 2 is the item A_1.

Often, if n is large, then a subset of items is ranked by each individual to give a partial ranking. The subset of items ranked by each individual is often their most preferred items and rankings deduced in this manner are also referred to as top-k rankings, where $k < n$. Partial rankings can also be generalised so that assessors do not have to provide a top-k ranking for the same value of k. So for M assessors, each individual can give their top-k_j items, $k_j < n,\ j \in \{1, \ldots, M\}$. We may also have partial rankings in which, for some unspecified reason, we have some random missing information from an individual's full ranking, resulting in a partial ranking.

2.4 Probabilistic models for preference data

Many probabilistic models for preference data have been proposed in the statistical and psychological literature. These are placed into four classes (Fligner and Verducci, 1986): order statistics models; paired comparison models; distance-based models and multistage models. Several review papers discuss these models and their recent developments, including Fligner and Verducci (1986); Marden (1995); Yu et al. (2019) and Liu et al. (2019a). We describe each class and provide some examples of the preference models that belong to each of them. We also discuss the links that some models share between classes.

2.4.1 Order statistics models

Order statistics models (Thurstone, 1927) assume that an ordering for a set of items can be inferred based on the scores each individual assigns to each item. The scores for each item are not observed and are treated as random variables. The relative ordering of these variables determines an individual's ordering of the items. Thurstone (1927) proposed a model to describe this behaviour and introduced the Law of Comparative Judgement. As summarised by Liu et al. (2019a), each item $A_i \in \mathcal{A},\ i \in 1 \ldots n$ has a random score variable $Y_1, \ldots, Y_n$ that is distributed according to F_i. The probability of observing a ranking $\mathbf{R}$ in an order statistics model is

$$p(\mathbf{R}) = P(Y_{X_1} > Y_{X_2} > \cdots > Y_{X_n}), \quad \mathbf{R} \in \mathcal{P}_n. \tag{2.4.1}$$

An alternative way to view the probability of observing the same ranking $\mathbf{R}$ is the following,

$$p(\mathbf{R}) = P\Big(Y_{R^{-1}(1)} > Y_{R^{-1}(2)} > \cdots > Y_{R^{-1}(n)}\Big), \quad \mathbf{R} \in \mathcal{P}_n, \tag{2.4.2}$$

where Y_i is the latent score of the item A_i. Note that $\mathbf{R}^{-1}(1)$ is the item ranked first and hence it should have the largest score. A simplification of (2.4.1) and (2.4.2) is to define each score variable as $Y_i = \mu_i + \epsilon_i$ where μ_i and ϵ_i are the mean score and the error for the item A_i, respectively, so that $F_i(y) = F(y - \mu_i)$. To deduce a consensus ranking of the items in a group of individuals, the aim is to estimate each μ_i, and then a ranking can be inferred by ordering the items in the descending order of the values of μ_i.

Thurstone model and extensions

Several cumulative distribution functions for F have been discussed in the literature on preference learning. One example of an order statistics model is Thurstone's model (Thurstone, 1927) where each F_i is assumed to follow a Gaussian distribution. Another example is the Bradley-Terry-Luce model (Bradley and Terry (1952); Luce (1959)) which assumes each F_i follows a Gumbel distribution so that $F_i(y) = 1 - \exp(-\exp(y))$. This gives equations (2.4.1) and (2.4.2) a closed form, and we note that several extensions to this model have been proposed in the literature.

The inference to estimate each μ_i is difficult to do using maximum likelihood estimation (MLE) methods because it requires high-dimensional numerical integration to perform parameter estimation if we have a large number of items; early attempts with the Thurstone model modelled at most four items (Yu et al., 2019). Instead, some alternative methods using MCMC have been proposed for Thurstone models (Stern, 1990; Yao and Böckenholt, 1999). See Yu et al. (2019) for more details.

2.4.2 Paired comparison models: Smith models

Alternatively, a probability model for rankings can be deduced from a model for paired comparisons. This was first described by Smith (1950), where it is assumed that the $\binom{n}{2} = \frac{n(n-1)}{2}$ possible pairwise comparisons for n items are independent and are consistent with the transitivity property. For each pair of items $A_i, A_j \in \mathcal{A}$ $(i \neq j)$, let $p_{ij} = p(A_i \prec A_j)$ denote the probability that item A_i is preferred over item A_j. Then, the probability of observing a ranking $\mathbf{R}$ under the Smith model is proportional to the product of each p_{ij} required to generate the ranking, i.e.,

$$p(\mathbf{R}|\mathbf{p}) = C(\mathbf{p})^{-1} \prod_{\mathbf{R}(A_i)<\mathbf{R}(A_j)} p_{ij}, \quad \mathbf{R} \in \mathcal{P}_n, \tag{2.4.3}$$

where $C(\mathbf{p})$ is the normalisation constant determined by $\sum_{\mathbf{r}\in\mathcal{P}_n} \prod_{\mathbf{r}(A_i)<\mathbf{r}(A_j)} p_{ij}$.

The number of parameters in the Smith model is $\binom{n}{2}$, making it difficult to work with if we have a large number of items. This meant that some subclasses of the Smith model, proposed by Mallows (1957), were introduced to place constraints on the number of parameters to make it analytically tractable. We summarise some of these subclasses.

Mallows-Bradley-Terry Model

In the Mallows-Bradley-Terry (MBT) model the pairwise comparison probabilities are assumed to follow a Bradley-Terry (BT) model (Bradley and Terry, 1952). The BT model, which was first proposed by Zermelo (1929), assumes that the probability that an item A_i is preferred over another item A_j is

$$p_{ij} = p(A_i \prec A_j) = \frac{\lambda_i}{\lambda_i + \lambda_j}, \tag{2.4.4}$$

where each $\lambda_k > 0,\ k = 1, \dots, n$, such that $\sum_{k=1}^{n} \lambda_k = 1$, represents the item's score parameter. A higher value of λ_k indicates a stronger preference for item A_k; the MBT model depends only on the n score parameters of the items. The probability of observing a ranking under the MBT model is

$$p(\mathbf{R}|\boldsymbol{\lambda}) = C(\boldsymbol{\lambda}) \prod_{i=1}^{n-1} \lambda_i^{n-R_i}, \quad \mathbf{R} \in \mathcal{P}_n$$

where $C(\boldsymbol{\lambda})$ is the normalisation constant.

The score parameters are estimated using iterative optimisation methods such as the majorised-minimisation algorithm (Hunter, 2004) which uses a surrogate function to find the MLE. Recently, MCMC has been incorporated to evaluate the expectation of the log-likelihood in the majorised-minimisation algorithm (Sawadogo et al., 2019).

Mallows ϕ and σ models

The second subclass of Smith models requires only two parameters. Mallows (1957) assumes that any two rankings with the same distance from an assumed consensus ranking $\boldsymbol{\rho}$, which is the ranking with the highest probability of occurring, should have the same probability of being observed. An additional assumption is that the pairwise comparison probabilities are now dependent on their position in a ranking $\mathbf{R}$. Mallows (1957) suggests two distances; the Kendall distance which counts the number of pairwise disagreements between $\mathbf{R}$ and $\boldsymbol{\rho}$, giving $d_K(\mathbf{R}, \boldsymbol{\rho}) = \sum_{i<j} \mathbb{1}\{(R_i - R_j)(\rho_i - \rho_j)| < 0\}$, $1 \leq i < j \leq n$, and the Spearman distance, known as the l_2 distance, which measures the sum of squared displacements in rank between $\mathbf{R}$ and $\boldsymbol{\rho}$, giving $d_S(\mathbf{R}, \boldsymbol{\rho}) = \sum_{i=1}^{n}(R_i - \rho_i)^2$. It is shown in Mallows (1957) that the probability of a ranking under the resulting model is

$$p(\mathbf{R}|\boldsymbol{\rho}, \sigma, \phi) = C(\sigma, \phi)\sigma^{d_S(\mathbf{R},\boldsymbol{\rho})}\phi^{d_K(\mathbf{R},\boldsymbol{\rho})}, \quad \mathbf{R} \in \mathcal{P}_n,$$

where $\sigma, \phi > 0$ and $C(\sigma, \phi)$ is the normalisation constant. If $\phi = 1$, then we have the Mallows σ model,

$$p(\mathbf{R}|\boldsymbol{\rho}, \sigma) = C(\sigma, 1)\sigma^{d_S(\mathbf{R},\boldsymbol{\rho})}, \quad \mathbf{R} \in \mathcal{P}_n,$$

which is the Mallows model with the Spearman distance where each p_{ij} is dependent on whether $(R_i - R_j) > 0$. If $\sigma = 1$, we have the Mallows ϕ-model,

$$p(\mathbf{R}|\boldsymbol{\rho}, \phi) = C(1, \phi)\phi^{d_K(\mathbf{R}, \boldsymbol{\rho})}, \quad \mathbf{R} \in \mathcal{P}_n,$$

which is the Mallows model with the Kendall distance metric where p_{ij} is dependent on $|R_i - R_j|$. This subclass of models belongs to the general class of distance-based models which we will discuss further in Section 2.4.3.

2.4.3 Distance-based models: Mallows model

The subclass of paired comparison models containing the Mallows ϕ and σ models (Mallows, 1957) is generalised by Diaconis (1988) to give distance-based models. This class creates a probabilistic model for ranking data over the permutation space $\mathcal{P}_n$ rather than relying on scores for items or orders.

The Mallows model is parameterised by $\boldsymbol{\rho} \in \mathcal{P}_n$, known as the consensus ranking, which is the permutation with the highest probability that an individual may rank the set of items as $\mathbf{R} = \boldsymbol{\rho}$. It is also parameterised by $\alpha > 0$, the scale parameter, which controls the variability of the permutations around the consensus ranking. The Mallows model assumes that the probability of an observed ranking decays as the distance between the predicted ranking, $\mathbf{R}$, and the consensus ranking increases. The probability of observing a ranking $\mathbf{R}$ is defined as

$$p(\mathbf{R}) = p(\mathbf{R}|\boldsymbol{\rho}, \alpha) = \frac{1}{Z_n(\boldsymbol{\rho}, \alpha)} \exp\Big\{ -\frac{\alpha}{n} d(\mathbf{R}, \boldsymbol{\rho}) \Big\}, \quad \mathbf{R} \in \mathcal{P}_n, \tag{2.4.5}$$

where the partition function is defined as

$$Z_n(\boldsymbol{\rho}, \alpha) = \sum_{\mathbf{r} \in \mathcal{P}_n} \exp\Big\{ - \frac{\alpha}{n}\, d(\mathbf{r}, \boldsymbol{\rho}) \Big\}. \tag{2.4.6}$$

The distance function, $d(\cdot, \cdot) : \mathcal{P}_n \times \mathcal{P}_n \to [0, \infty)$, measures the "closeness" of a ranking to the consensus ranking and is a key feature of the Mallows model. They satisfy typical axioms:

$$d(\boldsymbol{\rho}, \boldsymbol{\rho}) = 0 \quad \forall \boldsymbol{\rho} \in \mathcal{P}_n \quad \text{(reflexivity);}$$

$$d(\boldsymbol{\rho}, \mathbf{R}) > 0 \quad \forall \boldsymbol{\rho}, \mathbf{R} \in \mathcal{P}_n \text{ such that } \boldsymbol{\rho} \neq \mathbf{R} \quad \text{(positivity);}$$

$$d(\boldsymbol{\rho}, \mathbf{R}) = d(\mathbf{R}, \boldsymbol{\rho}) \quad \forall \boldsymbol{\rho}, \mathbf{R} \in \mathcal{P}_n \quad \text{(symmetry).}$$

The Mallows literature discusses the use of right-invariant distances, where for every permutation $\boldsymbol{\sigma}$,

$$d(\mathbf{R}, \boldsymbol{\rho}) = d(\mathbf{R}\boldsymbol{\sigma}, \boldsymbol{\rho}\boldsymbol{\sigma}).$$

If we set $\boldsymbol{\sigma} = \boldsymbol{\rho}^{-1}$, then $d(\mathbf{R}, \boldsymbol{\rho}) = d(\mathbf{R}\boldsymbol{\rho}^{-1}, \boldsymbol{\rho}\boldsymbol{\rho}^{-1}) = d(\mathbf{R}\boldsymbol{\rho}^{-1}, \mathbf{1}_n)$, where $\mathbf{1}_n = \{1, 2, \ldots, n\}$ is the identity permutation. This means that the relabelling of items between two rankings is unaffected (Diaconis, 1988). This also means that the partition function is, in fact, independent of the consensus ranking; no matter which ranking we choose to set as $\boldsymbol{\rho}$, the sum of the distances between $\boldsymbol{\rho}$ and each of the $n!$ ranks in $\mathcal{P}_n$ will remain the same. Therefore, we can redefine the partition function in (2.4.6) as

$$Z_n(\boldsymbol{\rho}, \alpha) = Z_n(\alpha) = \sum_{\mathbf{r} \in \mathcal{P}_n} \exp\Big\{ -\frac{\alpha}{n} d(\mathbf{r}, \mathbf{1}_n) \Big\}.$$

Various right-invariant distances have been described in Diaconis (1988) and Marden (1995), these are:

- Cayley distance: Defined as $d_C(\mathbf{R}, \boldsymbol{\rho})$, it counts the minimum number of transpositions to transform $\mathbf{R}$ to $\boldsymbol{\rho}$.

- Kendall distance: This measures the number of adjacent transpositions that convert $\mathbf{R}$ into $\boldsymbol{\rho}$. Equivalently, this involves counting the number of pairwise disagreements between $\mathbf{R}$ and $\boldsymbol{\rho}$, giving $d_K(\mathbf{R}, \boldsymbol{\rho}) = \sum_{i<j} \mathbb{1}\big\{(R_i - R_j)(\rho_i - \rho_j)| < 0\big\},\ 1 \le i < j \le n.$

- Footrule (l_1): This measures the sum of the displacements in rank between $\mathbf{R}$ and $\boldsymbol{\rho}$, giving $d_F(\mathbf{R}, \boldsymbol{\rho}) = \sum_{i=1}^{n} |R_i - \rho_i|$.

- Spearman distance (l_2): This measures the sum of squared displacements in rank between $\mathbf{R}$ and $\boldsymbol{\rho}$, giving $d_S(\mathbf{R}, \boldsymbol{\rho}) = \sum_{i=1}^{n} (R_i - \rho_i)^2$.

- Hamming distance: This counts the number of items, which are not placed in the same position, between $\mathbf{R}$ and $\boldsymbol{\rho}$, giving $d_H(\mathbf{R}, \boldsymbol{\rho}) = n - \sum_{i=1}^{n} \mathbb{1}_{\rho_i}(R_i)$.

- Ulam distance: Defined as $d_U(\mathbf{R}, \boldsymbol{\rho})$, it counts the length of the complement of the longest common subsequence in $\mathbf{R}$ and $\boldsymbol{\rho}$. Equivalently, this is the number of deletion-insertion operations to convert $\mathbf{R}$ into $\boldsymbol{\rho}$.

Some of the distances, such as footrule and Kendall, satisfy the triangle inequality, that is

$$d(\mathbf{R}, \boldsymbol{\rho}) \leq d(\boldsymbol{\rho}, \boldsymbol{\sigma}) + d(\boldsymbol{\sigma}, \mathbf{R}) \quad \forall \boldsymbol{\rho}, \mathbf{R}, \boldsymbol{\sigma} \in \mathcal{P}_n,$$

meaning that they are in fact distance metrics. However, this property is not necessary for a distance to be right-invariant. We also note that the Mallows ϕ and σ models in Section 2.4.2 are the Mallows models with Kendall and Spearman distances, respectively, by substituting $\alpha = -n \log \phi$ and $\alpha = -n \log \sigma$ in (2.4.5).

Some distances have previously been used for specific applications. For example, the Cayley distance has been applied to computer vision and cryptography (Ziegler et al., 2012), the Hamming distance has been studied in cryptography to measure the distance between binary sequences and the Ulam distance has applied in DNA research to measure the distance between molecule strings (Critchlow et al., 1991). The Mallows model with the Kendall distance has been well-studied for two reasons discussed in Busse et al. (2007). First, the distance is intuitive because rankings can be inferred from pairwise disagreements; Mallows (1957) argues that the definition of the Kendall distance reflects how a person ranks a set of items. Second, it can also be computed efficiently; a closed form of the partition function is given in Fligner and Verducci (1986). Currently, there is no guidance on which distance function to use with the Mallows model given the application (Liu et al., 2019a).

Approximating the partition function

The choice of distance gives the Mallows model some flexibility but affects the tractability of computing the partition function for large n. We highlight that the partition function can be analytically computed for certain right-invariant distances. For example, the Kendall distance (Fligner and Verducci, 1986; Lu and Boutilier, 2014), the Cayley distance (Irurozki et al., 2018) and the Hamming distance (Irurozki et al., 2014a) have been shown to have a closed form. However, the right-invariant distances, the footrule and the Spearman distances make the computation of the partition function NP-hard. So far, the partition function with these distances has been numerically solved for very small values of n, but it is infeasible for larger n. In particular, an exact approximation can be found for $n \leq 50$ items for the footrule and $n \leq 14$ for the Spearman distances respectively (Vitelli et al., 2018). Recently, Vitelli et al. (2018) suggested an importance sampling scheme to approximate $Z_n(\alpha)$ with some arbitrary precision for any right-invariant distance. Since it does not depend on $\boldsymbol{\rho}$, $Z_n(\alpha)$ can be computed offline over a grid for α, given n. An iterative algorithm which gives asymptotic approximation for $Z_n(\alpha)$ as $n \to \infty$ is proposed in Mukherjee (2016). Further details on the Mallows model will be discussed in Section 2.5.

2.4.4 Multistage models

In a multistage model, a ranking is created using an iterative process. This is performed by a sequence of independent stages in which the next preferred item is selected among the remaining items in $\mathcal{A}$ each time.

Plackett-Luce

The Plackett-Luce (PL) model (Luce (1959), Plackett (1975)) is a common example of a multistage model. Given a set of score parameters for n items, $\lambda_k,\ k = 1, \ldots, n$, an ordering is constructed by the following process. In the first iteration, the item with the highest ranking is selected with probability $\frac{\lambda_k}{\sum_{i=1}^{n} \lambda_i}$, then in the remaining iterations, the next item with the highest ranking is chosen from the collection of items remaining to be selected in $\mathcal{A}$ with normalised selection probability. The process occurs over $n-1$ iterations to create an ordering with probability

$$p(\mathbf{X}|\boldsymbol{\lambda}) = \prod_{i=1}^{n-1} \frac{\lambda_i}{\sum_{k=i}^{n} \lambda_k}, \quad \mathbf{X} \in \mathcal{X}_n.$$

Alternatively, the probability of observing a full ranking under the Plackett-Luce model is

$$p(\mathbf{R}|\boldsymbol{\lambda}) = \prod_{i=1}^{n-1} \frac{\lambda_{R^{-1}(i)}}{\sum_{k=i}^{n} \lambda_{R^{-1}(k)}}, \quad \mathbf{R} \in \mathcal{P}_n.$$

The PL model has been well-studied. To perform parameter inference, we can use MLE using the majorise-minimisation algorithm (Hunter, 2004). Bayesian approaches have also been considered, as in Guiver and Snelson (2009) and Caron and Doucet (2012). In Guiver and Snelson (2009), the power expectation propagation algorithm (EP) (Minka, 2004) is used instead of the expectation maximisation algorithm (EM) Dempster et al. (1977) and apply the method to large data sets containing partial rankings. In Caron and Doucet (2012), a data augmentation scheme is proposed using conjugate priors (gamma distribution) for the likelihood to create a Gibbs sampler for the PL parameters.

Generalised Mallows model

The Mallows model (Mallows, 1957) is criticised for having only one parameter, α, to handle the spread of the distribution of the ranking data since every possible permutation at the same distance from the consensus ranking has the same probability value (Yu et al., 2019). One way to avoid this is to incorporate scale parameters for each position of the ranking to indicate uncertainty for these positions. A popular extension is the generalised Mallows model (GMM) (Fligner and Verducci, 1986), which considers $n-1$ scale parameters $(\alpha_1, \alpha_2, \ldots, \alpha_{n-1})$, each of which affects the certainty of some positions of the permutation. Certain distances cannot be used with the model; the footrule and Spearman distances cannot be decomposed into $n-1$ terms.

MLE for the GMM has been shown for Kendall (Mandhani and Meila, 2009), Cayley (Irurozki et al., 2018), and Hamming distances (Irurozki et al., 2014a). This collection of work has been implemented into the `PerMallows` R package (Irurozki et al., 2016) which samples and learns the GMM with the Kendall, Cayley and Hamming distances using several algorithms. Meilă and Bao (2010) considered a Bayesian framework with Kendall distance with top-k rankings for an infinite number of items. This was extended by Meilă and Chen (2010) to handle clustering using Dirichlet processes.

2.5 Inference with the Mallows model

We have seen that several attempts have been made to produce a model that best summarises preference data in a meaningful way. These models can be grouped into several classes (Fligner and Verducci, 1986), one of which was the distance-based model in Section 2.4.3. In this class, we briefly introduced the Mallows model and discussed one extension of the model, the Generalised Mallows Model, in Section 2.4.4. The focus of this thesis is to apply sequential inference to the Mallows Model, so here we provide further details of the literature surrounding the Mallows model and its extensions. We will discuss the developments of the Mallows model when handling different kinds of preference data, including full rankings, partial rankings, pairwise comparisons and heterogeneous data.

2.5.1 Frequentist approaches

When discussing the Mallows model, we shall use the following definition

$$p(\mathbf{R}) = p(\mathbf{R}|\boldsymbol{\rho}, \alpha) = \frac{1}{Z_n(\boldsymbol{\rho}, \alpha)} \exp\left\{ -\frac{\alpha}{n} d(\mathbf{R}, \boldsymbol{\rho}) \right\}, \quad \mathbf{R} \in \mathcal{P}_n.$$

When estimating the model parameters given a set of M observed rankings, $\{\mathbf{R}_1, \dots, \mathbf{R}_M\}$, we find the maximum likelihood estimate for $\boldsymbol{\rho}$ (for a given α),

$$\arg\min_{\boldsymbol{\rho} \in \mathcal{P}_n} \sum_{j=1}^{M} d(\mathbf{R}_j, \boldsymbol{\rho}),$$

is not straightforward to solve (Liu et al., 2019a) because the space of permutations $\mathcal{P}_n$ has $n!$ possible rankings, which for large n makes the problem of minimising the sum of distances NP-hard (Bartholdi et al., 1989).

Inference from pairwise comparisons

An online learning setting has been considered to learn a single Mallows model with pairwise preferences. Busa-Fekete et al. (2014) devise algorithms to find the most probable top-ranked item, probable full ranking or estimate the Mallows model with Kendall distance only. In each iteration t, the learner is allowed to gather information by asking for a single pairwise comparison between two items $(A_i^t, A_j^t) \in \mathcal{A}$. The learner receives stochastic feedback on the outcome of each pairwise comparison: 1 if $A_i^t \prec A_j^t$ and 0 if $A_i^t \succ A_j^t$ with probability p_{ij} and $1 - p_{ij}$, respectively. Given the result, the learner updates their estimations and decides whether to continue learning or terminate with their prediction. The goal is to minimise the number of iterations to learn whilst guaranteeing a certain level of confidence in the prediction. This is one example of applying the Mallows model to an online setting; however, the underlying assumption of the model is that the complete information of every pairwise comparison is available which is not always possible.

Inference from partial rankings

Most applications of the Mallows model with partial rankings have typically been used in the case of top-k rankings and with the Kendall distance. Busse et al. (2007) use a maximum entropy approach to perform data augmentation for missing entries in the top-k rankings. This is where missing ranks in each observed partial ranking are allocated to positions that have the highest entropy. The entropy values are determined based on how often each item has been ranked in every position in the observed data set. This approach removes any assumptions about how each item is ranked.

Fligner and Verducci (1986) define a distribution on top-k rankings by considering the marginals of all possible completions of the top-k rankings. This is extended by Lebanon and Mao (2008) to general partial rankings and the distribution is estimated by proposing a nonparametric kernel density estimator. However, the authors assume that an underlying full ranking exists. Instead, Chierichetti et al. (2018) assume the existence of a top-k list and construct a Mallows model for top-k lists that uses a Kendall distance measure to measure between top-k lists. They considered two learning problems: learning the consensus top-k list for a given top-k Mallows model and learning a mixture of top-k Mallows models.

Clustering

The Mallows model has also been studied as a mixture model, where we assume that a collection of ranking data was produced by a collection of several homogeneous populations and each population can be modelled as an independent Mallows model. If we have C components, then the ranking model is

$$p(\mathbf{R}|\boldsymbol{\rho}_{1:C}, \alpha_{1:C}, \tau_{1:C}) = \sum_{c=1}^{C} \frac{\tau_c}{Z_n(\alpha_c)} \exp\Big\{ -\frac{\alpha_c}{n} d(\mathbf{R}, \boldsymbol{\rho}_c)\Big\}, \quad \mathbf{R} \in \mathcal{P}_n,$$

where τ_c is the probability that an observation $\mathbf{R}$ belongs to a cluster $c \in \{1, \dots, C\}$. The task is to determine the number of components, if this is unknown, as well as the model parameters $\boldsymbol{\rho}_{1:C}, \alpha_{1:C}$ and $\tau_{1:C}$.

To learn from a mixture model, a common method is to use the expectation maximisation algorithm (EM) (Dempster et al., 1977) to find the maximum likelihood for each parameter for different values of C and select the number of components that maximise the log-likelihood (Busse et al., 2007). This approach has been studied by Murphy and Martin (2003); Busse et al. (2007); Lu and Boutilier (2014); Stoyanovich et al. (2016) with the Mallows model.

Murphy and Martin (2003) studied the Mallows mixture model with Kendall, Cayley and footrule distances using full rankings. They use two different criteria to choose the appropriate model for the heterogeneous data set. These are the Bayesian information criterion (BIC) (Fraley and Raftery, 1998), defined as

$$\mathrm{BIC}(C) = 2\ell(\hat{\theta}_C) = v(C)\log(M),$$

where $\ell(\hat{\theta}_C)$ and $v(C)$ are the maximum log-likelihood and the number of free parameters, respectively, and the integrated complete likelihood (ICL) (Biernacki et al., 2000)

$$\mathrm{ICL}(C) = \mathrm{BIC}(C) + 2\sum_{j=1}^{M}\sum_{c=1}^{C} \hat{z}_c^{(j)} \log \hat{z}_c^{(j)}$$

where $\hat{z}_c^{(j)}$ are the latent membership variables, defined below. The model that maximises either criterion is selected.

To learn the Mallows mixture in Murphy and Martin (2003), each ranking is allocated latent indicator variables $z = (z_1, \dots, z_c)$, defined such that $z_c = 1$ means that observation belongs to component $c \in \{1, \dots, C\}$. So, for M rankings, we have the membership variables $\mathbf{z}_M = (z^{(1)}, \dots, z^{(M)})$. The complete-data log-likelihood for (2.5.1) can be expressed as

$$\ell(\boldsymbol{\rho}_{1:C}, \alpha_{1:C}, \tau_{1:C} | \mathbf{R}_{1:M}, \mathbf{z}_M) = \sum_{j=1}^{M} \sum_{c=1}^{C} z_c^{(j)} \Big[\log \tau_c - \log Z_n(\alpha_c) - \frac{\alpha_c}{n} d(\mathbf{R}_j, \boldsymbol{\rho}_c) \Big].$$

In the E-step, the expected allocation variables are calculated; then in the M-step we update the maxima of $\boldsymbol{\rho}_{1:C}$, $\alpha_{1:C}$ and $\tau_{1:C}$. The EM algorithm is easy to implement but can get stuck in local maxima, so Murphy and Martin (2003) recommend running the algorithm from several starting points. Busse et al. (2007) extend this method to handle partial rankings for the Mallows model with the Kendall distance and use the same methods, as described in Murphy and Martin (2003) to select the number of components. In both Murphy and Martin (2003) and Busse et al. (2007), the methods are applied to the American Psychological Association data set (Diaconis, 1988). The data set contains rankings for only five items, so the discussion on the difficulty in estimating the partition function, particularly for footrule distance, is avoided.

Lu and Boutilier (2014) learn Mallows mixtures using their proposed generalised repeated insertion model (GRIM) algorithm, an extension of the repeated insertion model (RIM) (Doignon et al., 2004) for sampling for the Mallows model, which can sample from the Mallows model with Kendall distance from a set of pairwise preferences. The algorithm is used in the EM algorithm as part of a Gibbs sampler to generate a tractable approximate posterior to find the local maximum of the model parameters. The authors perform preference learning and prediction with the MovieLens (GroupLens) and Sushi (Kamishima, 2003) data sets. However, no uncertainty estimates were provided. Stoyanovich et al. (2016) extend the work of Lu and Boutilier (2014). Instead of trying to fit a mixture model for a varying number of components before selecting the optimum, the authors use affinity propagation (Frey and Dueck, 2007), a message-passing clustering method, to identify an appropriate number of Mallows models in a mixture. Affinity propagation finds assessors or "exemplars" that are representative of clusters and iterates between assessors nominating another assessor as their exemplar based on their similarity, which is the number of pairs of matching comparisons that the two assessors share, and determining how appropriate the assessor's nomination for exemplar is given all the other nominations until convergence.

Theoretical work that relates to clustering with the Mallows model includes Awasthi et al. (2014), which provides theoretical identifiability of the model parameters of a two-component mixture model. They create a polynomial-time algorithm which uses tensor decomposition techniques to determine the component parameters. However, identifying the consensus rankings of each component is poor when they are close together; this is a typical problem in clustering. Chierichetti et al. (2015) show identifiability when the scale parameters are known and the same for all components. Liu and Moitra (2018) show that we can learn the parameters of the mixture model with a polynomial-time algorithm for any fixed number of components.

R packages

Finally, several R packages have been developed that use the Mallows models and the GMM (Mollica and Tardella, 2020). The `RMallows` R package (Gregory, 2012) uses the EM algorithm to fit a Mallows model given a set of full or partial rankings, with or without ties. The `pmr` R package (Lee and Yu, 2013) implements MLE functions for the Mallows model and the GMM under the Kendall, Spearman and footrule distances with full rankings. The package also provides descriptive statistics and visualisations of the preference data. The `PerMallows` R package (Irurozki et al., 2016) samples and learns the parameters of the Mallows model and the GMM with the Kendall, Cayley (Irurozki et al., 2018), Hamming (Irurozki et al., 2014a) and Ulam (Irurozki et al., 2014b) distances. It also provides distance-related functions and permutation-based operations. However, each algorithm cannot be applied to every distance; the package provides three different sampling algorithms (two exact, one approximate) and two learning algorithms (one exact, one approximate). Finally, `rankdist` (Qian and Philip, 2019) fits different distance-based ranking models to develop and test, including the Mallows mixture model, with the Kendall, Spearman, footrule and Hamming distances, given a set of full or partial rankings.

2.5.2 The Bayesian Mallows model

Several methodological developments have been made to the Mallows model to perform inference. However, there are some limitations to using the model. First, there is a lack of flexibility in existing methods to handle different kinds of preference data including full rankings, partial rankings, pairwise comparisons and heterogeneous data. The partition function can be found exactly or estimated for some of the right-invariant distances, but we notice that there appears to be no general approach that can use every distance listed in Section 2.4.3; most papers only consider the Kendall distance. Finally, the proposed algorithms discussed in the papers are tested with synthetic and real-world data sets, but the size of these data sets is very small in terms of the number of items.

Vitelli et al. (2018) provide a Bayesian hierarchical model for inference, which overcomes the main criticisms given for earlier attempts with the Mallows model. First, the Bayesian Mallows model can handle a variety of ranking data, such as a heterogeneous set of rankings, partial rankings, pairwise comparisons, and even implicit data. The model can account for any right-invariant distance when sampling, learning the parameters of the Mallows model and estimating the partition function, particularly for the footrule and Spearman distances which had not been previously tackled in the literature surrounding the Mallows model. Furthermore, the hierarchical Bayesian model can handle large numbers of assessors and items and this is demonstrated with a variety of examples with different preference data.

A key component of Bayesian models is the use of prior distributions to incorporate subjective beliefs about the parameters of interest. Given the prior distributions $\pi(\boldsymbol{\rho})$ and $\pi(\alpha)$ and assuming prior independence of these variables, then the Bayesian

Mallows model has the posterior density

$$\pi(\boldsymbol{\rho}, \alpha | \mathbf{R}_1, \ldots, \mathbf{R}_M) \propto \frac{\pi(\boldsymbol{\rho})\pi(\alpha)}{[Z_n(\alpha)]^M} \exp\left\{ -\frac{\alpha}{n} \sum_{j=1}^{M} d(\mathbf{R}_j, \boldsymbol{\rho}) \right\}. \tag{2.5.1}$$

In particular, Vitelli et al. (2018) specify a uniform prior for the consensus ranking, $\pi(\boldsymbol{\rho}) = (n!)^{-1}\mathbb{1}_{\mathcal{P}_n}(\boldsymbol{\rho})$ in the space $\mathcal{P}_n$, recalling that $\mathcal{P}_n$ is the permutation space of n items of size $n!$, and an exponential prior for α, with density $\pi(\alpha|\lambda) = \lambda \exp\{-\lambda\alpha\}\mathbb{1}_{[0,\infty)}(\alpha)$. The advantage of the Bayesian Mallows model is that we can compute posterior summaries of interest, for example, the marginal posterior distribution of $\boldsymbol{\rho}$,

$$p(\boldsymbol{\rho}|\mathbf{R}_1, \ldots, \mathbf{R}_M) \propto \pi(\boldsymbol{\rho}) \int_0^\infty \frac{\pi(\alpha)}{[Z_n(\alpha)]^M} \exp\left\{ -\frac{\alpha}{n} \sum_{j=1}^{M} d(\mathbf{R}_j, \boldsymbol{\rho}) \right\} d\alpha,$$

as well as produce any uncertainty estimates.

2.5.3 Markov chain Monte Carlo for the Bayesian Mallows model

The Metropolis-Hastings algorithm is used in Vitelli et al. (2018) to sample from (4.2.1), assuming, for now, that we have a set of M full rankings. The Markov chain Monte Carlo (MCMC) scheme begins with initial values $\boldsymbol{\rho}_0 \in \mathcal{P}_n$ and $\alpha_0 \geq 0$ and comprises two steps. In one iteration of the algorithm, a new consensus ranking $\boldsymbol{\rho}'$ is proposed to update $\boldsymbol{\rho}$ according to a distribution which is centred around the current ranking $\boldsymbol{\rho}$. The proposal step for $\boldsymbol{\rho}$ is computed using the leap-and-shift proposal algorithm of Vitelli et al. (2018). Similarly, a new value α' is proposed from the log-normal distribution to update the current value of α.

Sampling $\boldsymbol{\rho}$ using the leap-and-shift proposal

We first summarise how the proposal distribution for the consensus ranking is carried out. The leap-and-shift algorithm begins by fixing a value $L \in \{1, \ldots, \lfloor \frac{n-1}{2} \rfloor\}$. This is the leap size which determines the maximum number of positions that an item can increase or decrease by in rank. Next, a position $u \in \mathcal{U} = \{1, 2, \ldots, n\}$ is selected in the current ranking $\boldsymbol{\rho}$ with probability $\frac{1}{n}$. This is the item in $\boldsymbol{\rho}$ whose current rank $\boldsymbol{\rho}_u$ will change. Then, a support set $\mathcal{S}$ is defined which contains all possible ranks that $\boldsymbol{\rho}_u$ can leap to by at most $\pm L$ in rank. A value r is drawn randomly from the set $\mathcal{S}$ which is the rank that $\boldsymbol{\rho}_u$ will leap to. At the end of the leap step, we obtain an intermediate consensus ranking $\boldsymbol{\rho^*}$ which will contain elements $\rho_u^* = r$ and $\rho_i^* = \rho_i$ for $i \in \{1, \ldots, n\} \backslash \{u\}$. Thus $\boldsymbol{\rho}^*$ will contain two elements with value r and the element $\boldsymbol{\rho}_u$ will be missing. Finally, during the shift step, the ranks of the items are adjusted to create a valid consensus ranking $\boldsymbol{\rho'}$. The procedure for the leap-and-shift algorithm is given in Algorithm 1.

We demonstrate the leap-and-shift procedure with an example. Assuming we have $n = 5$ items, the current consensus ranking is $\boldsymbol{\rho} = (2, 4, 1, 5, 3)$, and the fixed leap size is $L = 2$. Then, using leap-and-shift could result in this potential outcome:

1. We draw $u = 4$ from the set $\{1, \ldots, 5\}$.

2. Using $\rho_u = 5$, the set $\mathcal{S}$ contains the elements $\{3, 4\}$.

3. We randomly draw $r \in \mathcal{S}$, with the result $r = 3$.

4. We define $\boldsymbol{\rho}^* = (2, 4, 1, 3, 3)$ as $\rho_u = 5$ when $u = 4$.

5. Since $\boldsymbol{\rho}^*_u = 3$ is fixed, the determined shifts we have to make sure that $\boldsymbol{\rho}'$ contains the elements $\{1, \dots, 5\}$ result in $\boldsymbol{\rho}' = (2,\ 5,\ 1,\ 3,\ 4)$.

Algorithm 1: Leap-and-shift (Vitelli et al., 2018)

```
Input: $\boldsymbol{\rho}$, $n$.
Output: Proposal consensus ranking $\boldsymbol{\rho}'$.
Fix $L \in \{1, \dots, \lfloor (n-1)/2 \rfloor\}$ .
Draw $u \sim \mathcal{U}\{1, \dots, n\}$.
Define a set of integers
  $\mathcal{S} = \{\max(1,\ \rho_u - L),\ \min(n,\ \rho_u + L)\} \backslash \{\rho_u\},\ \mathcal{S} \subseteq \{1, \dots, n\}$.
Draw $r$ from $\mathcal{S}$.
Let $\boldsymbol{\rho}^* \in \{1, \dots, n\}^n$ contain elements $\rho^*_u = r$ and $\rho^*_i = \rho_i$ for $i \in \{1, \dots, n\} \backslash \{u\}$,
  constituting the leap step.
Set $\Delta = \rho^*_u - \rho_u$ and the proposed $\boldsymbol{\rho}' \in \mathcal{P}_n$ with elements
```

$$\rho'_i = \begin{cases} \rho^*_u & \text{if } \rho_i = \rho_u \\ \rho_i - 1 & \text{if } \rho_u < \rho_i \leq \rho^*_u \text{ and } \Delta > 0 \\ \rho_i + 1 & \text{if } \rho_u > \rho_i \geq \rho^*_u \text{ and } \Delta < 0 \\ \rho_i & \text{else} \end{cases},$$

for $i = 1, \dots, n$, constituting the shift step.

The probability mass function associated with the transition from current $\boldsymbol{\rho}$ to $\boldsymbol{\rho}'$

is calculated in Vitelli et al. (2018) as

$$\begin{aligned}
p_L(\boldsymbol{\rho'}|\boldsymbol{\rho}) &= \sum_{u=1}^{n} p_L(\boldsymbol{\rho'}|U=u,\ \boldsymbol{\rho})P(U=u) \\
&= \frac{1}{n}\sum_{u=1}^{n}\Bigg\{ \mathbb{1}_{\boldsymbol{\rho_{-u}}}(\boldsymbol{\rho}^*_{-u}) \cdot \mathbb{1}_{\{0<|\boldsymbol{\rho}_u-\boldsymbol{\rho}^*_u|\le L\}}(\boldsymbol{\rho}^*_u) \\
&\cdot \left[\frac{\mathbb{1}_{\{L+1,\ldots n-l\}(\boldsymbol{\rho}_u)}}{2L} + \sum_{l=1}^{L}\frac{\mathbb{1}_{\{l\}}(\boldsymbol{\rho}_u)+\mathbb{1}_{\{n-l+1\}}(\boldsymbol{\rho}_u)}{L+l-1}\right]\Bigg\} \\
&+ \frac{1}{n}\sum_{u=1}^{n}\Bigg\{ \mathbb{1}_{\boldsymbol{\rho_{-u}}}(\boldsymbol{\rho}^*_{-u}) \cdot \mathbb{1}_{\{|\boldsymbol{\rho}_u-\boldsymbol{\rho}^*_u|=1\}}(\boldsymbol{\rho}^*_u) \\
&\cdot \left[\frac{\mathbb{1}_{\{L+1,\ldots n-l\}(\boldsymbol{\rho}^*_u)}}{2L} + \sum_{l=1}^{L}\frac{\mathbb{1}_{\{l\}}(\boldsymbol{\rho}^*_u)+\mathbb{1}_{\{n-l+1\}}(\boldsymbol{\rho}^*_u)}{L+l-1}\right]\Bigg\}.
\end{aligned} \tag{2.5.2}$$

Equation (2.5.2) simply accounts for all of the possible ways to transition from $\boldsymbol{\rho}$ to $\boldsymbol{\rho'}$. For any transition, (2.5.2) is the product of the probability of selecting $u \in \{1,\ldots,n\}$ and the probability of selecting a given $r \in \mathcal{S}$, $\frac{1}{n} \times \frac{1}{|\mathcal{S}|}$. The final term in (2.5.2) is the additional probability if the leap is of size 1 since there are two possible scenarios for the result $\boldsymbol{\rho'}$. A simpler way to interpret the transition probability is to consider the probability of moving to $\boldsymbol{\rho}^*$ from $\boldsymbol{\rho}$. Since the mapping from $\boldsymbol{\rho}^*$ to $\boldsymbol{\rho'}$ is deterministic and the candidate u is drawn randomly, the probability of moving to $\boldsymbol{\rho}^*$ given $\boldsymbol{\rho}$ is

$$p(\boldsymbol{\rho}^*|\boldsymbol{\rho}_u) = \begin{cases} \frac{1}{\boldsymbol{\rho}_u-1+L} & \text{if } \boldsymbol{\rho}_u \in \{1,\ldots,L\}; \\ \frac{1}{2L} & \text{if } \boldsymbol{\rho}_u \in \{L+1,\ldots,n-L\}; \\ \frac{1}{n-\boldsymbol{\rho}_u+L} & \text{if } \boldsymbol{\rho}_u \in \{n-L+1,\ldots,n\}. \end{cases}$$

The probability of accepting the new proposed value of $\boldsymbol{\rho}$ in the Metropolis-

Hastings MCMC algorithm is

$$\min\left\{1,\ \frac{p_L(\boldsymbol{\rho}|\boldsymbol{\rho'})\pi(\boldsymbol{\rho'})}{p_L(\boldsymbol{\rho'}|\boldsymbol{\rho})\pi(\boldsymbol{\rho})}\exp\left[-\frac{\alpha}{n}\sum_{j=1}^{N}(d(\mathbf{R}_j,\boldsymbol{\rho'})-d(\mathbf{R}_j,\boldsymbol{\rho}))\right]\right\}.$$

Sampling α using the log-normal proposal distribution

When updating the value of α, the algorithm samples a proposal α' from a log-normal distribution $\log\mathcal{N}(\alpha,\sigma_\alpha^2)$ and is accepted with probability

$$\min\left\{1,\ \frac{Z_n(\alpha)^M\pi(\alpha')\alpha'}{Z_n(\alpha')^M\pi(\alpha)\alpha}\exp\left[-\frac{(\alpha'-\alpha)}{n}\sum_{j=1}^{M}d(\mathbf{R}_j,\boldsymbol{\rho})\right]\right\}.$$

The parameter σ_a^2 can be tuned to obtain a desired acceptance probability. When sampling complete rankings, Vitelli et al. (2018) suggest using an integer-valued parameter, α_{jump}, in the MCMC algorithm so that an α' is proposed only every α_{jump} iterations. This is because in one MCMC iteration, $\boldsymbol{\rho}$ cannot explore the posterior distribution easily, whereas α can move around the parameter space much more. Again, this parameter can be tuned to ensure better mixing of the MCMC algorithm.

2.5.4 Inference with the Bayesian Mallows model

Inference is performed using the MCMC scheme discussed in Section 2.5.3, but several adaptations are made to incorporate a variety of preference data formats, particularly if the observed information from each user is incomplete and/or heterogeneous. The process of estimating the parameters of the Mallows model is then calculated with the posterior summaries. This has been implemented in the `BayesMallows` R package

(Sørensen et al., 2020) on CRAN. We summarise the adaptations for each type of preference data, but further details can be found in Vitelli et al. (2018).

Inference from partial rankings

Vitelli et al. (2018) consider top-k rankings, where either each individual ranks a subset $\mathcal{A}_j$ of $k < n$ items from 1 to k or where item ranks are missing at random meaning that each assessor has specified $n_j < n$ ranks for items but they have not necessarily given their top-k items.

Data augmentation assigns the remaining ranks to unranked items to give full augmented rankings $\tilde{\mathbf{R}}_1, \ldots, \tilde{\mathbf{R}}_M$, so the posterior is now defined as

$$p(\boldsymbol{\rho}, \alpha | \mathbf{R}_1, \ldots, \mathbf{R}_M) = \sum_{\tilde{\mathbf{R}}_1 \in \mathcal{S}_1} \cdots \sum_{\tilde{\mathbf{R}}_M \in \mathcal{S}_M} p(\boldsymbol{\rho}, \alpha, \tilde{\mathbf{R}}_1, \ldots, \tilde{\mathbf{R}}_M | \mathbf{R}_1, \ldots, \mathbf{R}_M),$$

where each $\mathcal{S}_j = \{\tilde{\mathbf{R}}_j \in \mathcal{P}_n : \tilde{\mathbf{R}}_{ij} = \mathbf{X}_j^{-1}(A_i) \text{ if } A_i \in \mathcal{A}_j,\ j = 1, \ldots, M\}$ is the set of possible augmented rankings that are compatible with the observed partial rankings. In the MCMC algorithm, the augmented rankings are initialised by assigning missing ranks to unranked items randomly with the allowable augmentations of the missing ranks. The algorithm alternates between sampling $\tilde{\mathbf{R}}_1, \ldots, \tilde{\mathbf{R}}_M$ given the current α and $\boldsymbol{\rho}$, then sampling α and $\boldsymbol{\rho}$ given the current $\tilde{\mathbf{R}}_1, \ldots, \tilde{\mathbf{R}}_M$. When sampling the candidate augmented ranks, Vitelli et al. (2018) suggest to use a uniform proposal distribution to sample each $\tilde{\mathbf{R}}'_j$ from $\mathcal{S}_j$, which is then accepted with probability

$$\min\left\{1,\ \exp\left[-\frac{\alpha}{n}\Big(d(\tilde{\mathbf{R}}'_j, \boldsymbol{\rho}) - d(\tilde{\mathbf{R}}_j, \boldsymbol{\rho})\Big)\right]\right\}.$$

Inference from pairwise comparisons

If each assessor can specify preferences between pairs of items in $\mathcal{A}$, then we can define $\mathcal{B}_j$ to be the set of pairwise comparisons that each assessor $j = 1, \ldots, M$ gives. The size of each pairwise preference set is $|\mathcal{B}_j| \leq \binom{n}{2}$. Then, we can define $\text{tc}(\mathcal{B}_j)$ as the transitive closure of all possible valid pairwise comparisons generated in $\mathcal{B}_j$; in other words, $\text{tc}(\mathcal{B}_j)$ contains the pairwise comparisons in $\mathcal{B}_j$ and any additional pairwise comparisons that involve elements in $\mathcal{A}$ that are compatible with in $\mathcal{B}_j$.

Since each of the items in $\mathcal{A}$ do not necessarily have a fixed rank, the MCMC algorithm needs to propose augmented complete rankings which will be compatible with constraints provided by $\text{tc}(\mathcal{B}_j)$ for each assessor. This is carried out in Vitelli et al. (2018) by using a modified leap-and-shift proposal step, which is now symmetric, to sample augmented ranks $\tilde{\mathbf{R}}_j$ for each assessor j that are compatible with $\text{tc}(\mathcal{B}_j)$. The MCMC algorithm alternates between sampling $\tilde{\mathbf{R}}_1, \ldots, \tilde{\mathbf{R}}_M$ given the current α, $\boldsymbol{\rho}$ and $\text{tc}(\mathcal{B}_j)$, then sampling α and $\boldsymbol{\rho}$ given the current augmented rankings $\tilde{\mathbf{R}}_1, \ldots, \tilde{\mathbf{R}}_M$.

Clustering

We consider the Bayesian fitting of the Mallows mixture model in Section 2.5.1, where we assume initially that the number of components C is known. Each ranking has a cluster assignment $\{z_1, \ldots, z_M\} \in \{1, \ldots, C\}$ to indicate the component that each assessor is assigned. Assuming conditional independence across clusters, the likelihood for observed full rankings $\mathbf{R}_1, \ldots, \mathbf{R}_M$ under the Bayesian Mallows model can be expressed as

$$p(\mathbf{R}_1, \ldots, \mathbf{R}_M | \{\boldsymbol{\rho}_c, \alpha_c\}_{c=1,\ldots,C}, z_1, \ldots, z_M) = \prod_{j=1}^{M} \frac{\mathbb{1}_{\mathcal{P}_m}(\mathbf{R}_j)}{Z_m(\alpha_{z_j})} \exp\Big\{ - \frac{\alpha_{z_j}}{m} d(\mathbf{R}_j, \boldsymbol{\rho}_{z_j}) \Big\},$$

The cluster assignments are distributed according to $P(z_1, \ldots, z_M | \tau_1, \ldots, \tau_C) = \prod_{j=1}^{N} \tau_{zj}$, where τ_c is the probability of assigning an assessor to component $c \in \{1, \ldots, C\}$. These assignment probabilities have a standard symmetric Dirichlet prior $\pi(\tau_1, \ldots, \tau_C) = \Gamma(\psi C)\Gamma(\psi)^{-C} \prod_{c=1}^{C} \tau_c^{\psi - 1}$, where $\Gamma(\cdot)$ is the gamma function. The MCMC algorithm alternates between sampling $\boldsymbol{\rho}_1, \ldots, \boldsymbol{\rho}_C$ and $\alpha_1, \ldots, \alpha_C$ in a Metropolis-Hastings step and $\tau_1, \ldots, \tau_C$ and $z_1, \ldots, z_M$ in a Gibbs sampler step.

If the number of components is unknown then we can run a thinned subset of the MCMC algorithm for clustering with different values of C and calculate the posterior distance between the rankings of each assessor and their corresponding component's consensus ranking. This is known as the within-sum cluster distance. The results are plotted in an elbow plot; the point at which an "elbow" occurs in the plot is considered to be the optimal number of components.

2.5.5 Extensions of the Bayesian Mallows model

Several interesting extensions have been proposed with the Bayesian Mallows model. We provide a summary of the existing developments.

Handling intransitive pairwise comparisons

Handling intransitive pairwise comparisons to perform inference is one example of a Bayesian Mallows model extension. To recall, a pairwise comparison is when an individual selects an item from a pair of distinct items that they prefer based on some attribute. It is assumed that each individual does not contradict themselves when specifying their preferences between pairs of items within a set, however, if this occurs, then the pairwise comparisons are intransitive. These can occur for several reasons: there may be too many items to consider for a user to be aware of an overall transitive ordering; several items are considered too similar by the user or the user preferences may have changed over time.

In Crispino et al. (2016), the result of intransitive pairwise comparisons means that we cannot obtain a latent ranking for each user. Here, $\mathcal{B}_j$ is defined as a set of pairwise preferences specified by a user j on a set of n items, $A_1, \dots, A_n$. It is assumed that each user has a latent ranking $\tilde{\mathbf{R}}_j \in \mathcal{P}_n$ that follows a Mallows model distribution. The likelihood for $\mathcal{B}_j$ in this scenario is

$$p(\mathcal{B}_j|\boldsymbol{\rho}, \alpha) = \sum_{\mathbf{R}_j \in \mathcal{P}_n} p(\mathcal{B}_j|\tilde{\mathbf{R}}_j = \mathbf{R}_j)\pi(\tilde{\mathbf{R}}_j = \mathbf{R}_j|\boldsymbol{\rho}, \alpha).$$

The authors incorporate an additional layer of latent variables into the hierarchical model to account for the mistakes that are assumed to be made by the assessor. They propose two models: the Bernoulli model to handle random mistakes, and the logistic model for mistakes from ordering similar items.

This particular model was used to carry out a case study on an experiment with acousmatic music (Crispino et al., 2019). Each assessor listened to pairs of different abstract sounds, and were asked to choose which sound was perceived as more human. Furthermore, each assessor only compared a subset of all possible pairs of sounds, creating some sparsity in the data set. The study was tasked to perform inference at an individual level and to cluster assessors.

Time-varying rankings

An alternative extension to the Bayesian Mallows model considers the situation where the rankings are allowed to vary over time. Time dependence in preference data occurs when assessors are asked to rank the same set of items repeatedly over time which can result in an assessor's preferences being altered. Asfaw et al. (2017) created the framework for the Mallows model for time-dependent rank data. The consensus ranking at each time $\boldsymbol{\rho}_t$ was assumed to change little between consecutive time steps to allow the use of a smooth transition kernel between latent consensus rankings. This is modelled under another Mallows model,

$$p(\boldsymbol{\rho}^{(t)}|\boldsymbol{\rho}^{(t-1)}) = \frac{1}{Z_n(\beta)} \exp\Big\{ -\frac{\beta}{n} d(\boldsymbol{\rho}^{(t)}, \boldsymbol{\rho}^{(t-1)}) \Big\}.$$

Similarly, for the scale parameters at each time step, a normal distribution acts as the smoothing kernel between time steps. The inference is performed at the individual level by sampling from a posterior distribution using a Metropolis-within-Gibbs algorithm. Asfaw et al. (2017) considered a case study for the model that focused on updating the overall ranking of students based on their school performance test results over four years. The study also factored in the absence of students from the tests, creating some missing data.

Eliciting Informative Priors

Some progress has been made in selecting informative priors for the consensus ranking in the Mallows model with the Spearman distance (Crispino and Antoniano-Villalobos, 2022) when the scale parameter is known or unknown. Typically, the uniform distribution is the prior density for the consensus ranking. The use of different informative priors for each distance function is still an ongoing research direction for the Bayesian Mallows model.

Clicking data

The Bayesian Mallows model typically performs inference with explicit forms of data rather than implicit data. Implicit data can be in the form of clicks, listens/watches, purchases, likes, etc., and are interpreted to indicate a preference for a particular item. Liu et al. (2019b) develop a Bayesian Mallows model for clicking data to perform inference using click observations as preference data in a recommender system setting. They assume that each user j has clicked on a subset of items $\mathcal{A}_j \subseteq \mathcal{A}$ with $|\mathcal{A}_j| = c_j$ clicks. The number of times an item had been clicked indicates the popularity of the item, so any clicked item is preferred over any unclicked item. This information can be transformed into pairwise comparison data and full rankings. Each user j has a set of possible full rankings based on their clicking data, $\mathcal{S}_j(\mathcal{A}_j) = \{\tilde{\mathbf{R}}_j \in \mathcal{P}_n : \tilde{R}_{ij} < \tilde{R}_{kj} \text{ if } A_i \in \mathcal{A}_j \text{ and } A_k \in \mathcal{A}_j^c,\ \forall i, k,\ i \neq k\}$. Therefore, the objective is to sample from the posterior distribution

$$p(\boldsymbol{\rho}_{1:C}, \alpha_{1:C}, z_{1:M} | \mathcal{A}_{1:M}) = \sum_{\tilde{\mathbf{R}}_1 \in \mathcal{S}_1(\mathcal{A}_1)} \cdots \sum_{\tilde{\mathbf{R}}_M \in \mathcal{S}_M(\mathcal{A}_M)} p(\boldsymbol{\rho}_{1:C}, \alpha_{1:C}, z_{1:M}, \tilde{\mathbf{R}}_{1:M} | \mathcal{A}_{1:M}).$$

This particular model is an example of an alternative to using collaborative filtering (Koren et al., 2009) in a personalised recommendation setting where we are interested in providing accuracy and diversity in the recommendations for a particular user.

Chapter 3

Monte Carlo Methods

3.1 Introduction

A typical problem in statistics is to find the expectation of a function or a summary statistic. Suppose that X is a random variable with probability density function $f(x)$ and that we are interested in evaluating the expectation of a measurable test function $h(x) : \mathcal{X} \to \mathbb{R}$ with respect to f. The expected value of $h(x)$, with respect to f, can be found by evaluating the integral

$$f(h) = \mathbb{E}_f[h(X)] = \int_{\mathcal{X}} h(x)f(x)dx. \tag{3.1.1}$$

However, in many cases, the integral (3.1.1) cannot be evaluated analytically, and the use of deterministic numerical approximation methods is not feasible for high-dimensional spaces; this problem is known as the curse of dimensionality.

Instead, we can use Monte Carlo methods, which are a class of numerical methods that use simulation and random sampling from probability densities to perform Monte Carlo integration, such as evaluating (3.1.1). In this chapter, we review several Monte Carlo methods that can be used to evaluate integrals of expectations of functions or probability events. Many textbooks on this subject matter are available, and we refer the reader to Robert and Casella (2004) for further details.

3.2 Perfect Monte Carlo

Perfect Monte Carlo assumes that if we can obtain a large number of independent identically distributed (i.i.d.) samples, $\{x^{(i)}\}_{i=1}^{N} = x^{(1)}, \ldots, x^{(N)}$, from $f(x)$, then we can calculate the empirical approximation of the target density as

$$\hat{f}(dx) = \frac{1}{N}\sum_{i=1}^{N} \delta_{x^{(i)}}(dx), \tag{3.2.1}$$

where $\delta_{x^{(i)}}(dx)$ is the Dirac delta function located at $x^{(i)}$. Given the result from (3.2.1), we can also evaluate (3.1.1) to obtain the Monte Carlo estimate of the expectation of the test function

$$\hat{f}(h) = \int_{\mathcal{X}} h(x)\hat{f}(dx) = \frac{1}{N}\sum_{i=1}^{N} h(x^{(i)}). \tag{3.2.2}$$

There are several desirable properties of perfect Monte Carlo, which have been summarised in Robert and Casella (2004). First, it can be shown that the Monte Carlo estimate, $\hat{f}(h)$, is an unbiased estimator for $\mathbb{E}_f[h(X)]$ and the Monte Carlo variance decreases as the sample size N increases. By the Strong Law of Large Numbers, as the sample size increases then $\hat{f}(h)$ will converge almost surely (a.s.) to $\mathbb{E}_f[h(X)]$;

that is,

$$\hat{f}(h) \xrightarrow{a.s.} \mathbb{E}_f[h(X)] \quad \text{as} \quad N \to \infty. \tag{3.2.3}$$

Furthermore, if we assume the variance is finite, i.e. $\text{Var}(h(X)) < \infty$, then the Central Limit Theorem implies that

$$\sqrt{N}(\hat{f}(h) - \mathbb{E}_f[h(X)]) \xrightarrow{\mathcal{D}} \mathcal{N}(0, \text{Var}[h(X)]) \quad \text{as} \quad N \to \infty,$$

where $\xrightarrow{D}$ denotes the convergence in distribution. Monte Carlo approximation is a popular choice due to its simplicity and useful properties, but these results are only possible if we are able to draw i.i.d. samples directly from the target distribution. This is not always possible and so we need to consider alternative sampling methods.

3.3 Importance sampling

In some cases, the target density is not possible to sample directly from, but it can be evaluated pointwise. When this occurs, we can use importance sampling which generates i.i.d. samples from an alternative distribution, $q(\cdot)$, which is often referred to as the proposal or importance distribution. The samples from the proposal density are used to estimate the target density in (3.2.1) as

$$\hat{f}(dx) = \frac{1}{N} \sum_{i=1}^{N} \frac{f(x^{(i)})}{q(x^{(i)})} \delta_{x^{(i)}}(dx). \tag{3.3.1}$$

Here, we can define the importance weight function as the ratio $W(\cdot) = f(\cdot)/q(\cdot)$. The weight of each sample reflects its contribution to the target distribution, and we see that the samples empirically estimate $\hat{f}(x)$ as a weighted sample $\{W(x^{(i)}), x^{(i)}\}_{i=1}^{N}$, so (3.3.1) can be rewritten as

$$\hat{f}(dx) = \frac{1}{N}\sum_{i=1}^{N} W(x^{(i)})\delta_{x^{(i)}}(dx). \tag{3.3.2}$$

We can also rewrite the Monte Carlo estimate in (3.1.1) as

$$\begin{aligned}\mathbb{E}_f[h(X)] &= \int_{\mathcal{X}} h(x)\frac{f(x)}{q(x)}q(x)dx = \mathbb{E}_q\Big[h(x)\frac{f(x)}{q(x)}\Big] = \mathbb{E}_q\Big[h(x)W(x)\Big] \\ &= \frac{1}{N}\sum_{i=1}^{N} W(x^{(i)})h(x^{(i)}).\end{aligned}$$

Sometimes the target distribution $f(x)$ has an unknown normalisation constant, that is, $f(x) = \tilde{f}(x)/Z$ may be known up to a constant of proportionality, where $\tilde{f}(x)$ is the unnormalised density and Z is the normalisation constant. Under the importance sampling scheme, we can rewrite the target density as

$$f(x) = \frac{\tilde{f}(x)}{Z} = \frac{w(x)q(x)}{Z} = \frac{w(x)q(x)}{\int w(x)q(x)dx},$$

where $w(x) = \frac{\tilde{f}(x)}{q(x)}$. Typically, the weights are normalised by prescribing $W(x^{(i)}) = w(x^{(i)})/\sum_{i=1}^{N} w(x^{(i)})$, to ensure that $\mathbb{E}_q[w(X)] = 1$. The approximate target density is (3.3.2), and the Monte Carlo estimate is

$$\hat{f}(h) = \frac{\sum_{i=1}^{N} h(x^{(i)}) f(x^{(i)})/q(x^{(i)})}{\sum_{i=1}^{N} f(x^{(i)})/q(x^{(i)})} = \sum_{i=1}^{N} W(x^{(i)}) h(x^{(i)}).$$

The pseudocode for the importance sampler is provided in Algorithm 2.

```
Algorithm 2: Importance Sampling
  Input: $f,\ h,\ q,\ N$.
  Output: Monte Carlo estimate of $\hat{f}(h)$.
  Draw i.i.d. samples $\{x^{(i)}\}_{i=1}^{N} \sim q(\cdot)$.
  Calculate the importance weights $w^{(i)} = f(x^{(i)})/q(x^{(i)})\ \forall i$.
  Normalise the importance weights $W^{(i)} = w^{(i)}/\sum_{i=1}^{N} w^{(i)}\ \forall i$.
  Calculate the Monte Carlo estimate $\hat{f}(h) = \sum_{i=1}^{N} W(x^{(i)}) h(x^{(i)})$.
```

The main advantage of using importance sampling is that we can select a proposal density that is easier to simulate than the target distribution, and given the same set of samples $\{x^{(1)}, \ldots, x^{(N)}\} \sim q$, we can potentially evaluate different test or target distributions. Although there is bias in the Monte Carlo estimate, this decreases as the number of samples increases by the Strong Law of Large Numbers.

The chosen proposal distribution should be easy to simulate, but the main constraint is that the support of q covers the support of f, i.e., $f(x) > 0 \Rightarrow q(x) > 0$, to ensure that the expectation of the test function over $f(x)$ can instead be evaluated over $q(x)$. An appropriate proposal distribution will achieve a consistent estimate of $\hat{f}(h)$, but the variance of the estimator depends on the choice of q, since

$$\mathrm{Var}_q[\hat{f}(h)] = \frac{1}{N}\mathrm{Var}_q[w(x)h(x)] = \frac{1}{N}\mathbb{E}_q[w(x)^2 h^2(x)] - \mathbb{E}_f[h(x)]^2. \tag{3.3.3}$$

To minimise (3.3.3), the term $\mathbb{E}_q[w(x)^2h^2(x)]$ must be minimised. Jensen's inequality shows that a lower bound is

$$\mathbb{E}_q[w(x)^2h^2(x)] \geq \mathbb{E}_q[w(x)|h(x)|]^2 = \left(\int_{\mathcal{X}} |h(x)|f(x)dx\right)^2$$

and Geweke (1989) shows that the optimal choice of q which minimises the variance of the estimator $\mathbb{E}_f[h(X)]$ and obtains the lower bound is

$$q^*(x) = \frac{|h(x)|f(x)}{\int_{\mathcal{X}} |h(z)|f(z)dz}.$$

The proof of these results is given in Robert and Casella (2004).

3.4 Markov chain Monte Carlo

In this section, we review Markov chain Monte Carlo (MCMC) methods which do not use i.i.d. samples but dependent samples instead. These are generated in a Markov chain to approximate the integrals of interest. A Markov chain contains a sequence of samples $x^{(1)}, x^{(2)}, \ldots, x^{(N)}$, where the probability distribution of each $x^{(i)}$ given the previous samples depends only on the previous sample $x^{(i-1)}$; in other words,

$$p(x^{(i)}|x^{(i-1)}, x^{(i-2)}, \ldots, x^{(1)}) = p(x^{(i)}|x^{(i-1)}).$$

Each sample in the Markov chain is generated by sampling from a conditional distribution

$$x^{(i)} \sim K(x^{(i-1)}, x^{(i)}),\ i = 1, 2, \ldots, N,$$

where $K(\cdot, \cdot)$ is known as the transition kernel. As the Markov chain advances, it converges to a probability distribution known as the stationary distribution or invariant distribution. If the target distribution $f(x)$ is the invariant distribution, then $x^{(i-1)} \sim f(\cdot) \Rightarrow x^{(i)} \sim f(\cdot)$, and the transition kernel satisfies the property

$$\int_{\mathcal{X}} K(x', x) f(x) dx = f(x').$$

Given a Markov chain that is irreducible, aperiodic and has a stationary distribution $f(x)$, then it has the ergodic property,

$$\bar{h}_N = \frac{1}{N} \sum_{i=1}^{N} h(x^{(i)}) \rightarrow \mathbb{E}_f[h(X)],$$

with probability 1 as $N \rightarrow \infty$, where $\bar{h}_N$ is the ergodic average. Further details on the properties of Markov chains can be found in Robert and Casella (2004).

An MCMC sampler constructs a Markov chain that has the target distribution as the invariant distribution. The sampler is designed so that it satisfies detailed balance, that is, for a Markov chain, $\{x^{(i)}\}_{i=1}^{N}$, with a transition kernel, $K(x, x')$, then

$$K(x, x') f(x) = K(x', x) f(x'),\ \text{for } x, x' \in \mathcal{X}. \tag{3.4.1}$$

When designing a sampler, it must converge quickly. In practice, when the MCMC sampler is run, the initial sample $x^{(0)}$ is set in a region of $f(x)$ with high probability to ensure that it converges quickly to the invariant distribution. However, this is not always possible, so we obtain a large number of samples and discard the first $n < N$ samples as burn-in before calculating any summary statistics of interest. We can also, if we have a large Markov chain, select every k^{th} sample in the chain as a post-processing step, known as thinning, to reduce the dependence between samples.

3.4.1 Gibbs sampling

Gibbs sampling (Geman and Geman, 1984), also known as alternating conditional sampling, is one example of an MCMC algorithm that is often used for multivariate problems where we have several variables $\mathbf{x} = (x_1, \ldots, x_k)$ to sample from. To start the Gibbs sampler we an initial value for each component is selected, i.e., $\mathbf{x}^{(0)} = (x_1^{(0)}, \ldots, x_k^{(0)})$. Then, in each iteration, we sample a new value of each component $x_j^{(i)}$ conditional on the remaining components $f(x_j|\mathbf{x}_{-j}^{(i-1)})$, where $\mathbf{x}_{-j}^{(i-1)} = (x_1^{(i-1)}, \ldots, x_{j-1}^{(i-1)}, x_{j+1}^{(i-1)}, \ldots, x_k^{(i-1)})$. The sampling steps in each iteration are often referred to as Gibbs updates, and when we update each variable, we use the most recent values of the other components. The pseudocode for the Gibbs sampler is provided in Algorithm 3.

Algorithm 3: Gibbs Sampling

```
Input: $f$, $h$, $N$.
Output: Monte Carlo estimate of $\hat{f}(h)$.
Initialise: $\mathbf{x}^{(0)} = (x_1^{(0)}, \ldots, x_k^{(0)})$.
for $i = 1, \ldots, N$ do
    Generate $x_1^{(i)} \sim f(x_1 | x_2^{(i-1)}, \ldots, x_k^{(i-1)})$.
    Generate $x_2^{(i)} \sim f(x_2 | x_1^{(i)}, x_3^{(i-1)}, \ldots, x_k^{(i-1)})$.
    ...
    Generate $x_k^{(i)} \sim f(x_k | x_1^{(i)}, \ldots, x_{k-1}^{(i)})$.
end
```

3.4.2 Metropolis-Hastings

The Metropolis-Hastings algorithm (Metropolis et al., 1953; Hastings, 1970) is another example of an MCMC algorithm. A key aspect of the algorithm is the proposal distribution $q(x'|x)$, which draws new samples, x', given the current value in the Markov chain, x. The proposal distribution is easy to sample from and is known up to a constant of proportionality; however, it is not necessarily symmetric, that is, $q(x'|x) \neq q(x|x')$. At the i^{th} iteration, given the current value in the Markov chain, $x^{(i-1)} = x$, we propose a new sample x' using $q(x'|x)$, and it is either accepted with probability

$$\alpha(x, x') = \min\left\{1, \frac{f(x')q(x|x')}{f(x)q(x'|x)}\right\}, \tag{3.4.2}$$

as the next value in the Markov chain, meaning $x^{(i)} = x'$, or rejected. If the proposal is rejected, we set $x^{(i)} = x$ as the next value in the Markov chain. The pseudocode for the Metropolis-Hastings algorithm is provided in Algorithm 4.

Algorithm 4: Metropolis-Hastings

```
Input: $f,\ h,\ q,\ N$.
Output: Monte Carlo estimate of $\hat{f}(h)$.
Initialise: $x^{(0)}$.
for $i = 1, \dots, N$ do
    Sample proposal $x' \sim q(x'|x^{(i-1)})$
    Calculate acceptance probability $\alpha(x^{(i-1)}, x') = \min\left\{1, \frac{f(x')q(x^{(i-1)}|x')}{f(x^{(i-1)})q(x'|x^{(i-1)})}\right\}$
    Accept $x'$ with probability $\alpha(x^{(i-1)}, x')$ and set $x^{(i)} = x'$. Otherwise, set
      $x^{(i)} = x^{(i-1)}$
end
```

The Metropolis-Hastings algorithm is a generalised version of the Gibbs sampler, but the two algorithms are popular variants of MCMC. If we consider the Gibbs sampler for the multivariate problem and imagine that we are aiming to update each component of $f(\mathbf{x})$ individually, then we can rewrite $f(\mathbf{x}) = f(x'_j|\mathbf{x}_{-j})f(\mathbf{x}_{-j})$ because only the j^{th} component can vary. We can also set the proposal $q(\mathbf{x}'|\mathbf{x}) = q(x'_j, \mathbf{x}'_{-j}|x_j, \mathbf{x}_{-j}) = f(x'_j|\mathbf{x}_{-j})$ which results in the following acceptance probability

$$
\begin{aligned}
\alpha(x, x') &= \min\left\{1, \frac{f(\mathbf{x}')q(\mathbf{x}|\mathbf{x}')}{f(\mathbf{x})q(\mathbf{x}'|\mathbf{x})}\right\} \\
&= \min\left\{1, \frac{f(x'_j|\mathbf{x}'_{-j})f(\mathbf{x}'_{-j})q(x_j, \mathbf{x}_{-j}|x'_j, \mathbf{x}'_{-j})}{f(x_j|\mathbf{x}_{-j})f(\mathbf{x}_{-j})q(x'_j, \mathbf{x}'_{-j}|x_j, \mathbf{x}_{-j})}\right\} \\
&= \min\left\{1, \frac{f(x'_j|\mathbf{x}'_{-j})f(\mathbf{x}'_{-j})f(x_j|\mathbf{x}_{-j})}{f(x_j|\mathbf{x}_{-j})f(\mathbf{x}_{-j})f(x'_j|\mathbf{x}'_{-j})}\right\} \qquad (3.4.3) \\
&= 1.
\end{aligned}
$$

We see that the terms in (3.4.3) cancel because $\mathbf{x}'_{-j} = \mathbf{x}_{-j}$. This means that we always accept the proposal in the Gibbs sampler as only the j^{th} component is updated.

The Metropolis-Hastings algorithm creates a Markov chain which is guaranteed to converge to a unique stationary distribution that is the target density. This is shown using detailed balance. If $f(x)$ is the stationary distribution of a Markov chain with transition kernel $K(x, x')$, then it can be shown that (3.4.1) is satisfied. The transition kernel for the Metropolis-Hastings algorithm is defined as $K(x, x') = q(x'|x)\alpha(x, x') + \big(1 - \int \alpha(x, y)q(y|x)dy\big)\delta_x(dx')$, where the acceptance probability is defined as (3.4.2). If we reject x', then it is trivial to show that

$$\left(1 - \int \alpha(x, y)q(y|x)dy\right)\delta_x(dx') = \left(1 - \int \alpha(x', y)q(y|x')dy\right)\delta_{x'}(dx)$$

satisfies detailed balance. Otherwise, if we accept x', then it can be shown that

$$\begin{aligned}
K(x, x')f(x) &= q(x'|x)\alpha(x, x')f(x) \\
&= q(x'|x)\min\left\{1, \frac{f(x')q(x|x')}{f(x)q(x'|x)}\right\}f(x) \\
&= \min\{f(x)q(x'|x), f(x')q(x|x')\} \\
&= \min\left\{\frac{f(x)q(x'|x)}{f(x')q(x|x')}, 1\right\}f(x')q(x|x') \\
&= \alpha(x', x)q(x|x')f(x') \\
&= K(x', x)f(x').
\end{aligned}$$

3.5 Sequential Monte Carlo methods

We have shown how Monte Carlo methods can be employed to evaluate integrals of interest in statistics. MCMC samplers are a widely-used Monte Carlo method for Bayesian inference when we are interested in quantifying aspects of posterior distributions that cannot be directly calculated. However, there are some drawbacks to using MCMC. First, the method can suffer from slow convergence, making it difficult to know low long to run the chain if it is slow-mixing. If we have an online problem, where we observe a data stream from a target distribution, then we cannot use MCMC to perform recursive estimation; the last MCMC sample cannot be used, and so we have to rerun the sampler each time we receive new data. This motivates the need for sequential Monte Carlo (SMC) methods that can perform Monte Carlo integration on a sequential basis.

SMC methods are a class of sampling algorithms that estimate a target distribution of interest sequentially. Each target distribution is approximated by a collection of random samples, defined as particles. The particles evolve at each iteration using importance sampling and resampling steps. An introduction to SMC methods can be found in Doucet et al. (2001), and summaries on their development are discussed in Del Moral et al. (2006); Doucet and Johansen (2009); Naesseth et al. (2019).

The literature on SMC typically discusses how these methods obtain approximations of the filtering densities in state-space models and are often referred to as particle filters in this context. In standard Bayesian inference problems, SMC methods have also been considered as an alternative method to using MCMC (Neal (2001); Chopin (2002); Del Moral et al. (2006)) to estimate a sequence of posterior distributions in an online setting. In this thesis, we are interested in the latter application, so we acknowledge that the review of SMC methods for state-space models is not fully comprehensive, but the methods illustrate how SMC methods have developed over time. We begin by introducing the main two concepts, sequential importance sampling and resampling, and describe the applications, where these methods have been used, before summarising the general SMC framework.

3.5.1 Sequential importance sampling

First, we describe how importance sampling (see Section 3.3) is applied sequentially to estimate a sequence of target densities. In sequential importance sampling (SIS) (Liu and Chen, 1998), we denote a sequence of target distributions to estimate as $f_t(\cdot),\ t = 1, \ldots, T$. A sequence of proposal distributions is also constructed with the following recursion,

$$\begin{aligned} q_t(x_{1:t}) &= q_{t-1}(x_{1:t-1})q_t(x_t|x_{1:t-1}) \\ &= q_1(x_1)\prod_{k=2}^{t} q_k(x_k|x_{1:k-1}). \end{aligned}$$

At time $t-1$, we have N samples, henceforth referred to as particles, and their associated normalised importance weights in a particle set $\{x_{t-1}^{(i)}, W_{t-1}^{(i)}\}_{i=1}^{N}$ that approximate the density $f_{t-1}(x_{1:t-1})$. The assumption in SIS is that if any two consecutive distributions f_{t-1} and f_t are relatively similar, then we should be able to propagate the particles appropriately (Del Moral et al., 2006). The next target distribution, $f_t(x_{1:t})$, is estimated by sampling the new particle values $\{x_t^{(i)}\}_{i=1}^{N}$ from the proposal distribution $q_t(\cdot|x_{1:t-1}^{(i)})$, storing the paths of each particle, $x_{1:t}^{(i)} = \{x_{1:t-1}^{(i)}, x_t^{(i)}\}_{i=1}^{N}$, and adjusting the weights accordingly to reflect the current target density. The unnormalised importance weights can also be calculated as a recursion

$$\begin{aligned}
w_t(x_{1:t}) &= \frac{\tilde{f}_t(x_{1:t})}{q_t(x_{1:t})} \\
&= \frac{\tilde{f}_{t-1}(x_{1:t-1})}{q_{t-1}(x_{1:t-1})} \frac{\tilde{f}_t(x_{1:t})}{\tilde{f}_t(x_{1:t-1})q_t(x_t|x_{1:t-1})} \\
&= w_{t-1}(x_{1:t-1}) \ \frac{\tilde{f}_t(x_{1:t})}{\tilde{f}_t(x_{1:t-1})q_t(x_t|x_{1:t-1}).}
\end{aligned}$$

Typically the particle weights are normalised after the adjustment, $W_t^{(i)} = w_t^{(i)} / \sum_{i=1}^N w_t^{(i)}$, and we repeat the procedure for each time step. We present the algorithm for SIS in Algorithm 5.

Algorithm 5: Sequential Importance Sampling

```
Input: $\tilde{f}_{1:T},\ h,\ q_{1:T},\ N.$
Output: Monte Carlo estimate of $\hat{f}(h)$.
for $t = 1$ do
    Sample particles, $\{x_1^{(i)}\}_{i=1}^N \sim q_1(\cdot)$.
    Calculate importance weights, $w_1^{(i)} = \tilde{f}_1(x_1^{(i)})/q_1(x_1^{(i)})\ \forall i$.
    Normalise the importance weights $W_1^{(i)} = w_1^{(i)} / \sum_{i=1}^N w_1^{(i)}\ \forall i$.
end
for $t = 2, \dots, T$ do
    Sample particles, $\{x_t^{(i)}\}_{i=1}^N \sim q_t(\cdot|x_{1:t-1}^{(i)})$.
    Calculate importance weights, $w_t^{(i)} = W_{t-1}(x_{1:t-1})\ \frac{\tilde{f}_t(x_{1:t})}{\tilde{f}_t(x_{1:t-1})q_t(x_t|x_{1:t-1}),}\ \forall i$.
    Normalise the importance weights $W_t^{(i)} = w_t^{(i)} / \sum_{i=1}^N w_t^{(i)}\ \forall i$.
end
```

Particle degeneracy

With each iteration, the variance of the importance weights will increase over time. This can result in a few particles, or even a single particle, being assigned significant weight to approximate the target density whilst an increasing number of particles will have negligible weight. This problem is known as particle degeneracy and causes the accuracy of the approximation of the target distribution to deteriorate with each iteration. In practice, a resampling step is incorporated to replicate the heavier weighted particles and discard the particles with negligible weight.

3.5.2 Sequential importance resampling

Resampling (with replacement) is used in SMC to overcome the problem of particle degeneracy, as resampling aims to improve the approximation of the target distribution by taking a sample of the current approximation (Doucet and Johansen, 2009). We resample N times from the particle set, $\{x_t^{(i)}, W_t^{(i)}\}_{i=1}^N$, to obtain a new equally weighted particle set, reducing the variance of the weights. The particles with negligible weights are discarded, and those with heavier weights are replicated. The main advantage of resampling particles is that when calculating estimates of interest, the computational cost of particles with negligible weights is removed whilst we can use the more informative particles. This step is incorporated into the SIS scheme after the particles have been reweighted from f_{t-1} to f_t, creating the sequential importance resampling (SIR) framework (see Algorithm 6).

Several resampling schemes have been considered in the SMC literature. Gordon et al. (1993) uses multinomial resampling where the expected number of copies of a particle is proportional to its weight. Since then, alternative resampling schemes have been proposed, such as systematic (Kitagawa, 1996), residual (Liu and Chen, 1998), and stratified resampling (Carpenter et al., 1999), and are described in Douc and Cappé (2005); Doucet and Johansen (2009). It is shown in Douc and Cappé (2005) that residual and stratified resampling improves over multinomial resampling by having a lower conditional variance. Although systematic resampling lacks full theoretical analysis, it is also preferred over multinomial resampling because it can give comparable results, and is easier to implement.

There may be a possibility that resampling is unnecessary. In some cases, repeated resampling can lead to particle impoverishment (Berzuini and Gilks, 2003) on the set of particles, which occurs when too few distinct particle values represent the given target distribution. This can be avoided by carefully choosing how and when we resample. One option is to observe the variance of the importance weights; a small variance indicates a good approximation of the target distribution. Another option is to monitor the effective sample size (ESS) (Kong et al., 1994),

$$\mathrm{ESS} = \frac{1}{\sum_{i=1}^{N} (W_t^{(i)})^2}.$$

ESS can take a value between 1 and N, and we typically resample when it is below a specified threshold. A typical suggestion, proposed by Liu (2001), is to resample if $\mathrm{ESS} < N/2$.

```
Algorithm 6: Sequential Importance Resampling
Input: $\tilde{f}_{1:T},\ h,\ q_{1:T},\ N$.
Output: Monte Carlo estimate of $\hat{f}(h)$.
for $t = 1$ do
    Sample particles, $\{x_1^{(i)}\}_{i=1}^N \sim q_1(\cdot)$.
    Calculate importance weights, $w_1^{(i)} = \tilde{f}_1(x_1^{(i)})/q_1(x_1^{(i)})\ \forall i$.
    Normalise the importance weights $W_1^{(i)} = w_1^{(i)} / \sum_{i=1}^N w_1^{(i)}\ \forall i$.
    if Resampling condition is met then
        Resample particles to obtain $\{x_1^{(i)}, W_1^{(i)} = 1/N\}_{i=1}^N$.
    end
end
for $t = 2, \dots, T$ do
    Sample particles, $\{x_t^{(i)}\}_{i=1}^N \sim q_t(\cdot|x_{1:t-1}^{(i)})$.
    Calculate importance weights, $w_t^{(i)} = \tilde{f}_t(x_{1:t}^{(i)})/q_t(x_{1:t}^{(i)})\ \forall i$.
    Normalise the importance weights $W_t^{(i)} = W_{t-1}(x_{1:t-1})\ \frac{\tilde{f}_t(x_{1:t})}{\tilde{f}_t(x_{1:t-1})q_t(x_t|x_{1:t-1}),}\ \forall i$.
    if Resampling condition is met then
        Resample particles to obtain $\{x_t^{(i)}, W_t^{(i)} = 1/N\}_{i=1}^N$.
    end
end
```

3.5.3 Particle filters

State space models

The original application for SMC methods was to perform analysis on state space models (SSMs), known in this context as particle filters. SSMs are a class of models for time series data. They represent a sequence of observations, $\{y_t\}_{t=1}^T$, which are conditionally independent given a sequence of latent variables $\{x_t\}_{t=0}^T$ and are generated by the observation density $g_\theta(y_t|x_t)$. The unobserved states of the system $\{x_t\}_{t=0}^T$ are dependent on their previous states and are generated by the state evolution density $f_\theta(x_t|x_{t-1})$ with an initial latent state density $\mu_\theta(x_0)$. Both densities are dependent on the model parameters θ which are assumed to be known. The Markovian dependencies in a general SSM can be represented graphically as in Figure 3.5.1 and algebraically,

$$
\begin{aligned}
&X_0 \sim \mu_\theta(x_0), \\
&X_t|X_{t-1} = x_t \sim f_\theta(x_t|x_{t-1}), \quad \text{for } t = 1, \dots, T, \\
&Y_t|X_t = y_t \sim g_\theta(y_t|x_t), \quad \text{for } t = 1, \dots, T.
\end{aligned}
$$

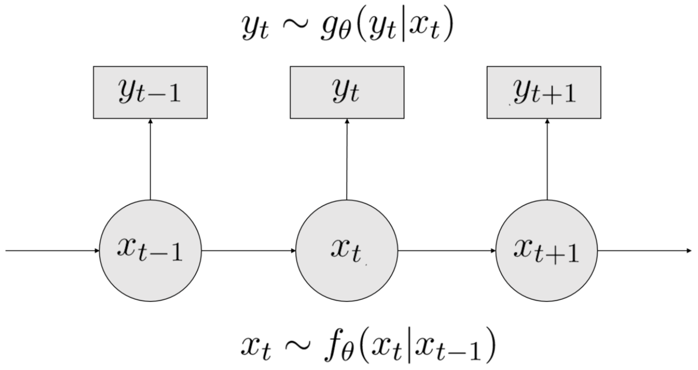

$y_t \sim g_\theta(y_t|x_t)$
y_{t-1}
y_t
y_{t+1}
x_{t-1}
x_t
x_{t+1}
$x_t \sim f_\theta(x_t|x_{t-1})$

Figure 3.5.1: A graphical representation of the dependence structure between the states and the observations in a general SSM.

In this framework, we are interested in inference about $\{x_t\}_{t=1}^T$ given the sequence of observations $\{y_t\}_{t=1}^T$. This is conducted using the posterior distribution $p_\theta(x_{0:t}|y_{1:t})$ which is proportional to the joint distribution $p_\theta(x_{0:t}, y_{1:t})$,

$$
\begin{aligned}
p_\theta(x_{0:t}|y_{1:t}) &= \frac{p_\theta(x_{0:t}, y_{1:t})}{p_\theta(y_{1:t})} \\
&\propto \mu_\theta(x_0) \prod_{k=1}^{t} f_\theta(x_k|x_{k-1}) \prod_{k=1}^{t} g_\theta(y_k|x_k).
\end{aligned}
$$

Typically, we are interested in estimating the marginal posterior, or filtered densities,

$\{p_\theta(x_t|y_{1:t})\}_{t=1}^T$. Alternatively, we may be interested in estimating the smoothing densities, $\{p_\theta(x_t|y_{1:T})\}_{t=1}^T$. Using Bayes' theorem, the posterior can be updated through the recursion

$$\begin{aligned} p_\theta(x_t|y_{1:t}) &= \int \frac{f_\theta(x_t|x_{t-1})g_\theta(y_t|x_t)}{p_\theta(y_t|y_{1:t-1})} p_\theta(x_{t-1}|y_{1:t-1})dx_{t-1} \\ &\propto \int f_\theta(x_t|x_{t-1})g_\theta(y_t|x_t)p_\theta(x_{t-1}|y_{1:t-1})dx_{t-1}, \end{aligned} \tag{3.5.1}$$

where the predictive likelihood is given as

$$p_\theta(y_t|y_{1:t-1}) = \int f_\theta(x_t|x_{t-1})g_\theta(y_t|x_t)p_\theta(x_{t-1}|y_{1:t-1})dx_{t-1:t}.$$

Inference for SSMs was first discussed by Kalman (1960), where $f_\theta(x_t|x_{t-1})$ and $g_\theta(y_t|x_t)$ are Gaussian models. These are known as Kalman filters and an exact expression for (3.5.1) can be calculated. If instead $f_\theta(x_t|x_{t-1})$ and $g_\theta(y_t|x_t)$ are not Gaussian, then it is difficult to find an analytical expression for (3.5.1). This has encouraged the use of Monte Carlo methods, such as MCMC and SMC, to estimate each latent state x_t given the observations $y_{1:t}$. SMC methods were first introduced in Gordon et al. (1993) and improvements in the methodology are given, for example, in Kitagawa (1996); Liu and Chen (1998); Pitt and Shephard (1999); Carpenter et al. (1999); Doucet et al. (2001); Fearnhead (2002). A review of these methods is presented in Godsill and Clapp (2001); Cappé et al. (2007); Doucet and Johansen (2009).

In each iteration of SMC, we have a weighted particle set $\{W_{t-1}^{(i)}, x_{t-1}^{(i)}\}_{i=1}^N$ that approximates the marginal posterior or filtered density,

$$\hat{p}_\theta(x_t|y_{1:t}) \propto \sum_{i=1}^{N} W_{t-1}^{(i)} f_\theta(x_t^{(i)}|x_{t-1}^{(i)}) g_\theta(y_t|x_t^{(i)}).$$

To approximate the next latent state upon observing y_t, the particles are propagated from time $t-1$ to t by sampling $\{x_t^{(i)}\}_{i=1}^N$ from a proposal distribution $q_\theta(x_t|x_{1:t-1}, y_t)$ so that the paths of the particles can be updated $x_{1:t}^{(i)} = \{x_{1:t-1}^{(i)}, x_t^{(i)}\}$. The particle weights are also updated,

$$w_t^{(i)} \propto W_{t-1}^{(i)} \frac{g_\theta(y_t|x_t^{(i)}) f_\theta(x_t^{(i)}|x_{t-1}^{(i)})}{q_\theta(x_t^{(i)}|x_{1:t-1}^{(i)}, y_t)}, \; i = 1, \ldots, N,$$

and are normalised after each iteration. The pseudocode for the general particle filter is presented in Algorithm 7.

Algorithm 7: General Particle Filter

```
Input: $f_{1:T},\ g,\ \mu,\ q_{1:T},\ N$.
Output: Monte Carlo estimate of $\hat{p}_\theta(x_t|y_{1:t}) \propto \sum_{i=1}^N W_t^{(i)} f_\theta(x_t^{(i)}|x_{t-1}^{(i)}) g_\theta(y_t|x_t^{(i)})$.
Sample particles, $\{x_0^{(i)}\}_{i=1}^N \sim \mu_0(\cdot)$ and set $\{w_0 = 1/N\}_{i=1}^N$.
for $t = 1, \ldots, T$ do
    Sample particles, $\{x_t^{(i)}\}_{i=1}^N \sim q_\theta(\cdot|x_{1:t-1}^{(i)}, y_t)$.
    Calculate importance weights, $w_t^{(i)} \propto W_{t-1}^{(i)} \frac{g_\theta(y_t|x_t^{(i)}) f_\theta(x_t^{(i)}|x_{t-1}^{(i)})}{q_\theta(x_t^{(i)}|x_{1:t-1}^{(i)}, y_t)}$ $\forall i$.
    Normalise the importance weights $W_t^{(i)} = w_t^{(i)} / \sum_{i=1}^N w_t^{(i)}$ $\forall i$.
    if Resampling condition is met then
        Resample particles to obtain $\{x_t^{(i)}, W_t^{(i)} = 1/N\}_{i=1}^N$.
    end
end
```

Parameter estimation of θ

In Section 3.5.3 it was assumed that the model parameters, θ, were known and the focus was to estimate the latent states given a sequence of observations. However, in some cases, the model parameters θ used to generate the latent states and the observations are unknown. Several approaches, both Bayesian and frequentist, have been considered for this problem, depending on whether data are observed in an offline or online manner. In a Bayesian approach, the unknown parameter is assigned a prior distribution, so the posterior density of θ given the observations is $p(\theta|y_{1:t})$. Inference for θ is performed using the joint posterior density in the offline case or by using the sequence of posterior densities in the online case as more observations become available. If we can compute $p(x_{1:t}, y_{1:t})$, and in turn $p(x_t|y_{1:t})$, then we can sequentially update the marginal likelihood

$$p_\theta(y_{1:t}) = p_\theta(y_1) \prod_{k=2}^{t} p_\theta(y_k|y_{1:k-1}),$$

and the model parameters can be estimated as follows

$$p(\theta|y_{1:t}) = \frac{p(\theta)p(y_{1:t}|\theta)}{\int p(\theta)p_\theta(y_{1:t})d\theta} \propto p(\theta)p_\theta(y_{1:t}).$$

This problem has been considered in Gordon et al. (1993); Berzuini and Gilks (2001); Liu and West (2001); Storvik (2002); Fearnhead (2002) and an overview of existing methods is provided in Goel et al. (2005); Cappé et al. (2007); Kantas et al. (2009, 2015). However, the online Bayesian inference problem remains a challenge because the methods suffer from the degeneracy problem since θ does not evolve through time (see Section 3.5.1 for more details).

3.5.4 Sequential Monte Carlo for Bayesian inference

The same methods employed for SSMs can be applied to a different Bayesian setting, where we now have a collection of i.i.d. observations, $y_1, \ldots, y_n$, generated from a probability distribution with density, $f(y|\theta)$. The parameter θ represents the density model parameters and given the prior density for the model parameters, $\pi(\theta)$, the objective is to calculate the posterior density

$$\pi(\theta|y_{1:n}) \propto \pi(\theta) \prod_{i=1}^{n} f(y_i|\theta).$$

Usually $\pi(\theta|y_{1:n})$ is analytically intractable, but a common approach is to use Monte Carlo methods, such as MCMC, which only require the calculation of the product of the prior and the likelihood to estimate the posterior distribution and other summary statistics of interest. However, a weakness of MCMC is when we consider the target distribution to evolve as we receive new observations over time. Hence, we now have a sequence of target distributions, $\pi_t(\cdot),\ t = 1, \ldots, T$, varying over time, that we wish to approximate. This suffers from a growing computational cost whenever a new observation is received. We can use SMC methods to propagate the particles we have from one distribution to the next in a way that admits fixed computational complexity at each time step. This application of SMC methods is a particular focus of this thesis.

3.5.5 Sequential Monte Carlo framework

We introduce the general framework for SMC for an arbitrary set of target distributions and a set of proposal distributions that do not always need to have a density. Here, we closely follow the methodology and notation in Del Moral et al. (2006). In this section, we define the target distribution as

$$\pi_t(x_{1:t}) = \frac{\gamma_t(x_{1:t})}{Z_t},$$

where γ_t is known pointwise, but the normalisation constant Z_t might be unknown. The goal is to be able to sample the target distributions sequentially. If we were to use SIS, then we would sample from an importance distribution $q_t(x_{1:t})$ and the associated unnormalised weights would be computed recursively using the decomposition

$$\begin{aligned} w_t(x_{1:t}) &= \frac{\gamma_t(x_{1:t})}{q_t(x_{1:t})} \\ &= \frac{\gamma_{t-1}(x_{1:t-1})}{q_{t-1}(x_{1:t-1})} \frac{\gamma_t(x_{1:t})}{\gamma_{t-1}(x_{1:t-1}) q_t(x_t|x_{1:t-1})} \\ &= w_{t-1}(x_{1:t-1}) \frac{\gamma_t(x_{1:t})}{\gamma_{t-1}(x_{1:t-1}) q_t(x_t|x_{1:t-1})}. \end{aligned}$$

If the density $q_t(x_t|x_{1:t-1})$ does not have a closed form solution, then we cannot calculate the weights at this step, as we would have to account for every possible path that each particle could take up to time $t-1$. The solution is to introduce auxiliary variables. Therefore, instead of approximating $\pi_t(x_t)$, which depends on every possible particle path, we use $\tilde{\pi}_t(x_t)$ which depends on the particle path we have generated.

Introducing backward kernels

For any general γ_t, we want to be able to sample from some approximate $\tilde{\gamma}_t$ that admits γ_t as a marginal distribution, so the sequence of joint target distributions can be defined as

$$\tilde{\pi}_t(x_{1:t}) = \frac{\tilde{\gamma}_t(x_{1:t})}{Z_t}.$$

Let $\{L_k\}_{k=1}^{t-1}$ represent a sequence of artificial backward Markov kernels such that each $L_t(x_{t+1}, x_t)$ is the probability density of moving from x_{t+1} to x_t. These kernels can be defined arbitrarily, and we can redefine the joint distribution of the unnormalised likelihood as the recursive decomposition,

$$\begin{aligned}
\tilde{\gamma}_t(x_{1:t}) &= \gamma_t(x_t) \prod_{k=1}^{t-1} L_k(x_{k+1}, x_k) \\
&= \frac{\gamma_t(x_t)}{\gamma_{t-1}(x_{t-1})} L_{t-1}(x_t, x_{t-1}) \; \gamma_{t-1}(x_{t-1}) \prod_{k=1}^{t-2} L_k(x_{k+1}, x_k) \\
&= \frac{\gamma_t(x_t)}{\gamma_{t-1}(x_{t-1})} L_{t-1}(x_t, x_{t-1}) \; \tilde{\gamma}_{t-1}(x_{1:t-1}).
\end{aligned}$$

Within this framework, we can summarise a generic SMC sampler. In each iteration, we begin with a set of weighted particles $\{x_{1:t-1}^{(i)}, W_{t-1}^{(i)}\}_{i=1}^{N}$. At the next iteration, t, we extend the path of each particle with some forward Markov kernel, denoted $K_t(x_{t-1}, x_t)$, to obtain $\{x_{1:t}^{(i)}\}_{i=1}^{N}$. The discrepancy between the proposal distribution $q_t(x_{1:t})$, now defined as

$$q_t(x_{1:t}) = q_1(x_1) \prod_{k=2}^{t} K_k(x_{k-1}, x_k),$$

and the estimated joint target distribution $\tilde{\pi}_t(x_{1:t})$ is calculated by importance sampling. The result is a new approximation of the unnormalised particle weights,

$$w_t = \frac{\tilde{\gamma}(x_{1:t})}{q_t(x_{1:t})} = W_{t-1}(x_{1:t-1})\tilde{w}_t(x_{t-1}, x_t)$$

where the unnormalised incremental weights are

$$\tilde{w}_t(x_{t-1}, x_t) = \frac{\gamma_t(x_t)L_{t-1}(x_t, x_{t-1})}{\gamma_{t-1}(x_{t-1})K_t(x_{t-1}, x_t)}.$$

After the reweighting step, we normalise the particle weights.

Since the discrepancy between the target distribution $\tilde{\pi}_t$ and the proposal q_t increases with t, the variance of the unnormalised importance weights increases as well, leading to a degeneracy of the particle approximation. This can be mitigated using resampling based on some specified decision criterion and resetting the particle weights to be equally weighted.

Optimal and sub-optimal backward kernels

The backward kernel is arbitrary; however, it should be selected with respect to the forward kernels to optimise the performance of the SMC sampler. Proposition 1 of Del Moral et al. (2006) shows that the sequence of backward kernels $\{L_k\}_{k=1}^{t}$, which minimises the variance of the unnormalised particle weights $w_t(x_{1:t})$, is given by

$$L_{k-1}(x_k, x_{k-1}) = \frac{q_{k-1}(x_{k-1})K_k(x_{k-1}, x_k)}{q_k(x_k)}, \tag{3.5.2}$$

and in this case, the weights are

$$w_t(x_{1:t}) = \frac{\gamma_t(x_t)}{q_t(x_t)},$$

where $\gamma_t(x_t)$ is the target marginal distribution at time t. It is typically impossible to use the sequence of optimal kernels in practice because the kernels rely on marginal distributions that do not admit any closed-form expression.

Instead, a suboptimal alternative is to substitute π_{t-1} for q_{t-1} in (3.5.2) as an approximation to the optimal kernel, that is,

$$L_{t-1}(x_t, x_{t-1}) = \frac{\pi_{t-1}(x_{t-1})K_t(x_{t-1}, x_t)}{\int_{\mathcal{X}} \pi_{t-1}(x_{t-1})K_t(x_{t-1}, x_t)dx_{t-1}}, \tag{3.5.3}$$

which results in the unnormalised incremental weight update as

$$\tilde{w}_t(x_{t-1}, x_t) = \frac{\gamma_t(x_t)}{\int_{\mathcal{X}} \gamma_{t-1}(x_{t-1})K_t(x_{t-1}, x_t)dx_{t-1}}.$$

Del Moral et al. (2006) claim that the suboptimal kernel (3.5.3) is a good choice of kernel since γ_t is known analytically, whilst q_t is not.

Choosing backward kernel when using a forward MCMC kernel

If we select K_t to be an MCMC kernel that is invariant to π_t to act as a transition kernel, then we can use an alternative approximation to (3.5.3). Substituting π_t for

π_{t-1}, equation (3.5.3) becomes

$$\begin{aligned} L_{t-1}(x_t, x_{t-1}) &= \frac{\pi_t(x_{t-1})K_t(x_{t-1}, x_t)}{\int \pi_t(x_{t-1})K_t(x_{t-1}, x_t)dx_{t-1}} \\ &= \frac{\pi_t(x_{t-1})K_t(x_{t-1}, x_t)}{\pi_t(x_t)}, \end{aligned}$$

which is a good approximation of the backward kernel, provided that the discrepancy between π_{t-1} and π_t is small. We see that when this backward kernel is used, the unnormalised incremental weights in iteration t now depend on the particles in iteration $t-1$; that is,

$$\tilde{w}_t(x_{t-1}, x_t) = \frac{\gamma_t(x_{t-1})}{\gamma_{t-1}(x_{t-1})}. \tag{3.5.4}$$

This calculation of the weights in (3.5.4) is easy to compute; however, if the discrepancy between π_{t-1} and π_t is not small, then the variance of the incremental weights will be large. Incorporating MCMC kernels has been explicitly proposed in Neal (2001); Fearnhead (2002); Chopin (2002); Storvik (2002). Berzuini and Gilks (2001) also discuss applying MCMC moves within SMC, typically after resampling, to introduce particle diversity as the particles will still be a representation of the target density. This is known as the resample-move SMC framework. We can apply the MCMC kernel as many times as we would like and there is no need to specify a burn-in period, as would be the case in a standard MCMC scheme. This is because before moving, the set of equally weighted particles can be viewed as an approximated sample of the target density, π_t, which is preserved regardless of the number of times we apply the transition kernel. A graphical representation of the SMC sampler can be found in

Figure 3.5.2 and the corresponding pseudocode can be found in Algorithm 8.

In this thesis, we apply the resample-move framework of Berzuini and Gilks (2001) to the Mallows model to study the cases where we receive new full rankings for a single model (Chapter 4), new partial rankings and updated partial rankings for existing assessors in a single model (Chapters 5 and 6), and new full rankings for mixture model (Chapter 7). The resample-move pseudocode is presented in Algorithm 9.

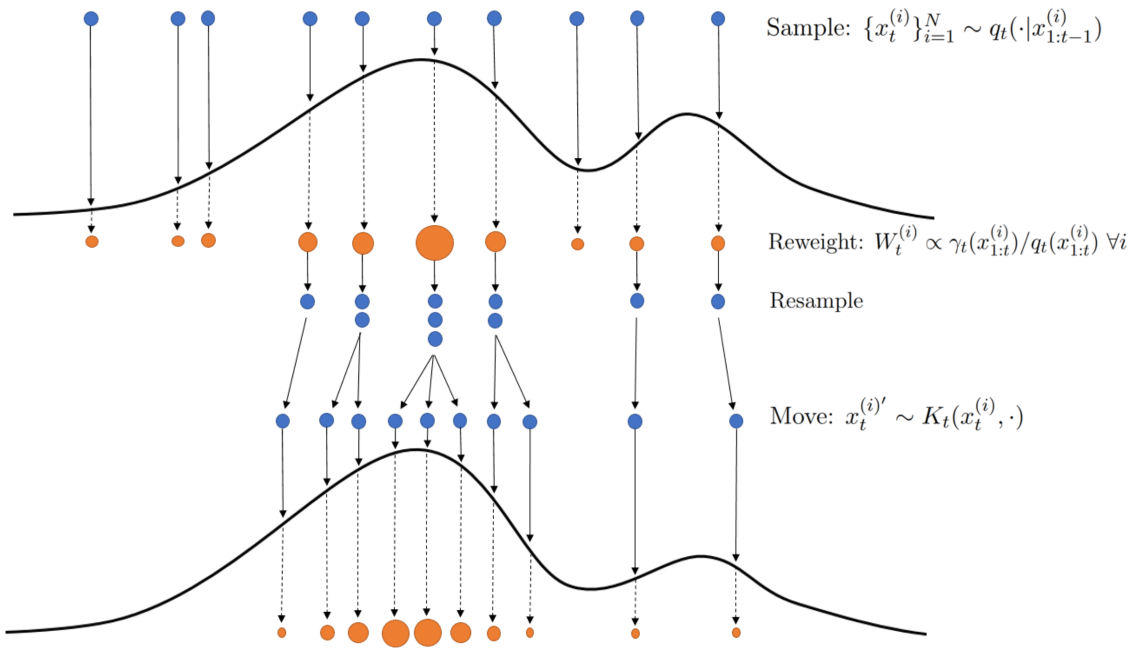

Sample: $\{x_t^{(i)}\}_{i=1}^N \sim q_t(\cdot|x_{1:t-1}^{(i)})$
Reweight: $W_t^{(i)} \propto \gamma_t(x_{1:t}^{(i)})/q_t(x_{1:t}^{(i)})\ \forall i$
Resample
Move: $x_t^{(i)\prime} \sim K_t(x_t^{(i)}, \cdot)$

Figure 3.5.2: A graphical representation of a general SMC sampler adapted from Doucet et al. (2001). Each particle is propagated through a series of importance sampling, resampling and move stages.

Algorithm 8: Sequential Monte Carlo

```
Input: $\pi = \gamma/Z,\ q_{1:T},\ N$.
Output: Monte Carlo estimate of $\hat{\pi}(\cdot)$.
for $t = 1$ do
    Sample particles, $\{x_1^{(i)}\}_{i=1}^N \sim q_1(\cdot)$.
    Calculate importance weights, $w_1^{(i)} = \gamma_1(x_1^{(i)})/q_1(x_1^{(i)})\ \forall i$.
    Normalise the importance weights $W_1^{(i)} = w_1^{(i)}/\sum_{i=1}^N w_1^{(i)}\ \forall i$.
    if Resampling condition is met then
        Resample particles to obtain $\{x_1^{(i)}, W_1^{(i)} = 1/N\}_{i=1}^N$.
        Sample $x_1^{(i)\prime} \sim K_1(x_1^{(i)}, \cdot)$.
    end
end
for $t = 2, \ldots, T$ do
    Sample particles, $\{x_t^{(i)}\}_{i=1}^N \sim q_t(\cdot|x_{1:t-1}^{(i)})$.
    Calculate importance weights, $w_t^{(i)} = W_{t-1}^{(i)} \frac{\gamma_t(x_{1:t}^{(i)})}{\gamma_{t-1}(x_{1:t-1}^{(i)}) q_t(x_t^{(i)}|x_{1:t-1}^{(i)})}\ \forall i$.
    Normalise the importance weights $W_t^{(i)} = w_t^{(i)}/\sum_{i=1}^N w_t^{(i)}\ \forall i$.
    if Resampling condition is met then
        Resample particles to obtain $\{x_{1:t}^{(i)}, W_t^{(i)} = 1/N\}_{i=1}^N$.
        Sample $x_t^{(i)\prime} \sim K_t(x_t^{(i)}, \cdot)$.
    end
end
```

Algorithm 9: Resample-move

```
Input: $\pi = \gamma/Z,\ q_{1:T},\ K,\ N$.
Output: Monte Carlo estimate of $\hat{\pi}(\cdot)$.
for $t = 1$ do
    Sample particles, $\{x_1^{(i)}\}_{i=1}^N \sim q_1(\cdot)$.
    Calculate importance weights, $w_1^{(i)} = \gamma_1(x_1^{(i)})/q_1(x_1^{(i)})\ \forall i$.
    Normalise the importance weights $W_1^{(i)} = w_1^{(i)}/\sum_{i=1}^N w_1^{(i)}\ \forall i$.
    Resample particles to obtain $\{x_1^{(i)}, W_1^{(i)} = 1/N\}_{i=1}^N$.
    for $k = 1, \dots, K$ do
        Sample $x_1^{(i)'} \sim K_1(x_1^{(i)}, \cdot)\ \forall i$.
    end
end
for $t = 2, \dots, T$ do
    Sample particles, $\{x_t^{(i)}\}_{i=1}^N \sim q_t(\cdot|x_{1:t-1}^{(i)})$.
    Calculate importance weights, $w_t^{(i)} = W_{t-1}^{(i)} \frac{\gamma_t(x_{1:t}^{(i)})}{\gamma_{t-1}(x_{1:t-1}^{(i)})q_t(x_t^{(i)}|x_{1:t-1}^{(i)})}\ \forall i$.
    Normalise the importance weights $W_t^{(i)} = w_t^{(i)}/\sum_{i=1}^N w_t^{(i)}\ \forall i$.
    Resample particles to obtain $\{x_{1:t}^{(i)}, W_t^{(i)} = 1/N\}_{i=1}^N$.
    for $k = 1, \dots, K$ do
        Sample $x_t^{(i)'} \sim K_t(x_t^{(i)}, \cdot)\ \forall i$.
    end
end
```

3.6 Clustering

In some situations, we may find that a collection of observations, $y_1, \dots, y_n$, is not generated from a single model but from a multi-model, often referred to as a mixture model. This means that each observation $y_i,\ i = 1, \dots, n$ is sampled from one of the C components in the mixture model. We assign a latent variable $z_i \in \{1, \dots, C\}$ to each observation to indicate which component of the mixture they belong to. The probability density function of the mixture model can be expressed as

$$\pi(y_{1:n}|\theta_{1:C}, z_{1:n}) = \prod_{i=1}^{n} \sum_{c=1}^{C} \tau_c \; p(y_i|z_i = c, \theta_c),$$

where $p(y_i|z_i = c, \theta_c)$ is the mixture component and $\tau_c,\ c = 1 \ldots, C$ is the component weight representing the probability that an observation belongs to the c^{th} mixture component. The proportions of the mixture are non-negative such that $\sum_{c=1}^{C} \tau_c = 1$. We assume that each component follows the same distributional form $f(y_i|\theta_c)$. Typically, it is assumed that the number of components, C, is known. However, in the case that this is unknown, we note that several methods exist for determining C. These are discussed in McLachlan et al. (2019).

Standard frequentist and Bayesian approaches can be applied to the clustering problem. The EM algorithm (Dempster et al., 1977) can find the MLE of each component of the mixture by finding the maximum of the complete log-likelihood function given the expected values of the latent variables, including the cluster assignment for each observation. In the Bayesian framework for clustering, several priors are specified for the latent variables, and we are interested in estimating the posterior density of the mixture model. This involves not only estimating the model parameters for each component $\theta_{1:C}$ but also the weights of the components $\tau_{1:C}$ and the cluster assignments for each observation $z_{1:n}$. We can use MCMC methods such as the Gibbs sampler (Geman and Geman, 1984), which performs augmentation the observable variable of the cluster assignments for each observation. These approaches have been applied to clustering with the Mallows model (see Chapter 2 for more details). Alternative

Bayesian approaches for clustering have been studied, such as variational Bayes (Blei and Jordan, 2006), which can handle high-dimensional mixture models and large data sets well but lacks an exact approximation of the mixture distribution.

3.6.1 Dirichlet process mixture model

It is also possible to construct a mixture without knowing the number of components to begin with. These can be constructed using Dirichlet process mixture (DPM) models (Antoniak, 1974). We assume that each y_i from a set of observations, $y_1, \ldots, y_n$, is drawn independently from one of the components of the mixture, $f(y_i|\theta_c)$, where θ_c is the unknown cluster parameters. The cluster parameters are independently drawn from the prior, $\pi(\theta)$, and a Dirichlet prior is also elicited for the cluster assignments $\pi(z_{1:n})$. Hence, the posterior density function for the mixture model is defined as

$$\pi(z_{1:n}, \theta_{1:c}|y_{1:n}) \propto \pi(z_{1:n}) \prod_{j=1}^{c} \pi(\theta_j) \prod_{i=1}^{n} f(y_i|\theta_{z_i}).$$

The Dirichlet prior for the cluster assignments $z_{1:n}$ is defined by the following recursion,

$$p(z_{i+1} = j|z_{1:i}) = \begin{cases} n_j/(i + \alpha_{DPM}) & \text{for } j = 1, \ldots, c_i \\ \alpha/(i + \alpha_{DPM}) & \text{for } j = c_i + 1 \end{cases},$$

where: n_j is the number of observations that $z_{1:i}$ assigns to component j; k_i is the number of components in the assignment $z_{1:i}$, and α_{DPM} is the concentration parameter in the DPM which controls the number of clusters.

3.6.2 Sequential Monte Carlo for mixture models

In Chapter 7, we consider SMC methods to fit a Mallows mixture model with full rankings. Mixture models have been studied in the SMC framework (Chopin, 2002; Del Moral et al., 2006) for finite mixture models. However, there has been a focus on applying SMC methods to DPM models. These were first developed by Liu (1996) and MacEachern et al. (1999) where observations are allocated to unobserved clusters in the particles in SIS. This was extended by Fearnhead (2004) to consider particle filters, which specified Dirichlet conjugate priors such that many parameters in the model can be integrated out to perform efficient estimation. Ulker et al. (2010) elaborates on Fearnhead (2004) and shows how the DPM model is presented under the general SMC sampler framework of Del Moral et al. (2006). A summary on how SMC methods can be applied to a variety of mixture models can be found in Carvalho et al. (2010).

3.6.3 Label switching

When applying Bayesian methods, such as MCMC, to explore the posterior of the mixture model, the permutation labels for the components can change multiple times between iterations as a result of the algorithm exploring one mode in the posterior to the next. This is known as label switching (Redner and Walker, 1984).

The challenge with label switching is that the likelihood,

$$p(y_{1:n}|\theta_{1:C}, z_{1:n}) = \prod_{i=1}^{n} \sum_{c=1}^{C} \tau_c \, p(y_i|z_i = c, \theta_c),$$

is invariant to the set $\mathcal{P}_C$ which contains $C!$ permutations to order C component labels. When estimating the component-specific parameters of the model or calculating the marginal posterior distributions, the parameters are not identifiable because the marginal posterior distributions will be identical for each mixture component. This may happen because when analysing the data, each data point assigned to component 1, for example, may not have the same corresponding model parameters for that particular component labelled component 1 across all Monte Carlo samples. To resolve this, we need to relabel the components during post-processing so that the components are in the same order at every iteration. We summarise the standard strategies for handling label switching problems.

Relabelling strategies

Sperrin et al. (2010) provides a summary of the strategy categories that exist to resolve the label switching problem. Recently an R package for handling the label switching problem with MCMC outputs, named `label.switching` was created by Papastamoulis (2016) which lists several algorithms under the following three categories.

A simple approach would involve creating identifiability constraints on the component-specific parameters (Peel and MacLahlan, 2000), e.g., $\pi_1 < \pi_2$ for a mixture model made up of two components, such that there exists a unique permutation that satisfies the constraints for each MCMC iteration. However, if we have many modes or the component-specific parameters are multi-dimensional, then it can be difficult to correctly select the constraints since we do not know how to distinguish the components.

Deterministic relabelling algorithms select a particular permutation of the components at each iteration of the MCMC output that minimises the posterior expectation of some loss function (Stephens, 2000). The component permutations are chosen based on some characteristics of each iteration to measure the closeness of each pair of order permutations. It is recommended to run the algorithm from multiple starting points to guarantee convergence at the global maximum rather than at a local maximum.

Lastly, the probabilistic relabelling approach (first introduced by Jasra et al. (2005)) assumes that instead of being able to determine the permutation at each iteration, we assign a probability density over all possible permutations at each iteration. In other words, the authors acknowledge the uncertainty in their relabelling. Stephens (2000) show that this approach can be viewed as an application of the EM algorithm (Dempster et al., 1977), where the available data are the MCMC output and the missing data are component permutations at each MCMC iteration.

Chapter 4

Sequential Monte Carlo for the Mallows Model with Full Rankings

4.1 Introduction

In this chapter, we introduce an SMC algorithm for approximating the Bayesian Mallows model posterior sequentially when we observe full rankings on a discrete timeline to perform inference. No previous attempts to use SMC methods for the Mallows model have been made in the literature. A typical approach would be to rerun the MCMC algorithm for the Bayesian Mallow model, such as the one described in Vitelli et al. (2018), to update the estimate of the posterior distribution for every additional ranking or collection of rankings observed. This is a viable option, but it is computationally intensive if we frequently need to update the posterior estimate each time we receive new information in a high volume; a typical problem with MCMC is to determine when to stop sampling the Markov chain. This motivates the need to reduce the computational effort between runs of the MCMC scheme for the Bayesian Mallows model when we observe new rankings. We aim to update the posterior distribution sequentially each time we receive new information quickly otherwise the posterior may not model the information correctly as we perform inference. One way to reduce computational effort would be to propagate the samples we have from one distribution to the next in such a way that has a fixed computational complexity at each time step.

The structure of this chapter is as follows. In Section 4.2, we formulate the problem and introduce the proposed SMC algorithm for estimating the Mallows posterior sequentially given that we observe complete rankings in each time step. The behaviour of the proposed algorithm is explored in Section 4.3 using a range of simulation studies in Section 4.3.1 before being assessed using real data sets against the MCMC algorithm for the Bayesian Mallows model (Vitelli et al., 2018) in Section 4.3.2.

4.2 Proposed method

4.2.1 Problem outline

Suppose that we are interested in performing inference with the Bayesian Mallows model,

$$p(\boldsymbol{\rho}, \alpha | \mathbf{R}_1, \ldots, \mathbf{R}_M) \propto \frac{\pi(\boldsymbol{\rho})\pi(\alpha)}{[Z_n(\alpha)]^M} \exp\left\{ -\frac{\alpha}{n} \sum_{j=1}^{M} d(\mathbf{R}_j, \boldsymbol{\rho}) \right\},$$

where: M represents a fixed number of observed rankings; $\pi(\boldsymbol{\rho}) = (n!)^{-1} 1_{\mathcal{P}_n}(\boldsymbol{\rho})$ is the uniform prior for the consensus ranking with support $|\mathcal{P}_n|$; and $\pi(\alpha|\lambda) =$

$\lambda \exp\{-\lambda\alpha\} 1_{[0,\infty)}(\alpha)$ is the exponential prior distribution for the scale parameter, as specified by Vitelli et al. (2018). Then consider the situation where each new ranking, or a collection of rankings, is observed over a sequence of time steps $t = 1, \ldots, T$. We assume that each complete ranking $\mathbf{R}$, expressed by each assessor, is drawn from a single Mallows model. Let $\mathbf{R}_{1:M_t}$ represent the rankings observed up to time t and assume that they remain fixed through the discrete-time sequence. We also define the likelihood for M_t observations as $p(\mathbf{R}_{1:M_t}|\boldsymbol{\rho}, \alpha)$. The task is to estimate the posterior,

$$\pi_t(\boldsymbol{\rho}, \alpha|\mathbf{R}_{1:M_t}) \propto \pi(\boldsymbol{\rho})\pi(\alpha)p(\mathbf{R}_{1:M_t}|\boldsymbol{\rho}, \alpha),$$

sequentially each time we receive new observations.

4.2.2 Methodology

The proposed SMC algorithm for the Mallows model with complete rankings follows the resample-move scheme (Berzuini and Gilks, 2001) which incorporates MCMC moves with SMC after resampling. The methodology is as follows. The algorithm starts at time $t = 0$ and draws N particles, each containing a sample of $\boldsymbol{\rho}_t^{(i)}$ and $\alpha_t^{(i)}$ $i = 1, \ldots, N$, to form the particle set $\{\boldsymbol{\theta}_0^{(i)} = (\boldsymbol{\rho}_0^{(i)}, \alpha_0^{(i)})\}_{i=1}^N$. The set can be generated by one of two methods: If M_0 is empty, then the posterior is the product of the priors, so we draw the values for $\{\boldsymbol{\rho}_0^{(i)}\}_{i=1}^N$ and $\{\alpha_0^{(i)})\}_{i=1}^N$ using the specified prior distributions described in Section 4.2.1. If we have an initial set of rankings of size $M_0 > 0$ to define the initial posterior at time $t = 0$, then we can run an MCMC algorithm and take a thinned set of samples generated after the burn-in period to create the particle set.

The algorithm iterates through a discrete-time sequence, $t = 1, \dots, T$. In each iteration, we observe $(M_t - M_{t-1})$ new complete rankings, and so we need to calculate the updated weights to account for the contributions each new observation has to the current estimated posterior distribution. At the start of iteration $t-1$, we have the particle set and their associated normalised weights $\{\boldsymbol{\theta}_{t-1}^{(i)}, W_{t-1}^{(i)}\}_{i=1}^{N}$ which approximate the density $\pi_{t-1}(\boldsymbol{\rho}, \alpha | \mathbf{R}_{1:M_{t-1}})$. To estimate the next updated target distribution based on the new observed ranking, $\pi_t(\boldsymbol{\rho}, \alpha | \mathbf{R}_{1:M_t})$, we propagate the particles $\{\boldsymbol{\theta}_{t-1}^{(i)}\}_{i=1}^{N}$ using a proposal distribution $q_t(\cdot | \boldsymbol{\theta}_{t-1}^{(i)})$, and adjust the weights accordingly to reflect the current posterior density. In SMC theory, the unnormalised incremental weight for each particle at iteration t is

$$\tilde{w}_t(\boldsymbol{\theta}_t^{(i)}) = \frac{\pi_t(\boldsymbol{\theta}_t^{(i)})}{\pi_{t-1}(\boldsymbol{\theta}_{t-1}^{(i)}) q_t(\boldsymbol{\theta}_t^{(i)} | \boldsymbol{\theta}_{t-1}^{(i)})}, \tag{4.2.1}$$

where $\pi_t(\cdot)$ represents some general target density at time t. In this scenario, the proposal distribution depends on the collection of new observed rankings at time t, but since we have no latent information to consider and the particles not moving at this stage of the proposed algorithm, we find $q_t(\boldsymbol{\theta}_t^{(i)} | \boldsymbol{\theta}_{t-1}^{(i)}) = p_t(\boldsymbol{\theta}_t^{(i)} | \boldsymbol{\theta}_{t-1}^{(i)}, \mathbf{R}_{M_{t-1}+1:M_t}) = 1$. Therefore, the incremental weight update is the ratio of the new and current target distributions, and we can also rewrite (4.2.1) as

$$\begin{aligned}
\tilde{w}_t(\boldsymbol{\theta}_t^{(i)}) &= \frac{\pi_t(\boldsymbol{\theta}_t^{(i)})}{\pi_{t-1}(\boldsymbol{\theta}_{t-1}^{(i)})} \\
&= \frac{\pi(\boldsymbol{\theta}_t^{(i)})p(\mathbf{R}_{1:M_t}|\boldsymbol{\theta}_t^{(i)})/p(\mathbf{R}_{1:M_t})}{\pi(\boldsymbol{\theta}_{t-1}^{(i)})p(\mathbf{R}_{1:M_{t-1}}|\boldsymbol{\theta}_{t-1}^{(i)})/p(\mathbf{R}_{1:M_{t-1}})} \qquad (4.2.2) \\
&= \frac{p(\mathbf{R}_{M_{t-1}+1:M_t}|\boldsymbol{\theta}_t^{(i)})p(\mathbf{R}_{1:M_{t-1}})}{p(\mathbf{R}_{1:M_t})} \qquad (4.2.3) \\
&\propto p(\mathbf{R}_{M_{t-1}+1:M_t}|\boldsymbol{\theta}_t^{(i)}) \\
&= (Z_n(\alpha_{t-1}^{(i)}))^{-(M_t-M_{t-1})} \exp\left\{ -\frac{\alpha_{t-1}^{(i)}}{n} \sum_{j=M_{t-1}+1}^{M_t} d(\mathbf{R}_j, \boldsymbol{\rho}_{t-1}^{(i)}) \right\},
\end{aligned}$$

where $\alpha_{t-1}^{(i)}$ and $\boldsymbol{\rho}_{t-1}^{(i)},\ i = 1, \ldots, N$ are the current values of the parameter values of the Mallows posterior. We can simplify (4.2.2) to (4.2.3) because at this stage of the SMC algorithm $\boldsymbol{\theta}_t^{(i)} = \boldsymbol{\theta}_{t-1}^{(i)}$, so the priors for the model parameters cancel and we can focus on calculating the likelihood for the new observed rankings. The particle weights at time t are the product of the incremental weights and the particle weights from the previous time step $w_t^{(i)}(\boldsymbol{\theta}_t) = W_{t-1}^{(i)}(\boldsymbol{\theta}_{t-1})\tilde{w}_t^{(i)}(\boldsymbol{\theta}_t)$. These are then normalised to obtain $W_t(\boldsymbol{\theta}_t)$.

Next, the particles are resampled using multinomial resampling to discard the particles with negligible weight and replicate the heavier weighted particles. At this point, we will have an equally weighted particle set, denoted $\{\bar{\boldsymbol{\theta}}_t^{(i)}, W_t^{(i)} = \frac{1}{N}\}_{i=1}^N$.

Lastly, we explore the current posterior density using the Metropolis-Hastings MCMC move kernel $K_t(\bar{\boldsymbol{\theta}}_t^{(i)}, \boldsymbol{\theta}_t^{(i)})$ to perturb the particle values for each parameter. In particular, we use the proposal distributions for sampling values of $\boldsymbol{\rho}$ and α, as suggested by Vitelli et al. (2018), for the Bayesian Mallows model. We use the leap-

and-shift proposal distribution (Vitelli et al., 2018) to propose $\boldsymbol{\rho}_t^{'(i)}$ and the log-normal distribution to propose $\alpha_t^{'(i)}$. Further details on this proposal can be found in Section 2.5.3 of Chapter 2. In theory, we can apply the MCMC move kernel as many times as we wish as we are invariant to the current target distribution π_t. However, we should not be heavily reliant on this aspect of the algorithm; we do not want to exceed the number of iterations it would take to run an MCMC chain for each iteration of SMC, otherwise the computational cost will not be reduced.

At the end of each iteration, the proposed SMC algorithm for the Mallows model will generate an equally weighted set of N samples of $\{\boldsymbol{\rho}_t^{(i)}, \alpha_t^{(i)}\}_{i=1}^N$, which represents the posterior density given all observed data. The pseudocode for the SMC algorithm for the Mallows model with full rankings can be found in Algorithm 10.

Algorithm 10: Sequential Monte Carlo for the Mallows Model with Full Rankings

```
Input: $\lambda,\ \sigma_\alpha,\ d(\cdot,\cdot),\ K,\ L,\ N,\ M_1,\ldots,M_T;\ \mathbf{R}_1,\ldots,\mathbf{R}_{M_T};\ T,\ Z_n(\alpha)$.
Output: Posterior distributions of $\boldsymbol{\rho},\alpha$.
Initialisation of SMC: randomly generate $\boldsymbol{\rho}_0^{(i)},\alpha_0^{(i)}$ for $i=1,\ldots,N$.
for $t=1,\ldots,T$ do
    Observe $\mathbf{R}_{M_{t-1}+1:M_t}$.
    for $i=1:N$ do
        Compute
          $\tilde{w}_t^{(i)}=(Z_n(\alpha_{t-1}^{(i)}))^{-(M_t-M_{t-1})}\exp\Big\{-\frac{\alpha_{t-1}^{(i)}}{n}\sum_{j=M_{t-1}+1}^{M_t}d(\mathbf{R}_j,\boldsymbol{\rho}_{t-1}^{(i)})\Big\}$.
        Compute $w_t^{(i)}=W_{t-1}^{(i)}\tilde{w}_t^{(i)}$.
    end
    Normalise the weights $W_t^{(i)}=w_t^{(i)}/\sum_{i=1}^N w_t^{(i)}$.
    Resample $(k_1,\ldots,k_N)\sim\mathcal{M}(W_t^{(1)},\ldots,W_t^{(N)})$ and set
      $\{\bar{\boldsymbol{\theta}}_t^{(1:N)},W_t^{(i)}\}\leftarrow\{\boldsymbol{\theta}_t^{(k_1:k_N)},\frac{1}{N}\}$.
    for $i=1:N$ do
        for $k=1,\ldots,K$ do
            M-H step: update $\boldsymbol{\rho}_t^{(i)}$ using the leap-and-shift proposal
              $(\boldsymbol{\rho'}\sim \mathrm{L\&S}(\boldsymbol{\rho}_t^{(i)},L))$.
            M-H step: update $\alpha_t^{(i)}$ with log-normal proposal $(\alpha'\sim\log\mathcal{N}(\alpha_t^{(i)},\sigma_a^2))$.
        end
    end
end
```

4.3 Experimental analyses

4.3.1 Exploratory analysis

First, we are interested in observing the general performance of the proposed SMC framework when certain variables and parameters in the algorithm are altered. We begin by describing the general default experiment set-up before explaining the variables we test to observe their behaviour. The results of these experiments are presented in Figure 4.3.1.

We analyse the behaviours for a different number of items, $n \in \{10, 20, \ldots,\ 50\}$, in the synthetic data sets, each of which are generated using the function `sample_mallows` in the `BayesMallows` R package (Sørensen et al., 2020). In particular, we specify `rho_0 = c(1:n_items)` and `alpha_0 = 2` as the underlying parameters of the Mallows model to generate the data set.

With each data set, we run the SMC algorithm by propagating $N = 1,000$ particles over $T = 10$ time steps, each time introducing ten new rankings and performing a sequential update of the estimate of the posterior given the new observed rankings. As a default, we specify ten applications of the MCMC move kernels for each model parameter for each particle after resampling. In total, we apply 100,000 MCMC kernels in total for each parameter in the particle set. We set the leap-size parameter in the leap-and-shift proposal for $\boldsymbol{\rho}$ to be $L = \lfloor n/5 \rfloor$, and we set the hyperparameter and the standard deviation for the exponential proposal distribution for α to be $\lambda = 0.1$ and $\sigma_\alpha = 0.5$ respectively. We use the footrule distance when measuring the distances between rankings throughout this chapter.

To measure the spread of the sampled values of $\boldsymbol{\rho}$ in the MCMC chain or particle set from the "true" parameter, we use the posterior expected mean squared error (MSE), defined in Liu et al. (2019a) as

$$\texttt{mse} = \mathbb{E}\Big(\frac{1}{n}||\boldsymbol{\rho} - \boldsymbol{\rho}^*||\ |\mathbf{R}_1, \ldots, \mathbf{R}_M\Big) = \frac{1}{n}\sum_{i=1}^{n}\sum_{r=1}^{n}(r - \rho_i^*)^2 p(\rho_i = r|\mathbf{R}_1, \ldots, \mathbf{R}_M),$$

since the shared consensus ranking is the parameter we are most interested in estimating. The MSE is an indicator of how close the sampler approximated the distribution that was used to generate the data set if we know the true consensus ranking and the scale parameter, denoted here as $\boldsymbol{\rho}^*$ and α^*, respectively. The sample spread may not be close to $\boldsymbol{\rho}^*$, but we expect the MSE to be the same for different samplers of the same posterior distribution given the data. We ran ten repetitions of the algorithm for each data set and calculated the 95% confidence interval of the MSE values.

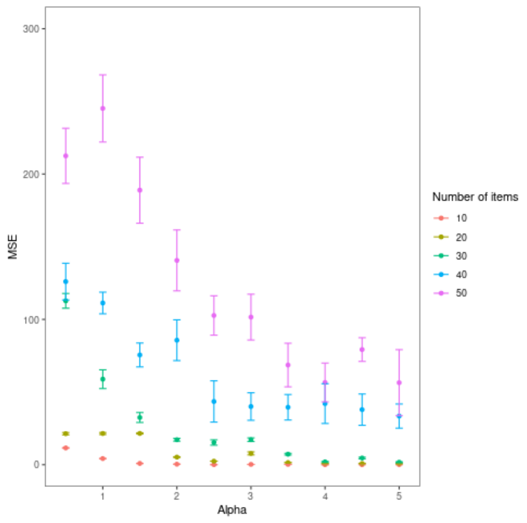
We first look at allowing the scale parameter to vary. Each data set that we create is generated by considering a different value of $\alpha \in \{0.5, 1, \ldots, 4.5, 5\}$ to change the level of variability in rankings around the consensus ranking. The results of this experiment are provided in Figure 4.3.1a. The behaviour observed is based on what we would expect: a high value of α would generate a data set with rankings that are close to the consensus ranking in distance, so the MSE would likely be smaller than if we estimated the MSE of a data set with a low value of α. This behaviour is clearly shown when we used data sets containing rankings for a larger number of items as α increases from 0.5 to 2.5.

Next, we assess the SMC algorithm with a different number of particles, $N \in \{500, 1000, 2500, 5000, 7500, 10000\}$, for each data set in Figure 4.3.1b. Given the total number of rankings, we find that a significantly larger number of particles does not affect the MSE value as we would expect. This also means that we can reduce the computational effort if we want to limit the total computation.

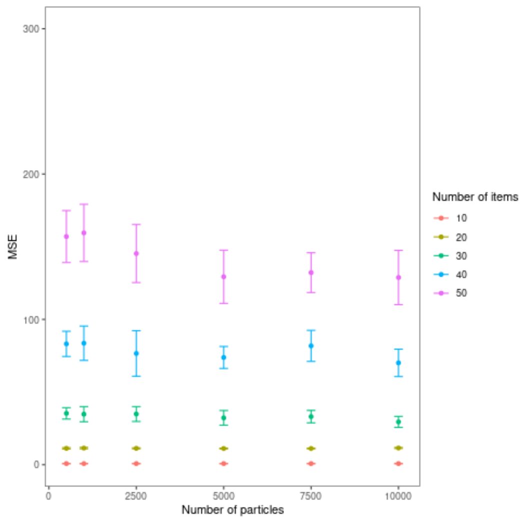
In Figure 4.3.1c, we inspect the number of applications of the MCMC move kernels $\{1, 2, 5, 10, 20\}$ to see if there is an ideal range that provides a sufficient exploration of the target posterior density at each time step in SMC. Each iteration of the proposed SMC algorithm has a resampling step, so we need to apply some move steps to replenish the diversity lost in resampling. Ideally, we do not want the total number of

MCMC move kernels applied across the particles in one SMC iteration to exceed the number of iterations to run a new MCMC chain each time we observe new rankings otherwise the proposed SMC methodology is not computationally effective compared to MCMC methods. The MSE decreases significantly when we increase the number of MCMC moves from 1 to 10, then it decreases gradually as the number of moves increases even more. This implies that incorporating some move kernels helps the spread of the particle values to move closer to the "true" parameter.

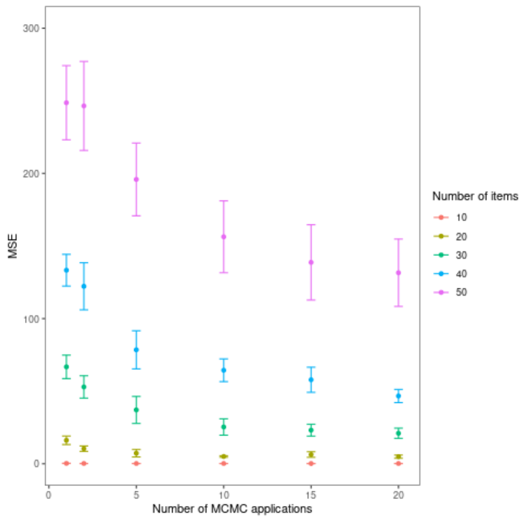
Then, we consider how the number of new observations and the number of MCMC moves, in each iteration affects the MSE estimation if we have a fixed amount of computational effort allowed. For this experiment, we generate several data sets containing 1,000 rankings for various numbers of items and consider the number of new rankings in each iteration to be 5, 10, 20, 25, 50 and 100. This means that there will be 200, 100, 50, 40, 20 and 10 times steps respectively for each case we consider. To keep an overall fixed computational cost for MCMC iterations we use 1, 2, 4, 5, 10, and 20 MCMC kernel moves in each iteration of each experiment. We see in Figure 4.3.1d, that for small numbers of items in the rankings, we do not see any major difference in the MSE. However, for larger numbers of items, if we increase the number of observations too much then we observe a lower value. This behaviour does match with the SMC theory that assumes that so long as each sequential posterior does not vary too much from the previous time step, then we should be able to update the posterior estimate accordingly. A large number of new rankings would risk changing the posterior distribution significantly and the particles will risk not being able to estimate the evolving posterior.



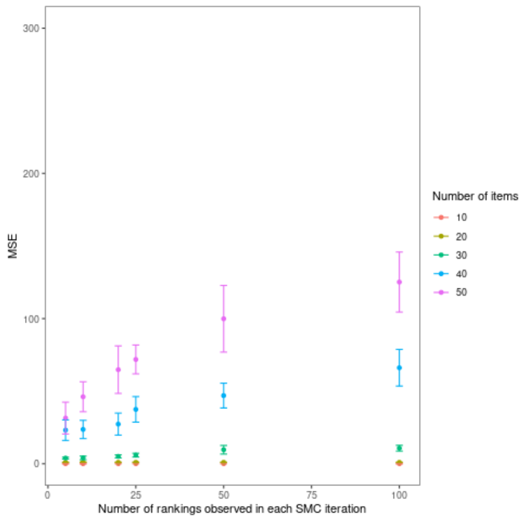
(a) Scale parameter value.



(b) Number of particles.



(c) MCMC move kernel applications.



(d) Number of rankings observed.

Figure 4.3.1: Summary plots containing the 95% confidence intervals of the MSE for ten runs of each experiment for each synthetic data set containing $n = \{10, 20, 30, 40, 50\}$ items. The parameter or variable investigated is included in the captions.

4.3.2 Real data experiments

Potato data

We illustrate the performance of the SMC method against the MCMC method for the Bayesian Mallows model with an example data set from the `BayesMallows` R package (Sørensen et al., 2020) called `potato_visual`. This data set has been studied in Liu et al. (2019a) and contains $M = 12$ peoples' assessments of the weights of $n = 20$ potatoes that were obtained by visually inspecting them. A true ranking was also obtained by weighing the items to give `potato_true_ranking`.

In both methods, we set the exponential hyperparameter and the standard deviation for the proposal distribution for α to be $\lambda = 0.1$ and $\sigma_\alpha = 0.15$ respectively. We ran the MCMC algorithm of Vitelli et al. (2018) for 10,000 iterations with the complete data set as the observed data and discarded the first 5,000 as burn-in. In the SMC algorithm, we propagated $N = 1000$ particles and performed sequential model parameter updates each time we received a new ranking in the data set for $T = 12$ time steps. At each SMC iteration, we performed five applications of the MCMC move kernels for $\boldsymbol{\rho}$ and α for each particle. This experiment is referred to as SMC-1 in the analysis provided in Figure 4.3.2 and Tables 4.3.1-4.3.2. Specifically, we set $L = 2$ in the leap-and-shift proposal for $\boldsymbol{\rho}$.

We use several estimates to assess the performance of each sampler. We can use the output of the MCMC and SMC algorithms to calculate the marginal posterior distributions of $\boldsymbol{\rho}$ and α. Due to the nature of $\boldsymbol{\rho}$ being represented as a vector, we use a heat plot to illustrate the marginal posterior probabilities of ranking each item in $\mathcal{A}$

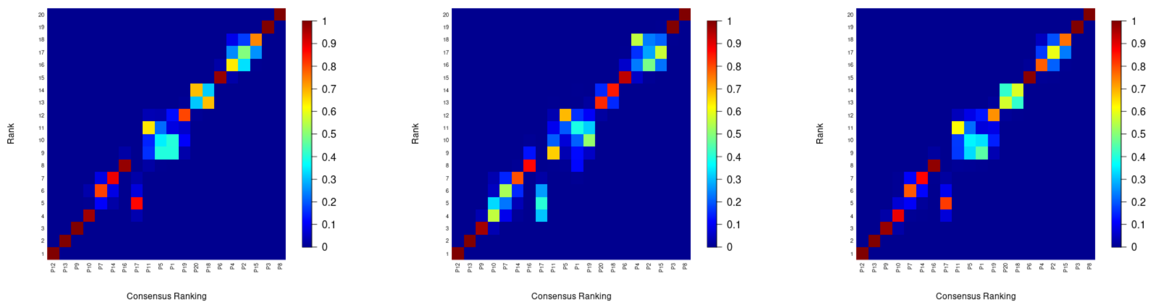
as rank $i \in \{1, 2, \ldots, n\}$ in $\boldsymbol{\rho}$. We can also report the posterior density of $\boldsymbol{\rho}$ as a single point estimate of a data set according to the maximum a posteriori (MAP) estimate. Alternatively, we can report the cumulative probability (CP) consensus ranking. This is determined by the procedure, as described in Vitelli et al. (2018): select the item which has the maximum a posteriori marginal probability of being ranked 1st, then the item which has the maximum a posteriori marginal posterior probability of being ranked 1st or 2nd among the remaining ones, etc. The MSE is another single point estimate that can be used to assess how well the sampler approximates the distribution of the data. We can also report the cumulative probability of each item to be ranked in shared consensus position, or higher, $p(\boldsymbol{\rho}_i \leq i)$ and the 95% highest posterior density interval (HPDI) for each item to show the posterior uncertainty. For the scale parameter, we report the 95% confidence interval estimate.

Figure 4.3.2 shows the marginal posterior distribution of $\boldsymbol{\rho}$ and α with both methods. In Figures 4.3.2a-4.3.2b, the x-axis of the heat plots has the items are ordered according to their true ranking, $\boldsymbol{\rho}^* =$ `potato_true_ranking`, representing the potatoes in descending order of weights. The heat plot obtained by SMC appears to have similar posterior probabilities, which have high accuracy, as the MCMC heat plot for the heaviest items (P12, P13, P9) and the lightest items (P3 and P8) whilst there is some variation between the two methods, in terms of accuracy, for the middle-range items. This is supported by the estimation of the CP consensus and the 95% HDPIs in Table 4.3.1. However, item P17 is shown in both methods to have a high posterior probability of being ranked higher than its actual weight, but the MCMC algorithm is more certain that P17 is likely to be ranked 5 whereas the SMC method could assign

P17 a rank of either 4 or 5. The MCMC method does have a lower MSE value than SMC due to displaying more certainty for the items to be allocated their “true” rank.

At this point, we want to see if increasing the computational effort of the proposed SMC sampler will give a similar posterior approximation to MCMC. We increase the number of MCMC move kernels from five to ten in each iteration. This experiment is referred to as SMC-2 in Figure 4.3.2 and Tables 4.3.1-4.3.2. The marginal posterior distributions of $\boldsymbol{\rho}$ are shown in Figures 4.3.2c with the MSE and posterior estimates for α, and the additional posterior estimates of these experiments are given in Table 4.3.1. We see that an increase in the computational effort does obtain a posterior estimate similar to that of MCMC. We obtain similar values of the MSE and the estimated scale parameter in Figures 4.3.2a- 4.3.2c. The CP consensus for this SMC experiment does produce slightly different ranks for items in $\boldsymbol{\rho}$ compared to the CP consensus obtained by MCMC, except for the ranks for middle-ranked items being swapped in pairs. The posterior densities for α in Figures 4.3.2a and 4.3.2c also appear to give a similar estimate.

We also perform ten runs of each experiment and record the MSE and time taken to run each algorithm. We record the time taken to run the MCMC algorithm with the full data set of twelve rankings whereas we record the time taken to run the SMC algorithm for the complete twelve iterations. This is summarised in Table 4.3.2. We notice that the MCMC algorithm produces a consistent and more accurate estimate of the MSE with a narrow confidence interval compared to both SMC experiments. We expect the MSE values in MCMC and SMC to be the same and it is clear that the SMC-2 experiment achieves a closer MSE result to the MCMC. The MCMC and the first SMC experiment take an equal amount of time to process the data set whereas the second SMC experiment takes more than twice as long to run in time. This is expected because we have twice as many MCMC moves iterations to process. However, if we were to observe several more rankings, then it would take the MCMC algorithm at least 0.2 seconds to run the chain for 10,000 iterations with the existing and new rankings, whereas the two SMC experiments would take ∼0.02 and ∼0.05 seconds respectively to perform one iteration to update the particles with the new ranking and apply 5,000 and 10,000 MCMC move iterations across 1,000 particles. In this experiment, we see that the proposed SMC methodology can perform sequential posterior updates in less computational time compared to running a new MCMC chain for the same amount of computational effort in terms of the number of MCMC iterations applied.



(a) MCMC: MSE = 1.07, $\hat{\alpha} = 10.76\ (9.55, 12.34)$.

(b) SMC-1: MSE = 1.39, $\hat{\alpha} = 10.44\ (9.96, 11.88)$.

(c) SMC-2: MSE = 1.07, $\hat{\alpha} = 10.78\ (9.34, 12.22)$.

Figure 4.3.2: Heat plots of the posterior probabilities, for $n = 12$ potatoes, for being ranked as the k^{th} most preferred, for $k = 1, ..., 12$ using the MCMC algorithm (left), the SMC algorithm with five MCMC move kernel steps in each time step (middle), and the SMC algorithm with ten MCMC move kernel steps in each time step (right). On the x-axis, the items are ordered according to their true consensus. The MSE value and the mean estimate of α with its corresponding 95% HDPI are included in the captions.

	CP			$p(\boldsymbol{\rho}_i \leq i)$			**95% HDPI**		
$\boldsymbol{\rho}$	**MCMC**	**SMC-1**	**SMC-2**	**MCMC**	**SMC-1**	**SMC-2**	**MCMC**	**SMC-1**	**SMC-2**
ρ_1	P12	P12	P12	1	1	1	[1]	[1]	[1]
ρ_2	P13	P13	P13	1	1	1	[2]	[2]	[2]
ρ_3	P9	P9	P9	1	0.97	0.98	[3]	[3]	[3]
ρ_4	P10	P10	P10	0.97	0.59	0.92	[4]	[3,6]	[4,5]
ρ_5	P17	P17	P17	0.90	0.71	0.89	[4,6]	[4,6]	[4,6]
ρ_6	P7	P7	P7	0.92	0.85	0.89	[5,7]	[4,7]	[5,7]
ρ_7	P14	P14	P14	1	0.99	1	[6,7]	[5,7]	[6,7]
ρ_8	P16	P16	P16	0.98	0.87	0.98	[8]	[8,9]	[8]
ρ_9	P5	P11	P1	0.40	0.67	0.46	[9,12]	[9,11]	[9,12]
ρ_{10}	P1	P19	P5	0.80	0.57	0.69	[9,11]	[8,12]	[10,12]
ρ_{11}	P11	P1	P11	0.96	0.84	0.96	[9,12]	[9,12]	[9,12]
ρ_{12}	P19	P5	P19	0.99	1	0.99	[10,12]	[10,12]	[10,12]
ρ_{13}	P18	P20	P20	0.68	0.85	0.58	[13,14]	[13,14]	[13,14]
ρ_{14}	P20	P18	P18	1	1	1	[13,14]	[13,14]	[13,14]
ρ_{15}	P6	P6	P6	0.98	0.94	0.99	[15]	[15,16]	[15]
ρ_{16}	P4	P2	P4	0.66	0.49	0.79	[16,18]	[16,18]	[16,18]
ρ_{17}	P2	P15	P2	0.83	0.79	0.81	[16,18]	[16,18]	[16,18]
ρ_{18}	P15	P4	P15	1	1	1	[17,18]	[15,18]	[17,18]
ρ_{19}	P3	P3	P3	1	1	1	[19]	[19]	[19]
ρ_{20}	P8	P8	P8	1	1	1	[20]	[20]	[20]

Table 4.3.1: Posterior estimates of the items in the Potato data set. Items are arranged according to the CP consensus ranking with their corresponding cumulative probabilities and 95% HDPIs.

	MCMC		**SMC-1**		**SMC-2**	
Method	Mean	95% CI	Mean	95% CI	Mean	95% CI
MSE	1.05	(1.03, 1.08)	1.45	(1.29, 1.61)	1.12	(1.05, 1.18)
Time (s)	0.21	(0.20, 0.22)	0.24	(0.23, 0.25)	0.52	(0.52, 0.52)

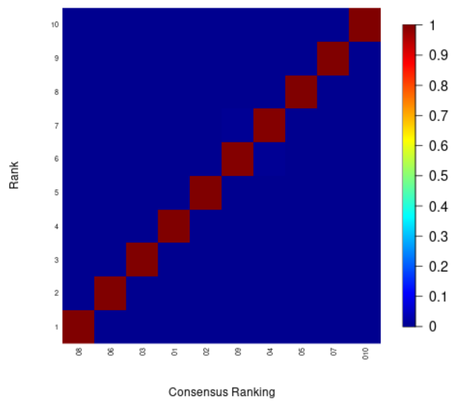
Table 4.3.2: Summary table of the MSE and time taken to perform ten experiment runs of the MCMC and SMC algorithms with the Potato data set.

Sushi data

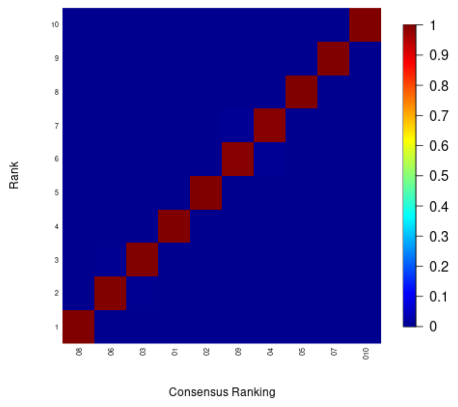
Next, we study the benchmark Sushi data set (Kamishima, 2003) that contains $M = 5000$ full rankings over $n = 10$ varieties of sushi indicating sushi preferences. This data set has typically been studied for the clustering problem (Lu and Boutilier, 2014; Vitelli et al., 2018) as it represents a sample of sushi preferences collected across Japan, and so there may be differences amongst the different regions of Japan and hence several different shared consensus rankings.

The experiment conditions are the same as those provided in Section 4.3.2 except that for the SMC experiments, given the number of rankings in the Sushi data set, we observe 100 rankings in each iteration for $T = 50$ iterations in total and the standard deviation for the scale parameter is set to $\sigma_\alpha = 0.5$. The posterior estimates of the Mallows model parameters are displayed in Figure 4.3.3. We find that the estimated shared consensus ranking is determined with a small number of iterations in the MCMC chain and the SMC method after a few time steps; since we have a large number of rankings for a small number of items, it is likely to achieve convergence quickly. The heat plots in Figures 4.3.3a-4.3.3b show that both methods estimate the shared consensus ranking with very high certainty: the MAP and CP consensus rankings are the same and the 95% HDPI intervals do not include other possible positions for each sushi item. We have also used the shared consensus ranking in Table 4.3.3 as the x-axis for the heat plots in Figures 4.3.3a-4.3.3b. The posterior estimate of the scale parameter obtained by SMC is comparable to that of the MCMC method.

We also ran ten repeated experiments to see if the two methods were consistent in the time taken to process the ranking data and can produce a consistent estimate of the shared consensus ranking for the sushi items. These results can be seen in Table 4.3.3. Notably, the SMC algorithm is consistent with its MSE value, implying that it achieves the same estimated shared consensus ranking, whereas the mean MSE obtained from the MCMC algorithm is 1.32. In three out of ten experiment runs, the MSE was 4.40, which implies that the MCMC algorithm may converge to an alternate estimate of $\boldsymbol{\rho}$. The SMC algorithm appears to process the Sushi data set in a fraction of the time compared to the MCMC algorithm. Again, similar to the discussion in Section 4.3.2, if we were to observe additional rankings, then the MCMC algorithm would need to process the existing rankings in the Sushi data set again along with the new rankings to create an MCMC chain 10,000 iterations long. With the SMC algorithm, though it would take several seconds longer than the MCMC algorithm to run, it would take half the computational effort in terms of the number of MCMC move kernel applications in one iteration of SMC to update the posterior estimates with the new rankings. In this experiment, the SMC algorithm is able to perform inference with less computational effort than MCMC.



(a) MCMC: MSE = 0.00, $\hat{\alpha}$ = 1.72 (1.68, 1.76).



(b) SMC: MSE = 0.00, $\hat{\alpha}$ = 1.72 (1.68, 1.76).

Figure 4.3.3: Heat plots of the posterior probabilities, for $n = 10$ sushi items, for being ranked as the k^{th} most preferred, for $k = 1, ..., 10$ using the MCMC algorithm (left) and the SMC algorithm (right). On the x-axis, the items are ordered according to their CP consensus. The MSE value and the mean estimate of α with its corresponding 95% HDPI are included in the captions

$\boldsymbol{\rho}$	ρ_1	ρ_2	ρ_3	ρ_4	ρ_5	ρ_6	ρ_7	ρ_8	ρ_9	ρ_{10}
Item	Fatty tuna	Salmon roe	Tuna	Shrimp	Sea eel	Tuna roll	Squid	Sea urchin	Egg	Cucumber roll

Table 4.3.3: The CP consensus ranking of the sushi items in the Sushi data set.

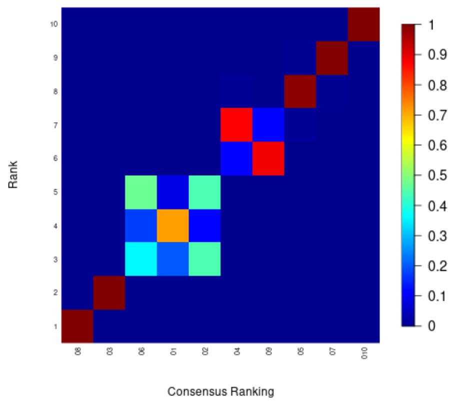
	MCMC		**SMC**	
Method	Mean	95% CI	Mean	95% CI
MSE	1.32	(0.00, 2.84)	0.00	(0.00, 0.00)
Time (s)	6.60	(6.55, 6.64)	98.59	(98.37, 98.82)

Table 4.3.4: Summary table of the MSE and time taken to perform ten experiment runs of the MCMC and SMC algorithm with the Sushi data set.

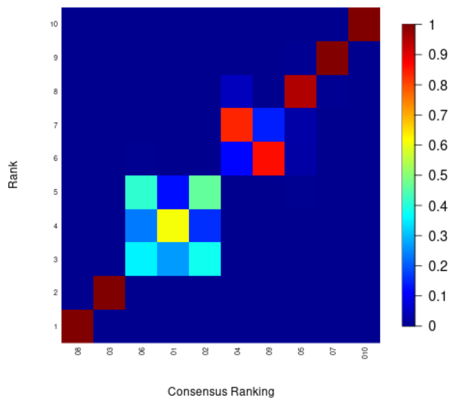
Reduced Sushi data

Based on the analysis of the full Sushi data set, we considered another experiment using the first $M = 250$ rankings of the data set. This is because we are interested to see how the SMC algorithm performs against the MCMC algorithm when having a reasonable number of rankings such that we can create some variability in the inference of the parameter estimates. We introduced 25 new rankings in each of the $T = 10$ iterations of the SMC algorithm and performed five MCMC move kernel iterations after resampling, whereas we ran the MCMC algorithm with the full data set of 250 rankings for 10,000 iterations and discarded the first 5,000 as burn-in.

Figure 4.3.4 presents the heat plots of the marginal posterior for the consensus ranking from the output of the MCMC and SMC algorithms, the MSE value and the posterior estimate of α. It is clear in both methods that there is more uncertainty in the sushi items ranked 3-7. This suggests that in the data set many assessors will have been similar in their choice of items they ranked top and bottom, leading to higher variability for middle-ranked items. The MSE and the posterior estimates for the scale parameter are close to matching. We found that the estimated CP consensus ranking and the corresponding 95% HDPI intervals were also the same between both methods in Table 4.3.5. We performed ten repetitions of each experiment and recorded the MSE and run time. The results are summarised in Table 4.3.6 and we see that the SMC method on average obtains a slightly higher MSE than MCMC. It takes on average 0.45 seconds to run the MCMC algorithm for 10,000 iterations, whilst it takes 1.15 seconds to perform ten time steps of the SMC algorithm, where each iteration performs 5,000 MCMC moves across 1,000 particles. If we were to observe another 25 complete rankings from the original Sushi data set, then an additional iteration of SMC would take ∼0.12 seconds to run with 5,000 applications of MCMC move kernels, whereas the MCMC algorithm would take at least 0.45 seconds to perform 10,000 MCMC iterations. In this instance, we see that we can save time and computational cost by using SMC to perform sequential inference.



(a) MCMC: MSE = 0.60, $\hat{\alpha}$ = 1.75 (1.59, 1.92).



(b) SMC: MSE = 0.60, $\hat{\alpha}$ = 1.75 (1.58, 1.92).

Figure 4.3.4: Heat plots of the posterior probabilities, for $n = 10$ sushi items in the reduced Sushi data set, for being ranked as the k^{th} most preferred, for $k = 1, ..., 10$ using the MCMC algorithm (left) and the SMC algorithm (right). On the x-axis, the items are ordered according to their CP consensus. The MSE value and the mean estimate of α with its corresponding 95% HDPI are included in the captions.

		$p(\boldsymbol{\rho}_i \leq i)$		**95% HDPI**	
$\boldsymbol{\rho}$	**CP**	**MCMC**	**SMC**	**MCMC**	**SMC**
ρ_1	Fatty tuna	1.00	1.00	[1]	[1]
ρ_2	Tuna	1.00	1.00	[2]	[2]
ρ_3	Sea eel	0.44	0.38	[3,5]	[3,5]
ρ_4	Shrimp	0.91	0.88	[3,5]	[3,5]
ρ_5	Salmon roe	1.00	1.00	[3,5]	[3,5]
ρ_6	Tuna roll	0.88	0.86	[6,7]	[6,7]
ρ_7	Squid	0.99	0.95	[6,7]	[6,7]
ρ_8	Sea urchin	1.00	1.00	[8]	[8]
ρ_9	Egg	1.00	1.00	[9]	[9]
ρ_{10}	Cucumber roll	1.00	1.00	[10]	[10]

Table 4.3.5: Posterior estimates of the items in the reduced Sushi data set. Items are arranged according to the CP consensus ranking with their corresponding cumulative probabilities and 95% HDPIs.

	MCMC		**SMC**	
Method	Mean	95% CI	Mean	95% CI
MSE	0.61	(0.58, 0.65)	0.63	(0.59, 0.67)
Time (s)	0.45	(0.44, 0.45)	1.25	(1.24, 1.26)

Table 4.3.6: Summary table of the MSE and time taken to perform ten experiment runs of the MCMC algorithm and SMC algorithm with the reduced Sushi data set.

4.4 Conclusion

In this chapter, we have developed an SMC algorithm to update the estimated posterior distribution of a single Mallows model in an online setting where we receive a collection of complete rankings over a discrete-time sequence. This has not been previously explored in the Mallows literature. SMC can be utilised as an alternative to MCMC to estimate the posterior distribution sequentially each time we receive a new full ranking, or a batch of new rankings, to reduce the computational effort. MCMC can still be used for sequential inference, but for every observational update of rankings, we would need to re-run the MCMC chain with all of the existing and new full rankings to estimate the new posterior. This can be computationally demanding and time-consuming.

We investigated the proposed SMC algorithm by conducting a range of simulation studies to study the overall behaviour of the method when we adjust the scale parameter or the number of rankings when generating the data set to perform inference on, or when we alter the number of particles, the number of rankings observed at each time step or the number of times we apply MCMC move kernels when running the SMC algorithm. We find that the general behaviour of the proposed method is as expected under the general SMC theory. We also applied the SMC method to some example data sets, including the benchmark Sushi data set, and we conclude that find that the SMC method can save time and computational effort to perform sequential inference and gives posterior estimates similar to that found when employing MCMC methods. We also found that although the MCMC algorithm has an edge in estimating the shared consensus ranking with a higher level of accuracy, converges to more than one solution.

Chapter 5

Sequential Monte Carlo for the Mallows Model with Partial Rankings

5.1 Introduction

In Chapter 4 we considered the problem of performing sequential inference with the Mallows model each time we observed a collection of new full rankings. In particular, we highlighted that the application of SMC methods can be used as an alternative to MCMC to perform inference with the Bayesian Mallows model in an online setting. We consider the problem addressed in Chapter 4 again except, this time, we aim to update the estimate of the posterior distribution for every additional partially observed ranking or collection of rankings. These are known as partial rankings and are how preference data is often presented, particularly when we have a large number of items. The partial rankings may contain an assessor's top-k preferred items, but they may also be rankings with some ranks missing at random. When handling partial rankings within an SMC framework, data augmentation is required before we can reweight the particles to be a representative sample of the new target posterior. This can be handled easily in the Bayesian framework and we can use a proposal distribution to create the auxiliary components of each new partial ranking.

The structure of this chapter is as follows. In Section 5.2, we formulate the problem that first considers the observation of new partial rankings. The proposed distributions that we use for data augmentation on the missing components of each partial ranking are discussed in Section 5.2.1. In particular, we consider the uniform proposal method and the pseudo-likelihood augmentation method, a new proposal distribution, that considers the current model parameters values of each particle to inform the augmentation process. The proposed SMC methodology is presented in Section 5.2.2 and is tested with simulated and real data sets in Section 5.3.

5.2 Proposed method

We assume that $\mathbf{R}$ now represents a partial ranking for n distinct items, and over a discrete timeline, $t = 1, \ldots, T$, we observe in total M_T partial rankings $\mathbf{R}_1, \ldots, \mathbf{R}_{M_T}$. We assume that each partial ranking $\mathbf{R}_j,\ j = 1, \ldots, M_T$, expressed by each assessor, is drawn from a single Mallows model. The missing components of each partial ranking can be augmented to obtain an auxiliary full ranking, defined as $\tilde{\mathbf{R}}$, from a set $\mathcal{S}_j = \{\tilde{\mathbf{R}}_j \in \mathcal{P}_n : \tilde{R}_{ij} = \mathbf{X}_j^{-1}(A_i) \text{ if } A_i \in A_j\},\ j = 1, \ldots, M_T$, which represents the set of possible complete rankings that are compatible with $\mathbf{R}$. Here, $\mathbf{X} = \mathbf{R}^{-1}$ is an ordering for a set of items $\mathcal{A} = \{A_1, \dots, A_n\}$. The task is to estimate the sequence of Mallows posterior distributions for M_t partial rankings over a discrete timeline $t = 1, \dots, T$, defined as

$$\pi_t(\boldsymbol{\rho}, \alpha | \mathbf{R}_1, \dots, \mathbf{R}_{M_t}) = \pi(\boldsymbol{\rho})\pi(\alpha) \sum_{\tilde{\mathbf{R}}_1 \in \mathcal{S}_1} \cdots \sum_{\tilde{\mathbf{R}}_{M_t} \in \mathcal{S}_{M_t}} p(\boldsymbol{\rho}, \alpha, \tilde{\mathbf{R}}_1, \dots, \tilde{\mathbf{R}}_{M_t} | \mathbf{R}_1, \dots, \mathbf{R}_{M_t}).$$

We assume throughout that the partial rankings expressed by each user remain fixed and we do not receive any further information over time. This is the simplest case to consider because we only need to incorporate a proposal for augmenting the missing components of each ranking which is then accounted for in the particle weight updates of the SMC algorithm.

5.2.1 Data augmentation methods

The proposed SMC algorithm for the Mallows model with partial rankings follows the resample-move framework of Berzuini and Gilks (2001). However, when handling partial rankings, we cannot perform an iteration of SMC unless we have full rankings to calculate the incremental weights of each particle during the reweighting stage. To overcome this issue, a key component of the methodology for this chapter is how the latent components of each partial ranking are augmented. The augmentation step is performed before we reweight the particles to perform the remaining steps of SMC. This approach has been discussed in Berzuini and Gilks (2003) to handle partially observed data within the resample-move framework. We describe two methods for creating auxiliary complete rankings given an observed partial ranking.

The uniform augmentation method

In the MCMC algorithm of Vitelli et al. (2018), the latent components of the partial rankings are initialised by assigning missing ranks to unranked items uniformly at random with the allowable augmentations of the missing ranks. A uniform proposal is then used to update the candidate augmented rankings in each MCMC iteration. We will use this method for data augmentation in the proposed SMC framework. Let $n_u < n$ be the number of unranked items in a partial ranking $\mathbf{R}$, then we select a possible augmentation from the set $\mathcal{S}$ with probability $q(\tilde{\mathbf{R}}|\mathbf{R}) = \frac{1}{n_u!}$. This approach is simple, but it does not take into account any existing information that we may have from the particle set in the SMC algorithm, such as the current estimates of the consensus ranking and the scale parameter of the Mallows model. It is likely that the uniform proposal distribution will not be effective in the proposed SMC framework. This motivates the need to propose another augmentation method that would improve the proposed auxiliary rankings with the current estimated posterior distribution.

The pseudo-likelihood augmentation method

We propose the pseudo-likelihood augmentation method as an alternative method that can make an informed choice of how each partial ranking's missing information is completed using the current estimated values of the Mallows model parameters. This is done by constructing several univariate Mallows distributions to sample a rank for each unranked item conditioned on the item's rank in the estimated consensus ranking, the scale parameter and the set of possible ranks. This approach is similar to the approximation of the partition function in Vitelli et al. (2018). We note that this particular method for assigning ranks only applies if $d(\cdot, \cdot)$ is either the footrule or the Spearman distance. This is because we cannot use alternative distance functions to measure the distances between the consensus ranking and the observed ranking for one item.

We describe the augmentation method formally. We define $\mathcal{S}_u = \{r_1, \dots, r_{n_u}\}$ as the set of missing ranks in $\mathbf{R}$. Before we perform the augmentation, we specify an item ordering, $\mathcal{O} = \{o_1, \dots, o_{n_u}\}$, that determines the order in which we assign the ranks in $\mathcal{S}_u$ to the set of unranked items, denoted $\mathcal{I} = \{i_1, \dots, i_{n_u}\}$. The item order can be viewed as a permutation of the elements in $\mathcal{I}$ and it can be chosen randomly with the elements in $\mathcal{I}$ or in some deterministic way, e.g., we select the order by the unranked item's rank in $\boldsymbol{\rho}$.

The augmentation process iterates through the elements in $\mathcal{O}$ and as we advance through each iteration, there will be one less option of a rank that an item in $\mathcal{I}$ can be assigned to. In other words, the first item in the item ordering will have all remaining possible ranks in $\mathcal{S}_u$ it can be assigned to, and the second item in the ordering will have $n_u - 1$ remaining possible ranks in $\mathcal{S}_u$ that it can be assigned to, and so on. To assign an item in $\mathcal{O}$ to a rank in $\mathcal{S}_u$, we need to calculate the probability of sampling each available rank, that has not been previously sampled for another item earlier in $\mathcal{O}$, for that item. We construct a Mallows model which has the item's rank in the consensus ranking as the consensus rank $\boldsymbol{\rho}_{o_k}$, $k \in \{1, \dots, n_u\}$, the scale parameter, α, and the number of unranked items n_u. Then, the probability observing each potential item rank $\tilde{\mathbf{R}}_{o_k}$ is

$$p(\tilde{\mathbf{R}}_{o_k}|o_k, \alpha, \boldsymbol{\rho}_{o_1}, \dots, \boldsymbol{\rho}_{o_{k-1}}, \tilde{\mathbf{R}}_{o_1}, \dots, \tilde{\mathbf{R}}_{o_{k-1}}) = \frac{\exp\{-\frac{\alpha}{n_u} d(\tilde{\mathbf{R}}_{o_k}, \boldsymbol{\rho}_{o_k})\}}{\sum_{\tilde{r}_{o_k} \in \mathcal{S}_u \setminus \{\tilde{r}_{o_1}, \dots, \tilde{r}_{o_{k-1}}\}} \exp\{-\frac{\alpha}{n_u} d(\tilde{\mathbf{r}}_{o_k}, \boldsymbol{\rho}_{o_k})\}}.$$

Then, we sample a rank multinomially using the normalised probabilities from the proposal distribution. The selected rank is assigned and removed from $\mathcal{S}_u$ before we assign a rank for the next item $\mathcal{O}$. We repeat the process until we have iterated through every element in the item ordering. The final item in the ordering to be assigned a rank is deterministic since we only have one remaining rank left. Therefore, the probability of creating a particular augmented ranking is the product of the marginal probabilities

$$\begin{aligned} q(\tilde{\mathbf{R}}|i_1, \dots, i_{m_u}, \alpha, \boldsymbol{\rho}) &= q(\tilde{\mathbf{R}}|o_1, \dots, o_{m_u}, \alpha, \boldsymbol{\rho}) \\ &= p(\tilde{\mathbf{R}}_{o_1}|o_1, \alpha, \boldsymbol{\rho}_{o_1}) \cdot p(\tilde{\mathbf{R}}_{o_2}|o_2, \alpha, \boldsymbol{\rho}_{o_1}, \tilde{\mathbf{R}}_{o_1}) \cdot \ \dots \ \cdot \\ &\quad p(\tilde{\mathbf{R}}_{o_{m_u-1}}|o_{m_u-1}, \alpha, \boldsymbol{\rho}_{o_1}, \dots, \boldsymbol{\rho}_{o_{m_u-2}}, \tilde{\mathbf{R}}_{o_1}, \dots, \tilde{\mathbf{R}}_{o_{m_u-2}}) \cdot \\ &\quad p(\tilde{\mathbf{R}}_{o_{m_u}}|o_{m_u}, \alpha, \boldsymbol{\rho}_{o_1}, \dots, \boldsymbol{\rho}_{o_{m_u-1}}, \tilde{\mathbf{R}}_{o_1}, \dots, \tilde{\mathbf{R}}_{o_{m_u-1}}). \end{aligned}$$

The pseudo-likelihood augmentation method is presented in Algorithm 11.

Algorithm 11: Pseudo-likelihood Augmentation Method

```
Input: $\alpha, \boldsymbol{\rho},\ n_u,\ \mathbf{R},\ \mathcal{O} = \{o_1, \dots, o_{n_u}\},\ \mathcal{S}_u = \{r_1, \dots, r_{n_u}\}$.
Output: Complete augmented ranking $\tilde{\mathbf{R}}$.
for $k = 1 : n_u$ do
    Sample $\tilde{R}_{o_k} \in \mathcal{S}_u$ with probability
        $p(\tilde{R}_{o_k}|o_k, \alpha, \boldsymbol{\rho}, \tilde{R}_{o_1}, \dots, \tilde{R}_{o_{k-1}}) = \frac{\exp\{-\frac{\alpha}{n_u} d(\tilde{R}_{o_k}, \boldsymbol{\rho}_{o_k})\}}{\sum_{\tilde{r}_{o_k} \in \mathcal{S}_u \setminus \{\tilde{r}_{o_1}, \dots, \tilde{r}_{o_{k-1}}\}} \exp\{-\frac{\alpha}{n_u} d(\tilde{r}_{o_k}, \boldsymbol{\rho}_{o_k})\}}$.
    Redefine $\mathcal{S}_u = \mathcal{S}_u \setminus \{\tilde{R}_{o_k}\}$.
end
```

5.2.2 Methodology

The methodology for this problem closely follows the SMC algorithm for full rankings in Chapter 4. If we begin with no observed rankings at time $t = 0$, that is, $M_0 = 0$, then we generate an equally weighted particle set $\{\boldsymbol{\theta}_0^{(i)} = (\boldsymbol{\rho}_0^{(i)}, \alpha_0^{(i)})\}_{i=1}^N$ using the prior distributions for the parameters of the Mallows model. Otherwise, if $M_0 > 0$, we can run an MCMC sampler with M_0 rankings to obtain a thinned sample (after burn-in) of size N, that can be utilised as an initial particle set. At iteration $t-1$, each particle contains the following information: the model parameter values, $\boldsymbol{\theta}_{t-1}^{(i)} = (\boldsymbol{\rho}_{t-1}^{(i)}, \alpha_{t-1}^{(i)})$, the particle weights $w_{t-1}^{(i)}$, and a complete updated ranking set $\tilde{\mathbf{R}}_{1:M_{t-1}}^{(i)}$.

At iteration t, we observe $(M_t - M_{t-1})$ new partial rankings and perform data augmentation using either the uniform or the proposed pseudo-likelihood augmentation methods to obtain $\tilde{\mathbf{R}}_{M_{t-1}+1:M_t}$. Next, we adjust the particle weights that currently approximate the density $\pi_{t-1}(\boldsymbol{\rho}, \alpha, \tilde{\mathbf{R}}_{1:M_{t-1}}|\mathbf{R}_{1:M_{t-1}})$ so that they are now weighted with respect to $\pi_t(\boldsymbol{\rho}, \alpha, \tilde{\mathbf{R}}_{1:M_t}|\mathbf{R}_{1:M_t})$. The incremental weight for each particle is now

$$\begin{aligned}
\tilde{w}_t(\boldsymbol{\theta}_t^{(i)}, \tilde{\mathbf{R}}_{1:M_t}^{(i)}) &= \frac{\pi_t(\boldsymbol{\theta}_t^{(i)}, \tilde{\mathbf{R}}_{1:M_t}^{(i)})}{\pi_{t-1}(\boldsymbol{\theta}_{t-1}^{(i)}, \tilde{\mathbf{R}}_{1:M_{t-1}}^{(i)})q_t(\boldsymbol{\theta}_t^{(i)}, \tilde{\mathbf{R}}_{1:M_t}^{(i)}|\boldsymbol{\theta}_{t-1}^{(i)}, \tilde{\mathbf{R}}_{1:M_{t-1}}^{(i)})} \\
&= \frac{\pi(\boldsymbol{\theta}_t^{(i)}, \tilde{\mathbf{R}}_{1:M_t}^{(i)}|\mathbf{R}_{1:M_t})}{\pi(\boldsymbol{\theta}_{t-1}^{(i)}, \tilde{\mathbf{R}}_{1:M_{t-1}}^{(i)}|\mathbf{R}_{1:M_{t-1}}^{(i)})q(\boldsymbol{\theta}_t^{(i)}, \tilde{\mathbf{R}}_{1:M_t}^{(i)}|\boldsymbol{\theta}_{t-1}^{(i)}, \tilde{\mathbf{R}}_{1:M_{t-1}}^{(i)})} \\
&= \frac{\pi(\boldsymbol{\theta}_t^{(i)})p(\tilde{\mathbf{R}}_{1:M_t}^{(i)}|\boldsymbol{\theta}_t^{(i)}, \mathbf{R}_{1:M_t})/p(\mathbf{R}_{1:M_t})}{\pi(\boldsymbol{\theta}_{t-1}^{(i)})p(\tilde{\mathbf{R}}_{1:M_{t-1}}^{(i)}|\boldsymbol{\theta}_{t-1}^{(i)}, \mathbf{R}_{1:M_{t-1}})/p(\mathbf{R}_{1:M_{t-1}})} \\
&\quad \times \frac{1}{q(\boldsymbol{\theta}_t^{(i)}, \tilde{\mathbf{R}}_{1:M_t}^{(i)}|\boldsymbol{\theta}_{t-1}^{(i)}, \tilde{\mathbf{R}}_{1:M_{t-1}}^{(i)})} \qquad (5.2.1) \\
&= \frac{p(\tilde{\mathbf{R}}_{M_{t-1}+1:M_t}^{(i)}|\boldsymbol{\theta}_t^{(i)}, \mathbf{R}_{M_{t-1}+1:M_t})p(\mathbf{R}_{1:M_{t-1}})}{q(\boldsymbol{\theta}_t^{(i)}, \tilde{\mathbf{R}}_{1:M_t}^{(i)}|\boldsymbol{\theta}_{t-1}^{(i)}, \tilde{\mathbf{R}}_{1:M_{t-1}}^{(i)})p(\mathbf{R}_{1:M_t})} \qquad (5.2.2) \\
&\propto \frac{p(\tilde{\mathbf{R}}_{M_{t-1}+1:M_t}^{(i)}|\boldsymbol{\theta}_t^{(i)}, \mathbf{R}_{M_{t-1}+1:M_t})}{q(\boldsymbol{\theta}_t^{(i)}, \tilde{\mathbf{R}}_{1:M_t}^{(i)}|\boldsymbol{\theta}_{t-1}^{(i)}, \tilde{\mathbf{R}}_{1:M_{t-1}}^{(i)})}. \qquad (5.2.3)
\end{aligned}$$

Equation (5.2.1) simplifies to (5.2.2) we use the estimated model parameters from the previous time step to calculate the new weights, so that the priors $\pi(\boldsymbol{\theta}_t^{(i)})$ and $\pi(\boldsymbol{\theta}_{t-1}^{(i)})$ cancel. We also only need to consider the likelihood for the new observed rankings at time t. However, we also need to account for the data augmentation for the missing components of each partial ranking with a proposal distribution. This is defined as $q(\boldsymbol{\theta}_t^{(i)}, \tilde{\mathbf{R}}_{1:M_t}^{(i)}|\boldsymbol{\theta}_{1:t-1}^{(i)}, \tilde{\mathbf{R}}_{1:M_{t-1}}^{(i)}) = q(\tilde{\mathbf{R}}_{M_{t-1}+1:M_t}^{(i)}|\boldsymbol{\theta}_{t-1}^{(i)}, \mathbf{R}_{M_{t-1}+1:M_t})$ and we consider the uniform and the pseudo-likelihood augmentation methods in the proposal distribution. Therefore, (5.2.3) can be rewritten as

$$\tilde{w}_t(\boldsymbol{\theta}_t^{(i)}, \tilde{\mathbf{R}}_{1:M_t}^{(i)}) = \frac{\exp\left\{ -\frac{\alpha_{t-1}^{(i)}}{n} \sum_{j=M_{t-1}+1}^{M_t} d(\tilde{\mathbf{R}}_j^{(i)}, \boldsymbol{\rho}_{t-1}^{(i)}) \right\}}{Z_n(\alpha_{t-1}^{(i)})^{(M_t - M_{t-1})} q_t(\tilde{\mathbf{R}}_{M_{t-1}+1:M_t}^{(i)}|\boldsymbol{\theta}_{t-1}^{(i)}, \mathbf{R}_{M_{t-1}+1:M_t})},$$

where $\alpha_{t-1}^{(i)}$ and $\boldsymbol{\rho}_{t-1}^{(i)}$, $i = 1, \ldots, N$ are the values of the parameter estimates of the

Mallows posterior before reweighting. The particle weights are updated, $w_t^{(i)}(\boldsymbol{\theta}_t) = W_{t-1}^{(i)}(\boldsymbol{\theta}_{t-1})\tilde{w}_t^{(i)}(\boldsymbol{\theta}_t)$, and are normalised to obtain $W_t(\boldsymbol{\theta}_t)$. We now have the following information in the particle set: $\{\bar{\boldsymbol{\theta}}_t^{(i)},\ \tilde{\mathbf{R}}_{1:M_t}^{(i)},\ W_t^{(i)}\}_{i=1}^N$. Then, we resample using multinomial resampling and propagate the particles using the Metropolis-Hastings MCMC move kernel $K_t(\bar{\boldsymbol{\theta}}_t^{(i)}, \boldsymbol{\theta}_t^{(i)})$. Specifically, we use leap-and-shift proposal distribution (Vitelli et al., 2018) to propose $\boldsymbol{\rho}_t^{'(i)}$; log-normal distribution to propose $\alpha_t^{'(i)}$, and either the uniform or the pseudo-likelihood proposal distribution to propose new augmented rankings $\tilde{\mathbf{R}}_{1:M_t}^{'}$ conditioned on the observed components of each ranking $\mathbf{R}_{1:M_t}$. The pseudocode for SMC with the Mallows model for new partial rankings is presented in Algorithm 12.

Algorithm 12: Sequential Monte Carlo for the Mallows Model with Partial Rankings

```
Input: $\lambda,\ \sigma_\alpha,\ d(\cdot,\cdot),\ K,\ L,\ N,\ M_1,\dots,M_T;\ \mathbf{R}_1,\dots,\mathbf{R}_{M_T};\ T,\ Z_n(\alpha)$.
Output: Posterior distributions of $\boldsymbol{\rho},\alpha$.
Initialisation of SMC: randomly generate $\boldsymbol{\rho}_0^{(i)},\alpha_0^{(i)}$ for $i=1,\dots,N$.
for $t=1,\dots,T$ do
    Observe $\mathbf{R}_{M_{t-1}+1:M_t}$.
    for $i=1:N$ do
        for $j=M_{t-1}+1:M_t$ do
            Sample $\tilde{\mathbf{R}}_j^{(i)}$ with proposal $\tilde{\mathbf{R}}_j \sim q(\tilde{\mathbf{R}}_j|\mathbf{R}_j,\boldsymbol{\rho}_t^{(i)},\alpha_t^{(i)})$.
        end
        Compute $\tilde{w}_t^{(i)} = \frac{\exp\left\{-\frac{\alpha_{t-1}^{(i)}}{n}\sum_{j=M_{t-1}+1}^{M_t} d(\tilde{\mathbf{R}}_j,\boldsymbol{\rho}_{t-1}^{(i)})\right\}}{Z_n(\alpha_{t-1}^{(i)}))^{M_t-M_{t-1}}\prod_{j=M_{t-1}+1}^{M_t} q(\tilde{\mathbf{R}}_j|\mathbf{R}_j,\boldsymbol{\rho}_t^{(i)},\alpha_t^{(i)})}$.
        Compute $w_t^{(i)} = W_{t-1}^{(i)}\tilde{w}_t^{(i)}$.
    end
    Normalise the weights $W_t^{(i)} = w_t^{(i)}/\sum_{i=1}^N w_t^{(i)}$.
    Resample $(k_1,\dots,k_N) \sim \mathcal{M}(W_t^{(1)},\dots,W_t^{(N)})$ and set
     $\{\bar{\boldsymbol{\theta}}_t^{(1:N)},W_t^{(i)}\} \leftarrow \{\boldsymbol{\theta}_t^{(k_1:k_N)},\frac{1}{N}\}$.
    for $i=1:N$ do
        for $k=1,\dots,K$ do
            M-H step: update $\boldsymbol{\rho}_t^{(i)}$ using the leap-and-shift proposal
             $(\boldsymbol{\rho'} \sim \text{L\&S}(\boldsymbol{\rho}_t^{(i)},L))$.
            M-H step: update $\alpha_t^{(i)}$ with log-normal proposal $(\alpha' \sim \log\mathcal{N}(\alpha_t^{(i)},\sigma_a^2))$.
            for $j=1:M_t$ do
                M-H step: update $\tilde{\mathbf{R}}_j^{(i)}$ with proposal $\tilde{\mathbf{R}}_j' \sim q(\tilde{\mathbf{R}}_j|\mathbf{R}_j,\boldsymbol{\rho}_t^{(i)},\alpha_t^{(i)})$.
            end
        end
    end
end
```

5.3 Experimental analyses

We assess the proposed SMC sampler using simulated and real data. The simulated data sets were created using the function `sample_mallows` in the `BayesMallows` R package (Sørensen et al., 2020). The parameters we used to generate the data set were `rho_0 = c(1:n_items)` and `alpha_0 = 2`, as the underlying parameters of the

Mallows model, and simulated a data set of 100 complete rankings. The simulated and real data sets are filtered so that only the top-$\frac{n}{2}$ ranked items could be observed in each ranking.

We describe the default experiment set-up. We ran the MCMC algorithm of Vitelli et al. (2018) using the function `compute_mallows` in the `BayesMallows` R package for 10,000 iterations with the full partial data set and we discarded the first 5,000 iterations as burn-in. In the SMC algorithm, we used $N = 1000$ particles and the latent components of each partial ranking were sampled using either the uniform or the pseudo-likelihood augmentation method before reweighting and resampling. We specified ten applications of the MCMC move kernels for $\boldsymbol{\rho}$, α and $\tilde{\mathbf{R}}_{1:M_t}$ for each particle after resampling in each SMC iteration. For both methods, we set as a default: the leap-size parameter in the leap-and-shift proposal for $\boldsymbol{\rho}$ to be $L = \lfloor n/5 \rfloor$; the hyper-parameter and the standard deviation for the exponential proposal distribution for α to be $\lambda = 0.1$ and $\sigma_\alpha = 0.5$ respectively; and the measured distance between rankings as the footrule distance.

The SMC algorithm with both augmentation methods is assessed in comparison to the MCMC algorithm using the same metrics as discussed in Chapter 4. The SMC algorithm with the uniform augmentation method is referred to as SMC-U in each figure and table in this chapter, whilst the SMC algorithm with the pseudo-likelihood augmentation method is referred to as SMC-P. Heat plots were used to visualise the marginal posterior probabilities of ranking each item in $\boldsymbol{\rho}$. The posterior expected mean squared error (MSE) was reported to indicate the spread of the sampled values for $\boldsymbol{\rho}$ from each of the Bayesian methods given the true consensus ranking. We also reported the posterior estimate of α and its 95% HDPI. When relevant, we also provided the CP consensus ranking; the cumulative probability of each item to be ranked in shared consensus position, or higher, $p(\boldsymbol{\rho}_i \leq i)$; and the 95% HDPI for each item in $\boldsymbol{\rho}$.

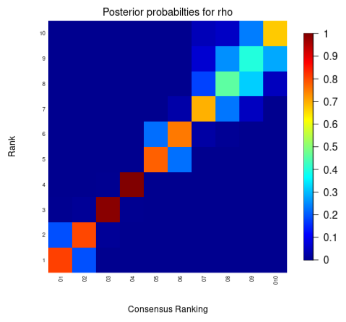
5.3.1 Simulated data: 10 items

In the first experiment, we propagated $N = 1000$ SMC particles over $T = 10$ time steps by introducing ten new top-5 rankings in each iteration. In total, 100,000 applications of MCMC move kernels were applied to each parameter in the particle set in the SMC algorithm. We ran the MCMC algorithm for 10,000 iterations to sample proposals for $\boldsymbol{\rho}$, α and $\tilde{\mathbf{R}}_{1:100}$. Figure 5.3.1 shows the marginal posterior distribution of $\boldsymbol{\rho}$ and α of the output from the MCMC algorithm and the SMC algorithm with both augmentation methods. The items in the heat plots in Figures 5.3.1a-5.3.1c are ordered according to $\boldsymbol{\rho}^* = (1, 2, \ldots, 10)$, which happened to be the CP consensus for all three methods. Table 5.3.1 shows the CP consensus estimate given the partial data set, the cumulative probability of each item ranked in the shared consensus position, or higher, $p(\boldsymbol{\rho}_i \leq i)$ and the 95% highest HPDI for each item across all three methods. The MSE for the MCMC and the SMC with the pseudo-likelihood augmentation method produce similar values whereas the SMC with the uniform augmentation method has a smaller MSE value. However, if we are assuming that SMC is to be used as an alternative to MCMC, then SMC with the uniform augmentation method is not estimating this posterior well. The MSE represents the sample spread around a reference point $\boldsymbol{\rho^*}$. The spread may not be close to $\boldsymbol{\rho^*}$, resulting in a value of 0, but we expect the MSE to be the same for different samplers of the same posterior distribution given the data. The small MSE value indicates a higher value in the posterior estimate for α as this implies that the particle set values for $\boldsymbol{\rho}$ have a small variation around the "true" consensus ranking in distance, resulting in the lower MSE value. The marginal posterior probabilities for the top-5 ranked items in the heat plots in Figures 5.3.1a-5.3.1c are similar across all three methods. We also see more uncertainty across the lowest five ranked items in $\boldsymbol{\rho}$. This is expected because these items will have been less likely to have been assigned a top-5 rank in the partially observed data set. The lowest-ranked item in $\boldsymbol{\rho}$ has a higher marginal posterior probability of being ranked the lowest rank in comparison to the items ranked 6-9 because this item will have received the least number of observed ranks in the data set, so we can be more certain that it is likely to be the lowest-ranked item.

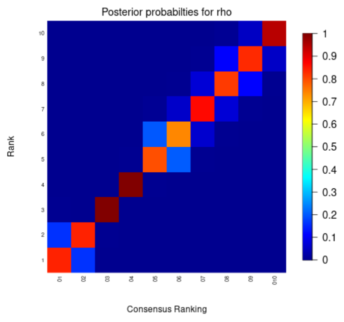
The cumulative probabilities and HDPIs in Table 5.3.1 are very similar across all three methods for the top-5 ranked items. However, there is a significant increase in certainty for items 6-10 when using the uniform augmentation method in SMC whilst the HDPIs for items 6-10 remain similar with the MCMC method and the SMC method with the pseudo-likelihood augmentation method.

We also repeated each experiment ten times and recorded the MSE and time taken to run each algorithm. This is summarised in Table 5.3.2. The MCMC algorithm takes, on average, 0.6 seconds to run with the full data set of 100 partial rankings, whereas the SMC algorithm takes, on average, 70 seconds (with the uniform augmentation method) and 103 seconds (with the pseudo-likelihood augmentation method) to perform ten sequential updates of the posterior estimate for every ten new rankings observed. If we were to observe an additional ten new rankings, then the MCMC algorithm would take at least 0.5 seconds to run whereas the SMC algorithm with each augmentation method would take approximately $\sim$7 and $\sim$10 seconds respectively to update the posterior for a similar amount of computation effort. The SMC algorithm takes longer to run because we are applying significantly more MCMC move kernels than the number of iterations in the MCMC algorithm itself. Further optimisation of the SMC code could improve the run time. In Table 5.3.2 the confidence interval for the MSE for the SMC with the pseudo-likelihood augmentation method is close to that of the MCMC.

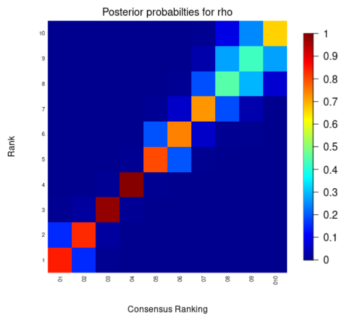
We conclude from this experiment that the pseudo-likelihood augmentation has a slight improvement over the uniform approach because we are able to achieve marginal posterior estimates for Mallows model parameters that are closer to the estimates achieved through MCMC. It is clear that the SMC methods suffer from slow computational speed for the same amount of computational effort in each iteration in comparison to MCMC.



(a) MCMC: MSE = 0.36, $\hat{\alpha}$ = 2.00, (1.70, 2.29).



(b) SMC-U: MSE = 0.14, $\hat{\alpha}$ = 2.07 (1.74, 2.37).



(c) SMC-P: MSE = 0.32, $\hat{\alpha}$ = 2.01 (1.71, 2.37).

Figure 5.3.1: Top: Heat plots of the posterior probabilities, for $n = 10$ items, for being ranked as the k^{th} most preferred, for $k = 1, ..., 10$ using the MCMC algorithm (left), the SMC algorithm with the uniform augmentation method (middle), and the SMC algorithm with the pseudo-likelihood augmentation method (right). On the x-axis, the items are ordered according to their true consensus. The MSE value and the mean estimate of α with its corresponding 95% HDPI are included in the captions.

		$p(\boldsymbol{\rho}_i \leq i)$			**95% HDPI**		
$\boldsymbol{\rho}$	**CP**	**MCMC**	**SMC-U**	**SMC-P**	**MCMC**	**SMC-U**	**SMC-P**
ρ_1	1	0.81	0.84	0.85	[1,2]	[1,2]	[1,2]
ρ_2	2	0.99	1.00	0.99	[1,2]	[1,2]	[1,2]
ρ_3	3	1.00	1.00	1.00	[3]	[3]	[3]
ρ_4	4	1.00	1.00	1.00	[4]	[4]	[4]
ρ_5	5	0.78	0.80	0.80	[5,6]	[5,6]	[5,6]
ρ_6	6	0.97	0.94	0.94	[5,6]	[5,7]	[5,7]
ρ_7	7	0.72	0.93	0.78	[7,10]	[6,8]	[6,8]
ρ_8	8	0.69	0.89	0.64	[7,10]	[7,9]	[7,10]
ρ_9	9	0.77	0.95	0.75	[8,10]	[8,10]	[8,10]
ρ_{10}	10	1.00	1.00	1.00	[9,10]	[9,10]	[8,10]

Table 5.3.1: Posterior estimates of the ten items in the partially observed simulated data set. Items are arranged according to the CP consensus ranking with their corresponding cumulative probabilities and 95% HDPIs.

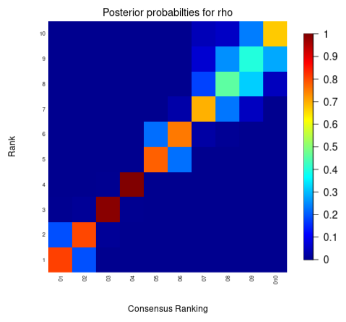
We reduce the number of MCMC kernel applications during the move stage of the proposed SMC algorithm to see if we can perform inference with SMC for less computational effort than using MCMC. Currently, in each iteration of the SMC algorithm, we apply as many MCMC iterations to sample for the consensus ranking,

	MCMC		**SMC-U**		**SMC-P**	
Method	Mean	95% CI	Mean	95% CI	Mean	95% CI
MSE	0.31	(0.29, 0.34)	0.13	(0.12, 0.13)	0.33	(0.31, 0.35)
Time (s)	0.57	(0.56, 0.58)	69.63	(69.31, 69,94)	102.62	(102.11, 103.13)

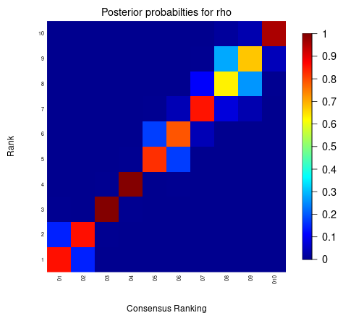
Table 5.3.2: Summary table of the MSE and time taken for the ten experiment runs of the MCMC algorithm and the SMC algorithm with both augmentation methods for the partially observed simulated data set of $n = 10$ items.

the scale parameter and the set of observed partial rankings across the particle set in SMC as we would if we were to run the MCMC sampler with the same set of partial rankings. We repeated the experiment and applied for each particle: five MCMC applications of the leap-and-shift proposal for sampling $\boldsymbol{\rho}$; one application of the exponential proposal for α; and one application to sample new augmentations for $\mathbf{R}_{1:M_t}$. We also introduced one new ranking at a time for 100 iterations in total. Therefore, for each new ranking introduced, we applied 7,000 MCMC move kernels.

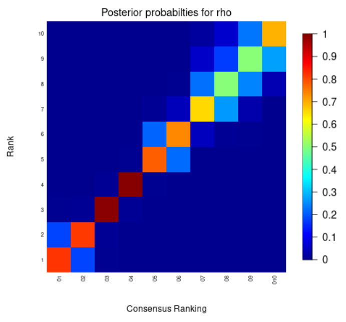
The results for the estimated marginal posterior distributions for $\boldsymbol{\rho}$ and α are provided in Figure 5.3.2. We note that the results from the MCMC experiment are the same as the results from the previous experiment. The MSE for MCMC and SMC with the pseudo-likelihood sampler still produce similar values whereas SMC with the uniform augmentation method has a much smaller MSE, resulting in a higher value in the posterior estimate for α. The posterior probabilities for items 7 and 9 in $\boldsymbol{\rho}$ with the SMC algorithm with uniform augmentation method are different to those obtained with the pseudo-likelihood method and this is reflected in the different HDPIs for these items in Table 5.3.3. However, the posterior estimate for α is reasonable. We repeated this experiment ten times and recorded the MSE and time taken to run each algorithm in Table 5.3.4. As we would expect, by reducing the number of MCMC move kernels, particularly for $\mathbf{R}_{1:M_t}$, the time taken to run 100 iterations of the SMC algorithm takes longer than to run one MCMC chain with the full data set of 100 partial rankings. If we were to observe an additional new ranking, then the MCMC algorithm would take at least 0.5 seconds to run whereas the SMC algorithm with each augmentation method would take approximately $\sim$0.8 and $\sim$1.1 seconds respectively to perform the sequential posterior update the posterior with less computation effort. Further optimisation with the algorithm could reduce computational time.



(a) MCMC: MSE = 0.36, $\hat{\alpha}$ = 2.00, (1.70, 2.29).



(b) SMC-U: MSE = 0.17, $\hat{\alpha}$ = 2.12 (1.84, 2.44).



(c) SMC-P: MSE = 0.32, $\hat{\alpha}$ = 2.01 (1.74, 2.34).

Figure 5.3.2: Top: Heat plots of the posterior probabilities, for $n = 10$ items, for being ranked as the k^{th} most preferred, for $k = 1, ..., 10$ using the MCMC algorithm (left), the SMC algorithm with the uniform augmentation method and reduced computational effort (middle), and the SMC algorithm with the pseudo-likelihood augmentation method (right) and reduced computational effort. On the x-axis, the items are ordered according to their true consensus. The MSE value and the mean estimate of α with its corresponding 95% HDPI are included in the captions.

5.3.2 Real data: Sushi data

We modified the benchmark Sushi data set (Kamishima, 2003) so that we only observe the first $M = 100$ rankings, each with their top-5 sushi preferences, and ran the algorithms with the default experiment set-up. The heat plots in the Figures 5.3.3a-

		$p(\boldsymbol{\rho}_i \leq i)$			**95% HDPI**		
$\boldsymbol{\rho}$	**CP**	**MCMC**	**SMC-U**	**SMC-P**	**MCMC**	**SMC-U**	**SMC-P**
ρ_1	1	0.81	0.86	0.82	[1,2]	[1,2]	[1,2]
ρ_2	2	0.99	1.00	1.00	[1,2]	[1,2]	[1,2]
ρ_3	3	1.00	1.00	1.00	[3]	[3]	[3]
ρ_4	4	1.00	1.00	1.00	[4]	[4]	[4]
ρ_5	5	0.78	0.83	0.78	[5,6]	[5,6]	[5,6]
ρ_6	6	0.97	0.96	0.96	[5,6]	[5,6]	[5,7]
ρ_7	7	0.72	0.89	0.70	[7,10]	[7,8]	[6,9]
ρ_8	8	0.69	0.71	0.77	[7,10]	[7,9]	[7,10]
ρ_9	9	0.77	0.97	0.77	[8,10]	[8,10]	[8,10]
ρ_{10}	10	1.00	1.00	1.00	[9,10]	[10]	[9,10]

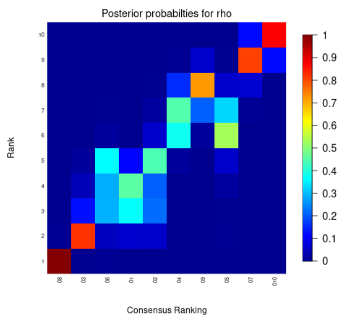
Table 5.3.3: Posterior estimates of the ten items in the partially observed simulated data set. Items are arranged according to the CP consensus ranking with their corresponding cumulative probabilities and 95% HDPIs.

	MCMC		**SMC-U**		**SMC-P**	
Method	Mean	95% CI	Mean	95% CI	Mean	95% CI
MSE	0.31	(0.29, 0.34)	0.18	(0.17, 0.19)	0.31	(0.30, 0.32)
Time (s)	0.57	(0.56, 0.58)	78.66	(78.08, 79.24)	108.35	(107.89, 108.81)

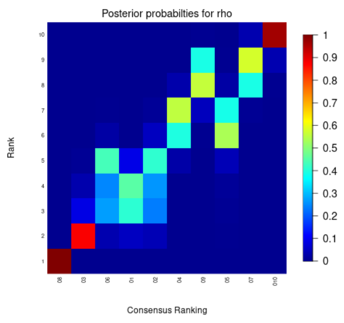
Table 5.3.4: Summary table of the MSE and time taken for the ten experiment runs of the MCMC algorithm and the SMC algorithm with both augmentation methods and reduced computational effort for the partially observed simulated data set of $n = 10$ items.

5.3.3c show that the SMC algorithm with the pseudo-likelihood augmentation method produces a similar marginal posterior for $\boldsymbol{\rho}$ that is obtained from the MCMC output. This behaviour is also shown in their respective marginal posteriors for α in the captions of Figures 5.3.3a-5.3.3c and their estimated CP consensus rankings in Table 5.3.5 are the same. We reach the same conclusions as described in the first simulated data set experiment in Section 5.3.1.

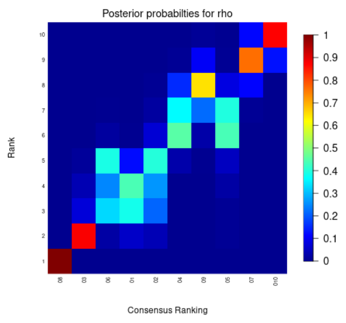
Similarly, after performing ten runs of this experiment with both algorithms, we see in Table 5.3.6 that all three methods give similar MSE values. We also see that if we were to introduce ten new rankings, then MCMC would take at least 0.5 seconds to run whilst the SMC algorithm would take approximately ∼7 and ∼10 seconds with each augmentation method to update the posterior estimate for the same amount of computational effort.



(a) MCMC: MSE = 1.06, $\hat{\alpha} = 1.71, (1.40, 1.99)$.



(b) SMC-U: MSE = 1.07, $\hat{\alpha} = 1.74, (1.43, 2.02)$.



(c) SMC-P: MSE = 1.04, $\hat{\alpha} = 1.71, (1.41, 2.05)$.

Figure 5.3.3: Top: Heat plots of the posterior probabilities, for $n = 10$ sushi items, for being ranked as the k^{th} most preferred, for $k = 1, ..., 10$ using the MCMC algorithm (left), the SMC algorithm with the uniform augmentation method (middle), and the SMC algorithm with the pseudo-likelihood augmentation method (right). On the x-axis, the items are ordered according to their CP consensus. The MSE value and the mean estimate of α with its corresponding 95% HDPI are included in the captions.

		$p(\boldsymbol{\rho}_i \leq i)$			**95% HDPI**		
$\boldsymbol{\rho}$	**CP**	**MCMC**	**SMC-U**	**SMC-P**	**MCMC**	**SMC-U**	**SMC-P**
ρ_1	Fatty tuna	1.00	1.00	1.00	[1]	[1]	[1]
ρ_2	Tuna	0.82	0.89	0.85	[2,3]	[2,3]	[2,4]
ρ_3	Shrimp	0.43	0.44	0.45	[3,5]	[3,5]	[2,5]
ρ_4	Salmon roe	0.62	0.56	0.59	[3,6]	[3,5]	[2,5]
ρ_5	Sea eel	0.92	0.94	0.92	[3,6]	[3,6]	[3,6]
ρ_6	Sea urchin	0.61	0.53	0.56	[5,8]	[6,8]	[5,8]
ρ_7	Squid	0.84	0.96	0.89	[5,8]	[5,7]	[5,8]
ρ_8	Tuna roll	0.94	0.63	0.88	[7,9]	[7,9]	[7,9]
ρ_9	Egg	0.88	0.97	0.85	[8,10]	[8,9]	[8,10]
ρ_{10}	Cucumber roll	1.00	1.00	1.00	[9,10]	[10]	[9,10]

Table 5.3.5: Posterior estimates of the items in the modified Sushi data set. Items are arranged according to the CP consensus ranking with their corresponding cumulative probabilities and 95% HDPIs.

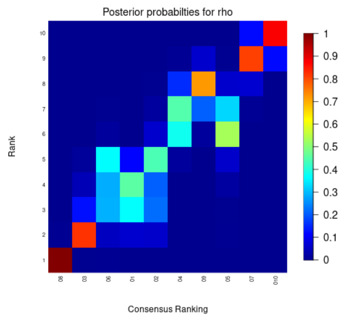
We repeated the experiment using the proposed SMC algorithm by reducing the

	MCMC		**SMC-U**		**SMC-P**	
Method	Mean	95% CI	Mean	95% CI	Mean	95% CI
MSE	1.11	(1.06, 1.15)	1.09	(1.06, 1.12)	1.10	(1.03, 1.16)
Time (s)	0.56	(0.56, 0.57)	72.42	(72.20, 72.64)	105.29	(104.98, 105.60)

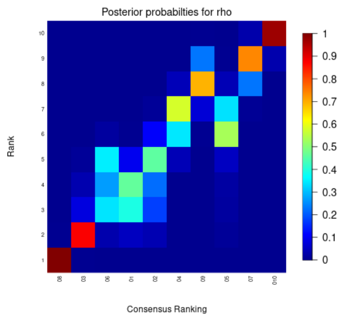
Table 5.3.6: Summary table of the MSE and time taken for the 10 experiment runs of the MCMC algorithm and SMC algorithm with both augmentation methods for the modified Sushi data set.

number of new rankings from ten to one in each iteration. We applied: five MCMC iterations for $\boldsymbol{\rho}$; one MCMC iteration to sample α; and one MCMC iteration to sample new augmentations for $\mathbf{R}_{1:M_t}$ for each particle. In total, we apply 7,000 applications of MCMC move kernel in each SMC iteration to the particle set. The experiment with the MCMC algorithm remained unchanged.

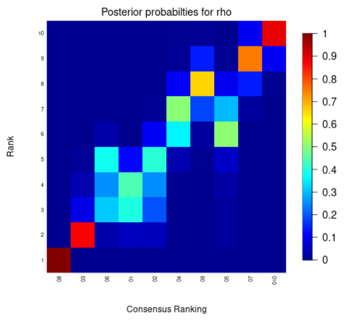
The estimated marginal posterior distributions for Mallows model parameters in Figure 5.3.4 appear to be similar for all three methods in terms of MSE values and the marginal posterior probabilities for the top-5 ranked items. However, SMC with the uniform augmentation shows slightly more certainty in the lower-ranked items. This can be seen in the corresponding cumulative probabilities for items ranked 7 and 9 in Table 5.3.7. The summary of the MSE and run time from ten repetitions of each algorithm is presented in Table 5.3.8. It takes $\sim$80 and $\sim$105 seconds to run the SMC algorithm with both augmentation methods respectively. If an additional ranking was observed, then another iteration of SMC would take approximately $\sim$0.8 and $\sim$1.1 seconds respectively to perform the sequential posterior update. Overall, similar conclusions with the Sushi data set can be made with the second experiment with the simulated data set of ten items in Section 5.3.1.



(a) MCMC: MSE = 1.06, $\hat{\alpha} = 1.71, (1.40, 1.99)$.



(b) SMC-U: MSE = 1.08, $\hat{\alpha} = 1.77, (1.48, 2.01)$.



(c) SMC-P: MSE = 1.09, $\hat{\alpha} = 1.71, (1.35, 1.99)$.

Figure 5.3.4: Top: Heat plots of the posterior probabilities, for $n = 10$ sushi items, for being ranked as the k^{th} most preferred, for $k = 1, ..., 10$ using the MCMC algorithm (left), the SMC algorithm with the uniform augmentation method and reduced computational effort (middle), and the SMC algorithm with the pseudo-likelihood augmentation method (right) and reduced computational effort. On the x-axis, the items are ordered according to their CP consensus. The MSE value and the mean estimate of α with its corresponding 95% HDPI are included in the captions.

		$p(\boldsymbol{\rho}_i \leq i)$			**95% HDPI**		
$\boldsymbol{\rho}$	**CP**	**MCMC**	**SMC-U**	**SMC-P**	**MCMC**	**SMC-U**	**SMC-P**
1	Fatty tuna	1.00	1.00	1.00	[1]	[1]	[1]
2	Tuna	0.82	0.88	0.87	[2,3]	[2,3]	[2,3]
3	Shrimp	0.43	0.44	0.44	[3,5]	[2,5]	[3,5]
4	Salmon roe	0.62	0.64	0.60	[3,6]	[3,5]	[2,5]
5	Sea eel	0.92	0.89	0.89	[3,6]	[3,6]	[2,6]
6	Sea urchin	0.61	0.63	0.60	[5,8]	[5,8]	[5,8]
7	Squid	0.84	0.96	0.89	[5,8]	[6,8]	[6,8]
8	Tuna roll	0.94	0.77	0.86	[7,9]	[7,9]	[7,9]
9	Egg	0.88	0.97	0.90	[8,10]	[8,9]	[8,10]
10	Cucumber roll	1.00	1.00	1.00	[9,10]	[10]	[9,10]

Table 5.3.7: Posterior estimates of the ten items in the modified Sushi data set. Items are arranged according to the CP consensus ranking with their corresponding cumulative probabilities and 95% HDPIs.

5.3.3 Simulated data: 20 items

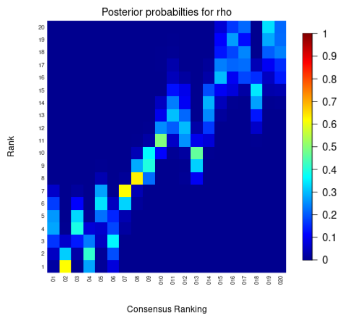
We test the SMC algorithm with a larger number of items in each ranking to observe the resulting behaviour when the size of the permutation space for the latent

	MCMC		**SMC-U**		**SMC-P**	
Method	Mean	95% CI	Mean	95% CI	Mean	95% CI
MSE	1.11	(1.06, 1.15)	1.09	(1.05, 1.12)	1.07	(1.05, 1.09)
Time (s)	0.56	(0.56, 0.57)	79.55	(78.98, 80.12)	105.29	(107.93, 109.29)

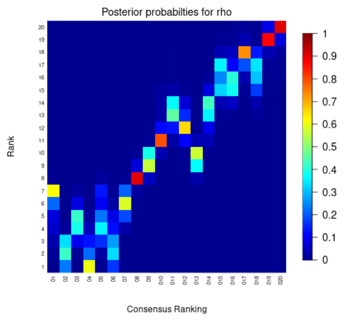
Table 5.3.8: Summary table of the MSE and time taken for the ten experiment runs of the MCMC algorithm and SMC algorithm with both augmentation methods and reduced computational effort for the modified Sushi data set.

components of each partial ranking is increased. We use a simulated data set containing $M = 100$ rankings of the top-10 rankings of $n = 20$ items. In the SMC algorithm, we introduced ten new rankings at a time for $T = 10$ time steps in total with the default experiment settings. In Figure 5.3.5 we have provided the heat plots for the posterior probabilities for the items using the MCMC algorithm of Vitelli et al. (2018), the proposed SMC algorithm with the uniform augmentation kernel and the pseudo-likelihood augmentation kernel. Figures 5.3.5a-5.3.5c contain the posterior probabilities for $\boldsymbol{\rho}$ given the observed data of top-10 rankings. The potential ranks for items 1-10 in $\boldsymbol{\rho}$ given the MCMC output are captured by the SMC algorithm with the uniform augmentation kernel approach, but the posterior probabilities vary. Despite not being able to capture the overall behaviour of the posterior obtained by MCMC, resulting in a larger MSE value and more options for potential ranks for items 1-10, the SMC algorithm with the pseudo-likelihood augmentation approach does capture the variability of the posterior for lower-ranked items compared to using the uniform augmentation kernel.

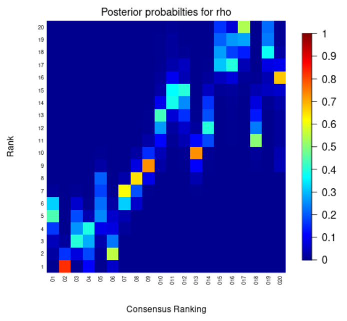
We increase the proportion of observed data in the simulated data set to see if the posterior estimates improve. Figures 5.3.5d-5.3.5f contain the posterior probabilities for $\boldsymbol{\rho}$ given the observed data of top-12 rankings and Figures 5.3.5g-5.3.5i contain the posterior probabilities for $\boldsymbol{\rho}$ given the observed data of top-15 rankings. We notice that the posterior estimate and 95% HDPI for α are now consistent across all three methods so the variability is captured. We also see an improved MSE value for the SMC algorithm with the pseudo-likelihood kernel and an increased MSE when using the uniform augmentation method. As we increase the amount of known information in each ranking, the possible ranks that each item could be in the shared consensus ranking are similar across all three methods, but there is variation in the certainty of these posterior probabilities resulting in different MSE values.



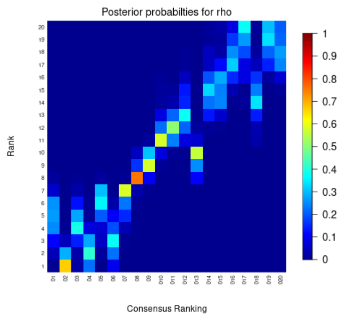
(a) MCMC: MSE = 4.93, $\hat{\alpha}$ = 2.06 (1.82, 2.27).



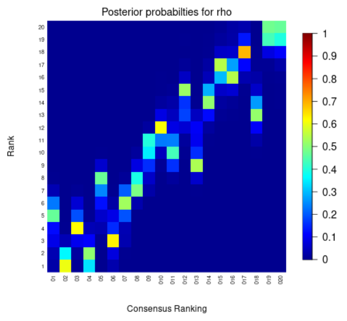
(b) SMC-U: MSE = 4.28, $\hat{\alpha}$ = 2.17 (1.89, 2.42).



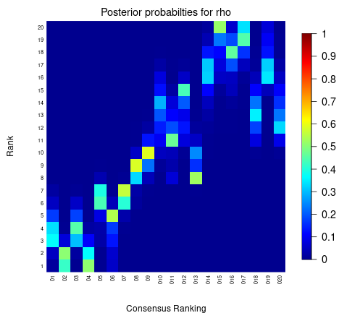
(c) SMC-P: MSE = 8.48, $\hat{\alpha}$ = 2.04 (1.80, 2.25).



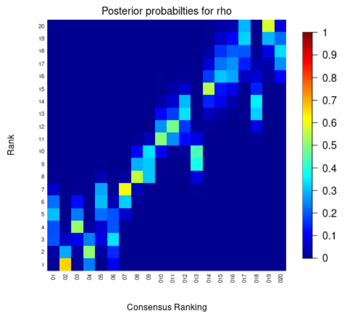
(d) MCMC: MSE = 4.63, $\hat{\alpha}$ = 2.00 (1.74, 2.18).



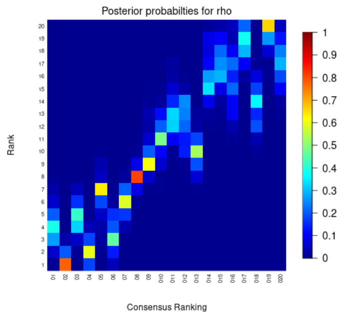
(e) SMC-U: MSE = 4.97, $\hat{\alpha}$ = 1.99 (1.75, 2.19).



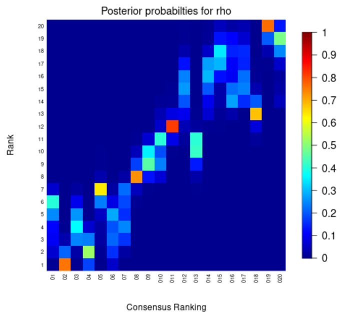
(f) SMC-P: MSE = 9.11, $\hat{\alpha}$ = 1.92 (1.72, 2.13).



(g) MCMC: MSE = 4.76, $\hat{\alpha}$ = 1.95 (1.78, 2.17).



(h) SMC-U: MSE = 5.24, $\hat{\alpha}$ = 1.96 (1.74, 2.15).



(i) SMC-P: MSE = 5.67, $\hat{\alpha}$ = 1.93 (1.72, 2.15).

Figure 5.3.5: Heat plots of the posterior probabilities, for n = 20 items, for being ranked as the k^{th} most preferred, for $k = 1, ..., 20$ using the MCMC algorithm (left), the SMC algorithm with the uniform augmentation method (middle), and the SMC algorithm with the pseudo-likelihood augmentation method (right). On the x-axis, the items are ordered according to their true consensus. The top row is obtained using the observations of the top-10 ranked items, the middle row with the top-12 ranked items and the bottom row with the top-15 ranked items. The MSE value and the mean estimate of α with its corresponding 95% HDPI are included in the captions.

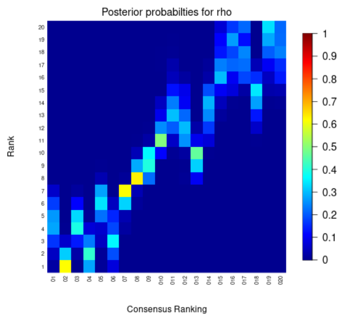
Next, we investigated the particle set in each SMC algorithm to diagnose the shortcomings of the method. Table 5.3.9 provides additional information about the particle set in each iteration of the proposed methods for each partial data set. The ESS is very low throughout; this implies that incorporating a threshold for resampling, such as ESS$< N/2$, would not improve the approximation of the posterior as we would never reach the threshold. The variance of the normalised importance weights does not differ greatly between both methods, though we would expect a smaller variance with the pseudo-likelihood approach because we are making informed proposals for the latent components of the partial rankings. The general pattern of behaviour we can observe is that the lower the particle weight variance, then the more likely we are to resample a larger number of particles. There are two instances where the number of particles resampled is a single digit when we observe the top-10 rankings and use the uniform augmentation approach, but overall we do not suffer from particle degeneracy.

			Time									
Method	**Observed Data**	**Observation**	1	2	3	4	5	6	7	8	9	10
SMC-U	Top 10	ESS	1.30	5.68	3.18	8.52	1.00	1.14	2.07	10.82	2.65	9.99
		Var(W_t)	7.72e-04	1.75e-04	3.14e-04	1.16e-04	9.96e-04	8.80e-04	4.84e-04	9.15e-05	3.77e-04	9.92e-05
		# resampled particles	6	76	65	61	3	25	75	92	51	203
SMC-P	Top 10	ESS	3.47	9.23	13.35	1.41	8.39	1.71	5.69	12.52	4.68	9.15
		Var(W_t)	2.88e-04	1.07e-04	7.40e-05	7.07e-04	1.18e-04	5.84e-04	1.75e-04	7.90e-05	2.13e-04	1.08e-04
		# resampled particles	32	75	59	41	43	31	132	121	105	137
SMC-U	Top 12	ESS	1.44	2.76	10.22	4.23	6.04	1.63	12.32	5.07	4.94	8.50
		Var(W_t)	6.95e-04	3.61e-04	9.69e-05	2.36e-04	1.65e-04	6.12e-04	8.03e-05	1.97e-04	2.02e-04	1.17e-04
		# resampled particles	12	64	83	63	56	36	114	100	56	120
SMC-P	Top 12	ESS	4.89	8.77	1.56	6.09	6.22	1.14	18.21	2.03	7.61	5.45
		Var(W_t)	2.04e-04	1.13e-04	6.41e-04	1.63e-04	1.60e-04	8.74e-04	5.40e-05	4.93e-04	1.31e-04	1.83e-04
		# resampled particles	35	100	28	72	44	19	188	47	60	104
SMC-U	Top 15	ESS	1.40	1.68	9.12	5.89	2.93	5.01	83.34	7.52	2.38	22.31
		Var(W_t)	7.15e-04	5.97e-04	1.09e-04	1.69e-04	3.40e-04	1.99e-04	1.10e-05	1.32e-04	4.19e-05	4.39e-05
		# resampled particles	26	20	80	73	33	55	281	34	56	152
SMC-P	Top 15	ESS	3.32	11.32	1.35	9.72	4.49	3.91	36.36	5.85	21.81	15.06
		Var(W_t)	3.00e-04	8.74e-05	7.41e-04	1.02e-04	2.22e-04	2.56e-04	2.65e-05	1.70e-04	4.49e-05	6.55e-05
		# resampled particles	39	52	37	149	26	28	130	72	203	66

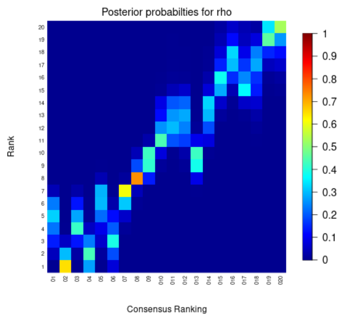
Table 5.3.9: A summary of the state of the particles in each iteration of the SMC algorithm with the uniform augmentation method and the pseudo-likelihood augmentation method for when we have observed data sets of the top-10, top-12 and top-15 rankings for $n = 20$ items. The ESS, the variance of the normalised particle weights and the number of particles resampled are provided.

The possible limitation of the proposed SMC algorithm is that it struggles to find a good augmentation for the initial set of partial rankings. For each top-10 ranking, there are 10! options for completing the latent components of each ranking. If we do not have a good set of augmented rankings when we perform the remaining stages of the SMC algorithm, we will find that few particles are resampled and we may struggle to move the particles to the areas of the posterior distribution with high density unless we use a sufficient number of applications of the MCMC move kernels. However, this implies that SMC is an ineffective method to use because we can use the MCMC sampler for less computational effort. The MCMC algorithm uses a uniform augmentation method to propose possible augmentations of each partial ranking. Despite the naive approach, it is able to obtain a good estimate of the posterior because the updates for each $\tilde{\mathbf{R}}$ are done sequentially for 10,000 iterations. In the SMC algorithm, we are effectively running 1,000 shorter chains in parallel that do not get updated sequentially for ten iterations.

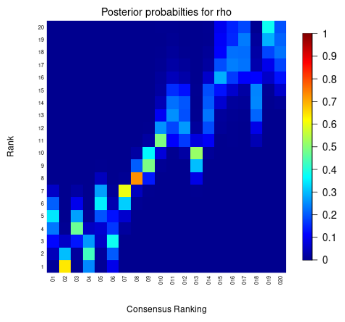
Then, we reduce the number of new observations in each SMC iteration to see if the posterior estimation improved. In each SMC iteration, we introduced one new partial ranking for 100 iterations instead of ten rankings for ten iterations. No other experiment variables were altered. The heat plots, the MSE for $\boldsymbol{\rho}$ and the summary statistics for α are presented in Figure 5.3.6. The marginal posterior probabilities for the top-10 items in $\boldsymbol{\rho}$ and α are captured across all three methods. We observe an unexpectedly high level of certainty for posterior probabilities for the two lowest-ranked items in $\boldsymbol{\rho}$ with the SMC algorithm with the uniform augmentation method. The SMC method with the pseudo-likelihood augmentation method can replicate the posterior estimation as with MCMC. Table 5.3.10 provides the results of 10 repetitions of the experiments of running both Bayesian methods where the MSE and time elapsed were recorded. We notice that if we introduced another new ranking then it would take SMC $\sim$9 and $\sim$16 seconds to run with uniform and pseudo-likelihood augmentation methods respectively whereas the MCMC algorithm would take at least $\sim$1.1 seconds to perform a new posterior estimate.



(a) MCMC: MSE = 4.93, $\hat{\alpha} = 2.06\ (1.82, 2.27)$.



(b) SMC-U: MSE = 3.87, $\hat{\alpha} = 2.07\ (1.81, 2.27)$.



(c) SMC-P: MSE = 5.32, $\hat{\alpha} = 2.06\ (1.86, 2.30)$.

Figure 5.3.6: Top: Heat plots of the posterior probabilities, for $n = 20$ items, for being ranked as the k^{th} most preferred, for $k = 1, ..., 20$ using the MCMC algorithm (left), the SMC algorithm with the uniform augmentation method (middle), and the SMC algorithm with the pseudo-likelihood augmentation method (right). On the x-axis, the items are ordered according to their true consensus. The MSE value and the mean estimate of α with its corresponding 95% HDPI are included in the captions.

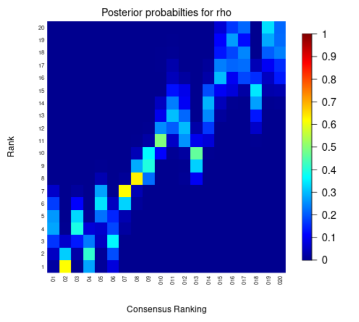
	MCMC		**SMC-U**		**SMC-P**	
Method	Mean	95% CI	Mean	95% CI	Mean	95% CI
MSE	5.09	(4.89, 5.29)	4.10	(3.99, 4.21)	5.20	(5.08, 5.31)
Time (s)	1.08	(1.07, 1.08)	911.38	(908.74, 914.01)	1621.52	(1617.31, 1625.73)

Table 5.3.10: Summary table of the MSE and time taken for the ten experiment runs of the MCMC algorithm and the SMC algorithm with both augmentation methods for the partially observed simulated data set of $n = 20$ items.

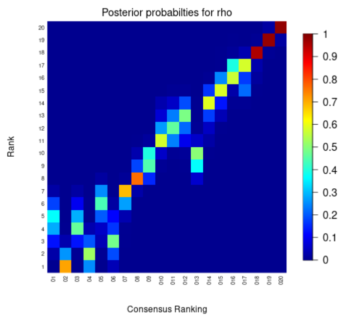
For the final test with the simulated data set, we reduced the number of MCMC move kernel applications in each SMC iteration. We applied: five MCMC iterations for $\boldsymbol{\rho}$; one MCMC iteration to sample α; and one MCMC iteration to sample new augmentations for $\mathbf{R}_{1:M_t}$ for each particle to reduce the computational effort. The results of the experiment are presented in Figure 5.3.7 and Table 5.3.11.

The SMC algorithm with the uniform augmentation method shows a higher certainty for the lowest three ranked items in $\boldsymbol{\rho}$ in Figure 5.3.7b. This contributes to a significantly lower MSE value and a higher value for the posterior estimate for α.

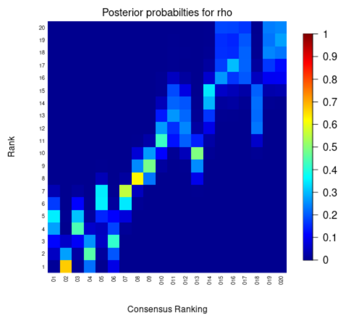
Though the MSE value for the SMC algorithm with the pseudo-likelihood augmentation method in Figure 5.3.7c is higher than that of the MCMC method in Figure 5.3.7a, the heat plot shows that the overall level of uncertainty for each item potential rank in $\boldsymbol{\rho}$ appear relatively similar, particularly for the top-15 ranked items. Table 5.3.11 shows that the computational time for each SMC algorithm has reduced to about one-ninth of the previous experiment times in Table 5.3.10. One sequential posterior update, based on observing a new partial ranking, would now take approximately $\sim$1.1 and $\sim$1.8 seconds respectively with each augmentation method.



(a) MCMC: MSE = 4.93, $\hat{\alpha} = 2.06\ (1.82, 2.27)$.



(b) SMC-U: MSE = 2.79, $\hat{\alpha} = 2.40\ (2.10, 2.64)$.



(c) SMC-P: MSE = 5.48, $\hat{\alpha} = 2.04\ (1.84, 2.28)$.

Figure 5.3.7: Top: Heat plots of the posterior probabilities, for $n = 20$ potatoes, for being ranked as the k^{th} most preferred, for $k = 1, ..., 20$ using the MCMC algorithm (left), the SMC algorithm with the uniform augmentation method and reduced computational effort (middle), and the SMC algorithm with the pseudo-likelihood augmentation method (right) and reduced computational effort. On the x-axis, the items are ordered according to their true consensus. The MSE value and the mean estimate of α with its corresponding 95% HDPI are included in the captions.

5.3.4 Real data: Potato data

Finally, we studied a modified version of the Potato data set (Liu et al., 2019a), which presents $M = 12$ rankings of the top-10 ranked items out of a possible $n = 20$

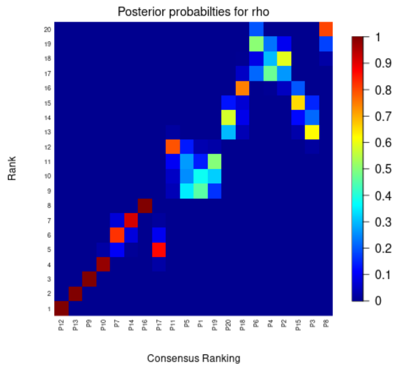
	MCMC		**SMC-U**		**SMC-P**	
Method	Mean	95% CI	Mean	95% CI	Mean	95% CI
MSE	5.09	(4.89, 5.29)	3.07	(2.93, 3.21)	5.31	(5.07, 5.55)
Time (s)	1.08	(1.07, 1.08)	109.10	(108.69, 109.51)	182.69	(182.04, 183.33)

Table 5.3.11: Summary table of the MSE and time taken for the ten experiment runs of the MCMC algorithm and the SMC algorithm with both augmentation methods and reduced computational effort for the partially observed simulated data set of $n = 20$ items.

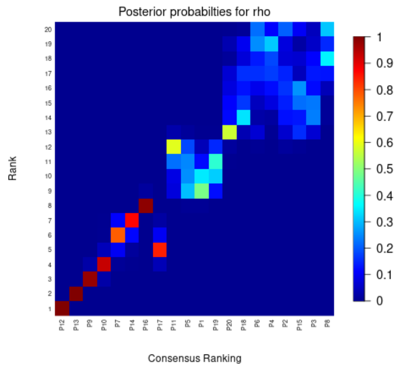
items. For this experiment the proposal distribution for the scale parameter was set to $\sigma_\alpha = 0.15$. Each SMC iteration introduced one new ranking, propagated $N = 1000$ particles, and performed ten applications of the MCMC move kernels for $\boldsymbol{\rho}$, α and $\tilde{\mathbf{R}}_{1:M_t}$.

Figure 5.3.8 displays the marginal posterior distributions of $\boldsymbol{\rho}$ and α with all three methods. The MSE and the HDPIs for α are also provided in the captions. The heat plot in Figure 5.3.8a, which was obtained using an MCMC chain of 5,000 iterations (after burn-in), appears to have not converged: it appears to display a high level of certainty for the lowest ranked items despite that no known rankings for these items are observed in the original data set. A longer MCMC chain was run with the same modified data set for 100,000 iterations with the first 50,000 discarded as burn-in. This result is presented in Figure 5.3.8b. The marginal posterior probabilities for the top-8 heaviest potatoes are captured across all three methods. For lower-ranked items, there is more variation in the marginal posterior probabilities. The heat plot obtained by SMC with the uniform augmentation method in Figure 5.3.8c is not able to capture the general behaviours of the remaining items in $\boldsymbol{\rho}$ as that obtained by MCMC in Figure 5.3.8b; it has not converged to the posterior obtained by MCMC. The heat plot obtained by the pseudo-likelihood augmentation in Figure 5.3.8d captures the general uncertainty about the possible ranks of the middle-ranked items (P11, P5, P1, P19), but cannot replicate the same behaviour as MCMC for the lower-ranked items in $\boldsymbol{\rho}$. This observation is supported by the estimation of the CP consensus in Table 5.3.12.

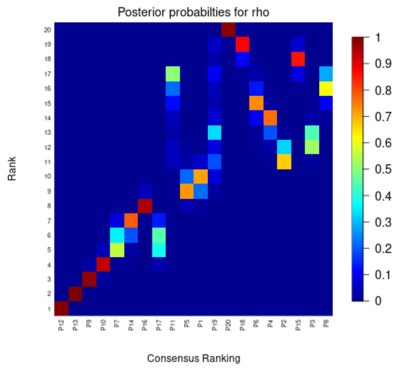
We investigated the output of each iteration of SMC with each augmentation method in a similar manner as described in Section 5.3.3. Table 5.3.13 provides a record of the ESS, the variation of the normalised particle weights and the number of particles resampled at each iteration. The ESS is consistently higher throughout when using the SMC algorithm with the pseudo-likelihood augmentation method, but it is still not high enough to incorporate a threshold for resampling unless we specify a threshold of ESS $< N/4$. The variance of the normalised importance weights does not differ greatly between both methods, though we still observe the general pattern of behaviour of a lower particle weight variance likely results in a larger number of particles being resampled. There is no case where the SMC algorithm suffers from particle degeneracy. We repeated the experiments ten times and recorded the total time to run the MCMC algorithm with the full data set and the SMC algorithms, where we introduce one new ranking at a time, in Table 5.3.14. If we were to introduce another new ranking then the SMC algorithms would take $\sim$1.8 and $\sim$3.1 seconds respectively whereas the MCMC algorithm would have to run for at least $\sim$2.7 seconds to run and update the posterior estimate. Since the data set is small and the number of MCMC move kernel applications that are applied to the particle set in one iteration of SMC is about one-tenth of the number of MCMC iterations used to run the longer MCMC chain. We also find that the SMC algorithm runs for a similar amount of computation time with fewer MCMC iterations. We conclude that the pseudo-likelihood augmentation method has been able to make reasonable proposals for the latent components of each new observed partial ranking and it can maintain a good representative sample of the posterior distribution. The uniform augmentation method, on the other hand, makes initial bad proposals and struggles to keep up with the sequential updates when incorporating new rankings.



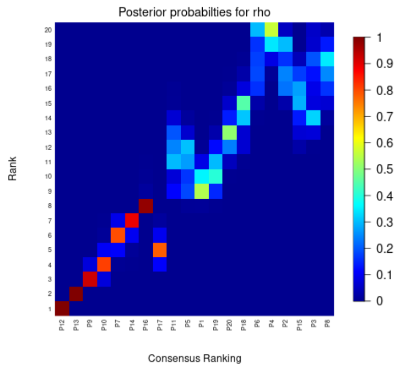
(a) MCMC: MSE = 4.53, $\hat{\alpha}$ = 10.30 (8.72, 11.74).



(b) MCMC: MSE = 4.26, $\hat{\alpha}$ = 10.02 (8.25, 11.77).



(c) SMC-U: MSE = 11.88, $\hat{\alpha}$ = 8.58 (7.38, 10.02).



(d) SMC-P: MSE = 4.49, $\hat{\alpha}$ = 10.01 (8.36, 11.63).

Figure 5.3.8: Top: Heat plots of the posterior probabilities, for $n = 20$ potatoes, for being ranked as the k^{th} most preferred, for $k = 1, ..., 20$ using the MCMC algorithm (top) with 10,000 (5,000 burn-in, left) and 100,000 (50,000 burn-in, right) iterations, the SMC algorithm (bottom) with the uniform augmentation method (left) and the pseudo-likelihood augmentation method (right). On the x-axis, the items are ordered according to their true consensus. The MSE value and the mean estimate of α with its corresponding 95% HDPI are included in the captions.

5.4 Conclusion

In this chapter, we extended the resample-move SMC algorithm for the Mallows model to make sequential updates of the estimated posterior distribution in an online

	CP			$p(\boldsymbol{\rho}_i \leq i)$			95% **HDPI**		
$\boldsymbol{\rho}$	**MCMC**	**SMC-U**	**SMC-P**	**MCMC**	**SMC-U**	**SMC-P**	**MCMC**	**SMC-U**	**SMC-P**
ρ_1	P12	P12	P12	1.00	1.00	1.00	[1]	[1]	[1]
ρ_2	P13	P13	P13	1.00	1.00	1.00	[2]	[2]	[2]
ρ_3	P9	P9	P9	0.98	0.98	0.93	[3]	[3]	[3,4]
ρ_4	P10	P10	P10	0.95	0.96	0.89	[4,5]	[4,5]	[3,5]
ρ_5	P17	P7	P17	0.88	0.57	0.89	[4,6]	[5,7]	[4,6]
ρ_6	P7	P17	P7	0.89	0.87	0.91	[5,7]	[5,7]	[5,7]
ρ_7	P14	P14	P14	1.00	1.00	1.00	[6,7]	[6,7]	[6,7]
ρ_8	P16	P16	P16	0.99	0.96	0.98	[8]	[8]	[8]
ρ_9	P1	P5	P1	0.49	0.75	0.55	[9,11]	[8,10]	[9,11]
ρ_{10}	P5	P1	P19	0.56	0.94	0.55	[9,12]	[9,11]	[9,12]
ρ_{11}	P19	P2	P5	0.85	0.67	0.62	[9,12]	[11,12]	[9,13]
ρ_{12}	P11	P3	P11	0.97	0.54	0.73	[9,12]	[9,14]	[9,14]
ρ_{13}	P20	P19	P20	0.58	0.68	0.79	[13,17],[19]	[12,13]	[11,15]
ρ_{14}	P18	P4	P18	0.37	0.98	0.44	[14,19]	[13,14]	[12,16]
ρ_{15}	P3	P6	P15	0.53	0.84	0.51	[13,19]	[14,16]	[13,18]
ρ_{16}	P15	P8	P3	0.77	0.72	0.56	[13,18],[20]	[15,17]	[13,20]
ρ_{17}	P2	P11	P2	0.65	1.00	0.49	[13,20]	[11,12], [14,17]	[16,20]
ρ_{18}	P4	P15	P8	0.58	0.94	0.84	[15,20]	[17,19]	[15,19]
ρ_{19}	P6	P18	P6	0.78	0.99	0.70	[13],[15,20]	[18,19]	[16,20]
ρ_{20}	P8	P20	P4	1.00	1.00	1.00	[16,20]	[20]	[18,20]

Table 5.3.12: Posterior estimates of the items in the modified Potato data set. Items are arranged according to the CP consensus ranking with their corresponding cumulative probabilities and 95% HDPIs.

		Iteration											
Method	**Observation**	1	2	3	4	5	6	7	8	9	10	11	12
SMC-U	ESS	346.01	202.37	69.33	37.45	56.26	2.62	51.41	67.80	113.18	43.32	79.44	80.90
	Var(W_t)	1.89e-06	3.95e-06	1.34e-05	2.57e-06	1.68e-05	3.81e-04	1.85e-05	1.38e-05	7.84e-06	2.21e-05	1.16e-05	1.14e-05
	# resampled particles	510	464	318	195	301	139	284	353	318	266	435	286
SMC-P	ESS	397.36	47.03	114.24	84.13	56.02	88.51	13.45	38.67	73.89	13.71	97.00	122.96
	Var(W_t)	1.52-06	2.03e-05	7.76e-06	1.09e-05	1.69e-05	1.03e-05	7.34e-05	2.49e-05	1.25e-05	3.22e-05	9.32e-06	7.14e-06
	# resampled particles	516	386	351	310	260	262	167	237	260	280	320	327

Table 5.3.13: A summary of the state of the particles in each iteration of the SMC algorithm with the uniform augmentation method and the pseudo-likelihood augmentation method with the modified Potato data set. The ESS, the variance of the normalised particle weights and the number of particles resampled are provided.

setting when we receive partially observed data. We assume that we observe new partial rankings from individuals and these remain unchanged over time.

We considered two proposal distributions to augment the missing components of each observed partial ranking before performing the reweighting stage of the SMC framework. The uniform augmentation method samples the latent components of the partial ranking uniformly at random with the allowable augmentations of the missing ranks. The pseudo-likelihood augmentation method makes an informed choice of the

	MCMC		**SMC-U**		**SMC-P**	
Method	Mean	95% CI	Mean	95% CI	Mean	95% CI
MSE	3.76	(3.30, 4.22)	10.44	(9.99, 10.88)	3.96	(3.36, 4.56)
Time (s)	2.97	(2.95, 2.99)	18.71	(18.48, 18.93)	31.00	(30.59, 31.32)

Table 5.3.14: Summary table of the MSE and time taken for the ten experiment runs of the MCMC algorithm and SMC algorithm with both augmentation methods with the modified Potato data set.

augmentation using the current estimated values of the Mallows model parameters. It samples each missing rank for each unranked item using several univariate Mallows distributions each conditioned on the item's rank in the estimated consensus ranking, the scale parameter and the set of remaining possible ranks. However, this particular method can only be utilised if the distance between rankings is measured with either the footrule or the Spearman distance.

We tested the proposed SMC algorithm for the Mallows model with partial rankings using simulated and real data to study the overall behaviour of the SMC algorithm against the MCMC algorithm for the cases considered. We discovered that for small examples where the size of the sampling space for the latent components of each partial ranking is small, the SMC algorithm with the pseudo-likelihood augmentation method is able to estimate the posterior given a set of partial rankings. However, when the number of missing components in each item is increased in the experiments related to new partial rankings, we notice that the estimation deteriorates. We conclude that the main reason that the SMC sampler struggles with larger examples is that the augmentation methods can struggle to make proposals that reflect the current posterior with a set of partial data. This means that potentially an insufficient number of particles are resampled or that we may struggle to propagate the particles to areas of the posterior distribution with high density without a heavy reliance on the MCMC move kernels to help mitigate this. Although the MCMC algorithm proposed possible augmentations for the latent components of the partial rankings using a uniform augmentation method as well, the algorithm also proposes these updates for each $\tilde{\mathbf{R}}$ sequentially for the same number of times as the length of the MCMC chain. The SMC algorithm suffers from running N chains in parallel, which do not necessarily update each $\tilde{\mathbf{R}}$ as often as MCMC before they are potentially discarded through resampling. The posterior estimation with the SMC methodology is improved when we reduce the number of new rankings to account for in each update, as this reduces the number of latent components to consider in the particle set.

Chapter 6

Sequential Monte Carlo for the Mallows Model with Updated Partial Rankings

6.1 Introduction

In Chapter 5, we developed an initial framework to perform SMC with the Mallows model given a collection of partial rankings on a discrete timeline, making the assumption that we could only observe new partial rankings and any previously observed rankings would remain unchanged. Now, we assume that for a given set of partial rankings, we may find that each assessor in the data set may want to provide an update to their existing information at some later stage in time. This means that the current SMC framework needs to be altered to account for these changes. The challenge is to "correct" the latent components of the ranking data in the particle set such that they are consistent with the new observed components of the updated partial rankings over time. We note that this cannot be termed a full correction since we may find that the latent components of each ranking may contradict the new observed information in future iterations.

The structure of this chapter is as follows. We formulate the updated partial ranking problem from existing individuals over a discrete time sequence in Section 6.2. The methodology is presented in Section 6.2.1, and then assessed in Section 6.3 with simulated and real data sets.

6.2 Proposed method

The context remains the same as in Section 5.2 of Chapter 5, except we now assume that we have observed a total of M partial rankings. The task is to estimate the sequence of Mallows posterior distributions for the evolving M partial rankings over a discrete timeline $t = 1, \dots, T$, defined as

$$\pi_t(\boldsymbol{\rho}, \alpha | \mathbf{R}_1, \dots, \mathbf{R}_M) = \sum_{\tilde{\mathbf{R}}_1 \in \mathcal{S}_1} \cdots \sum_{\tilde{\mathbf{R}}_M \in \mathcal{S}_M} p(\boldsymbol{\rho}, \alpha, \tilde{\mathbf{R}}_1, \dots, \tilde{\mathbf{R}}_M | \mathbf{R}_1, \dots, \mathbf{R}_M).$$

Each partial ranking has had its missing components filled in via data augmentation during the SMC process. We assume that we receive updates from known assessors in the form of a change to their given partial ranking. Specifically, the update we observe is a rank for an item that was previously not ranked by the known assessor. The previously observed elements of each ranking remain unchanged. For example, we may have observed a ranking

$$\mathbf{R} = (5, 3, \cdot, \cdot, \cdot).$$

from an assessor. Using SMC, we may predict the complete auxiliary ranking is

$$\tilde{\mathbf{R}} = (5, 3, 1, 2, 4).$$

However, at a future point in time, we observe some new additional information about the assessor's ranking, for example,

$$\mathbf{R} = (5, 3, 2, \cdot, \cdot),$$

which contradicts the predicted full ranking at the previous time step. This is a contrived example; it is very unlikely that we will have guessed every single latent item rank correctly during the augmentation stage, assuming that we will observe the true latent ranks in the future. This is difficult within the SMC framework because we cannot perform the remaining stages of the algorithm until we address the problem of correcting the auxiliary components of the augmented rankings to be compatible with the new observed components of each partial ranking. The task is to reweight the particles to account for the updated partial rankings from known assessors such that we are able to perform the other stages of the SMC process and obtain an updated estimation of the posterior sequentially.

6.2.1 Methodology

We assume that a total of M partial rankings have been observed over $t-1$ iterations. The missing components of each ranking are augmented to achieve $\tilde{\mathbf{R}}_{1:M,t-1}$ which form part of the particle set containing a representative sample of the Mallows posterior $\pi_{t-1}(\boldsymbol{\rho}, \alpha | \mathbf{R}_{1:M,t-1})$. However, at some future time step, say t, we observe $\mathbf{R}_{j,t},\ j \in \{1, \dots, M\}$ which provides an update about an assessor's ranking $\mathbf{R}_j$: we observe a rank for an item that was previously unobserved. Any previously observed elements in any ranking remain unchanged as we new information is observed over time. However, this additional information also means that the observed ranks for items in $\mathbf{R}_{j,t}$ may not be coherent with the auxiliary components of the item ranks in $\tilde{\mathbf{R}}_{j,t-1}$.

For each updated ranking received, the latent components of the partial rankings are corrected to be consistent with the observed components of the updated partial rankings. Here, each updated ranking is viewed as a known assessor being removed from the observed data set and re-entering as a new assessor with their new partial ranking. Formally, the ranking $\tilde{\mathbf{R}}_{j,t-1}$ is removed just before time t before they present their updated preferences. This result is defined as $\tilde{\mathbf{R}}_{1:M\setminus\{j\},t-1}$ to account for the removal of $\tilde{\mathbf{R}}_{j,t-1}$. The weight adjustment for each particle, $\boldsymbol{\theta}^{(i)},\ i \in \{1, \dots, N\}$, at this stage within the SMC framework is

$$\begin{aligned}\frac{\pi(\boldsymbol{\theta}_{t-1}^{(i)}, \tilde{\mathbf{R}}_{1:M\setminus\{j\},t-1}^{(i)}|\mathbf{R}_{1:M\setminus\{j\},t-1})}{\pi(\boldsymbol{\theta}_{t-1}^{(i)}, \tilde{\mathbf{R}}_{1:M,t-1}^{(i)}|\mathbf{R}_{1:M,t-1})} &= \frac{\pi(\boldsymbol{\theta}_{t-1}^{(i)})p(\tilde{\mathbf{R}}_{1:M\setminus\{j\},t-1}^{(i)}|\boldsymbol{\theta}_{t-1}^{(i)}, \mathbf{R}_{1:M\setminus\{j\},t-1})}{\pi(\boldsymbol{\theta}_{t-1}^{(i)})p(\tilde{\mathbf{R}}_{1:M,t-1}^{(i)}|\boldsymbol{\theta}_{t-1}^{(i)}, \mathbf{R}_{1:M,t-1})} \\ &\quad \times \frac{p(\mathbf{R}_{1:M,t-1})}{p(\mathbf{R}_{1:M\setminus\{j\},t-1})} \\ &= \frac{p(\mathbf{R}_{j,t-1}|\mathbf{R}_{1:M\setminus\{j\},t-1})}{p(\tilde{\mathbf{R}}_{j,t-1}^{(i)}|\boldsymbol{\theta}_{1:t-1}^{(i)}, \mathbf{R}_{j,t-1})} \\ &\propto \frac{1}{p(\tilde{\mathbf{R}}_{j,t-1}^{(i)}|\boldsymbol{\theta}_{t-1}^{(i)}, \mathbf{R}_{j,t-1})}. \end{aligned} \tag{6.2.1}$$

Then, at time t, $\mathbf{R}_{j,t}$ enters the system with its updated partial ranking. Assuming the assessor's new ranking is independent of the existing rankings, the weight update for the next stage is

$$\frac{\pi(\boldsymbol{\theta}_{1:t}^{(i)}, \tilde{\mathbf{R}}_{1:M,t}^{(i)}|\mathbf{R}_{1:M,t})}{\pi(\boldsymbol{\theta}_{1:t}^{(i)}, \tilde{\mathbf{R}}_{1:M\setminus\{j\},t}^{(i)}|\mathbf{R}_{1:M\setminus\{j\},t})q(\tilde{\mathbf{R}}_{j,t}^{(i)}|\boldsymbol{\theta}_{1:t}^{(i)}, \mathbf{R}_{j,t})} \propto \frac{p(\tilde{\mathbf{R}}_{j,t}^{(i)}|\boldsymbol{\theta}_{t}^{(i)}, \mathbf{R}_{j,t})}{q(\tilde{\mathbf{R}}_{j,t}^{(i)}|\boldsymbol{\theta}_{t}^{(i)}, \mathbf{R}_{j,t})}, \tag{6.2.2}$$

where $q(\tilde{\mathbf{R}}_{j,t}^{(i)}|\boldsymbol{\theta}_{t}^{(i)}, \mathbf{R}_{j,t})$ is the proposed augmentation kernel. The total incremental particle weight update at time t is therefore the product of (6.2.1) and (6.2.2),

$$\tilde{w}_t(\boldsymbol{\theta}_t^{(i)}, \tilde{\mathbf{R}}_{1:M}^{(i)}, t) = \frac{p(\tilde{\mathbf{R}}_{j,t}^{(i)}|\boldsymbol{\theta}_{t}^{(i)}, \mathbf{R}_{j,t})}{p(\tilde{\mathbf{R}}_{j,t-1}^{(i)}|\boldsymbol{\theta}_{t-1}^{(i)}, \mathbf{R}_{j,t-1})q(\tilde{\mathbf{R}}_{j,t}^{(i)}|\boldsymbol{\theta}_{1:t}^{(i)}, \mathbf{R}_{j,t})}.$$

After the reweighting stage, we can perform the remaining stages of the SMC algorithm. The pseudocode for the SMC algorithm for the Mallows model with updated partial rankings is given in Algorithm 13.

Algorithm 13: Sequential Monte Carlo for the Mallows Model with Updated Partial Rankings

```
Input: $\lambda,\ \sigma_\alpha,\ d(\cdot,\cdot),\ K,\ L,\ N,\ \mathbf{R}_1,\dots,\mathbf{R}_M;\ T,\ Z_n(\alpha)$.
Output: Posterior distributions of $\boldsymbol{\rho}, \alpha$ and $\tilde{\mathbf{R}}_{1:M}$.
Initialisation of SMC: generate particle set from MCMC or SMC
  $\boldsymbol{\rho}_0^{(i)}, \alpha_0^{(i)}, \tilde{\mathbf{R}}_{1:M,t}$ for $i = 1,\dots,N$.
for $t = 1,\dots,T$ do
    Observe $\mathbf{R}_{1:M,t}$
    for $i = 1 : N$ do
        for $j = 1 : M$ do
            if $\mathbf{R}_{j,t}$ not compatible with $\tilde{\mathbf{R}}_{j,t-1}$ then
                Sample $\tilde{\mathbf{R}}_{j,t}^{(i)}$ with proposal $\tilde{\mathbf{R}}_{j,t} \sim q(\tilde{\mathbf{R}}_{j,t}|\mathbf{R}_{j,t}, \boldsymbol{\rho}_t^{(i)}, \alpha_t^{(i)})$.;
            else
                $\tilde{\mathbf{R}}_{j,t}^{(i)} = \tilde{\mathbf{R}}_{j,t-1}^{(i)}$ and $q(\tilde{\mathbf{R}}_{j,t}|\mathbf{R}_{j,t}, \boldsymbol{\rho}_t^{(i)}, \alpha_t^{(i)}) = 1$.
            end
        end
        Compute $\tilde{w}_t = \prod_{j=1}^{M_t} \frac{p(\tilde{\mathbf{R}}_{j,t}|\boldsymbol{\rho}_t,\alpha_t)}{p(\tilde{\mathbf{R}}_{j,t-1}|\boldsymbol{\rho}_{t-1},\alpha_{t-1},\mathbf{R}_{j,t-1})q(\tilde{\mathbf{R}}_{j,t}|\boldsymbol{\rho}_{t-1},\alpha_{t-1},\mathbf{R}_{j,t})}$.
    end
    Resample $(k_1,\dots,k_N) \sim \mathcal{M}(w_t^{(1)},\dots,w_t^{(N)})$ and set
      $\{\boldsymbol{\theta}_t^{(1:N)}, w_t^{(i)}\} \leftarrow \{\boldsymbol{\theta}_t^{(k_1:k_N)}, \frac{1}{N}\}$.
    for $i = 1 : N$ do
        for $k = 1,\dots,K$ do
            M-H step: update $\boldsymbol{\rho}_t^{(i)}$ using the leap-and-shift proposal
              ($\boldsymbol{\rho'} \sim$ L&S$(\boldsymbol{\rho}_t^{(i)}, L)$).
            M-H step: update $\alpha_t^{(i)}$ with log-normal proposal ($\alpha' \sim \log\mathcal{N}(\alpha_t^{(i)}, \sigma_a^2)$).
            for $j = 1 : M$ do
                M-H step: update $\tilde{\mathbf{R}}_{j,t}^{(i)}$ with proposal $\tilde{\mathbf{R}}'_{j,t} \sim q(\tilde{\mathbf{R}}_j|\mathbf{R}_{j,t}, \boldsymbol{\rho}_t^{(i)}, \alpha_t^{(i)})$.
            end
        end
    end
end
```

6.3 Experimental analyses

We assessed the proposed SMC methodology with synthetic and real data. We note that each experiment with the SMC algorithm with the uniform augmentation method is referred to as SMC-U in each figure and table in this chapter, whilst the SMC algorithm with the pseudo-likelihood augmentation method is referred to as SMC-P.

6.3.1 Simulated data: 10 items

We generated $M = 100$ full rankings of $n = 10$ items and filtered the data set such that we observe only the top-5 ranked items. The SMC algorithm is initialised with this data set by re-running the experiment described in Section 5.3.1 in Chapter 5 with $N = 1000$ particles to obtain an initial estimate of the posterior. We receive the next top-ranked item from ten assessors at a time in decreasing rank order from 6 to 9 for a total of $T = 40$ iterations. In each iteration, we correct each augmented ranking if it is not consistent with the newly observed components of each assessor's partial ranking. We consider the uniform and pseudo-likelihood augmentation methods to augment the partial rankings which will affect the calculations for the particle weights given the proposed correction. Finally, we resampled and applied the MCMC move kernels for $\boldsymbol{\rho}, \alpha$ and $\tilde{\mathbf{R}}_{1:M}$ for ten iterations at each SMC iteration. In total, 10,000 applications of MCMC move kernels were applied to each parameter in the particle set in each iteration. The MCMC algorithm cannot update the chain to account for any updated partial rankings, so as a reference of comparison with the results obtained from SMC, we ran the MCMC algorithm several times, each with the top-k, $k = 5, \ldots, 9$, rankings in the simulated data set for 10,000 iterations and discarded the first 5,000 as burn-in.

The results from the proposed SMC methodology with both augmentation methods and the MCMC method are compared by observing the heat plot, the MSE for $\boldsymbol{\rho}$, the mean estimate and the HDPI of α in Figure 6.3.1. The first column of heat plots shows the initial marginal posterior distributions for $\boldsymbol{\rho}$ when we observe the top-5 rankings. These results were also shown in Figure 5.3.1 and discussed in Section 5.3.1 in Chapter 5. We see that the SMC method with both augmentation methods provides similar estimated posterior distributions as the one obtained by MCMC. There are some discrepancies in the posterior probabilities for certain items in $\boldsymbol{\rho}^*$ in the heat plots, but the suggestions about which rank to assign to each item in the consensus look relatively similar across all three methods.

Table 6.3.1 presents the average MSE and time elapsed to run ten repetitions of the MCMC algorithm with each top-k, $k = 5, \ldots, 9$, data set and the SMC algorithm with both augmentation methods when we reach the iterations where we observe the top-k, $k = 6, \ldots, 9$, ranked items for all assessors in the data set. The time taken to run the MCMC algorithm decreases as we observe more information about the full data set. In SMC, to perform one sequential update of ten updated rankings, the algorithm takes $\sim$14 seconds with the uniform and $\sim$20 seconds with the pseudo-likelihood augmentation method respectively.

Next, we repeat the experiment with a reduced number of MCMC move kernel applications to assess whether SMC can perform sequential posterior estimations for less computational effort well in comparison to using MCMC. We specified five applications of the MCMC move kernels for each parameter in each iteration of SMC, so a total of 5,000 MCMC move kernels were applied to each particle. The results of this experiment are presented in Figure 6.3.2. After we have observed the top-6 rankings from each individual, we notice that more uncertainty is shown for the lower-ranked items in $\boldsymbol{\rho}$ when using the SMC algorithm with the uniform augmentation method in

Method	**Observation**	**Metric**	**Mean**	**95% CI**
MCMC	Top-5	Time (s)	0.55	(0.55, 0.55)
		MSE	0.32	0.30, 0.34)
	Top-6	Time (s)	0.53	(0.52, 0.53)
		MSE	0.23	(0.17, 0.29)
	Top-7	Time (s)	0.51	(0.50, 0.51)
		MSE	0.19	(0.18, 0.20)
	Top-8	Time (s)	0.48	(0.48, 0.49)
		MSE	0.18	(0.17, 0.19)
	Top-9	Time (s)	0.24	(0.24, 0.25)
		MSE	0.17	(0.15, 0.18)
SMC-U	Top-6	Time (s)	138.63	(137.97, 139.30)
		MSE	0.34	(0.27, 0.40)
	Top-7	Time (s)	276.14	(275.33, 276.95)
		MSE	0.23	(0.17, 0.28)
	Top-8	Time (s)	411.81	(410.47, 413.14)
		MSE	0.17	(0.16, 0.17)
	Top-9	Time (s)	545.40	(544.38, 546.42)
		MSE	0.16	(0.15, 0.16)
SMC-P	Top-6	Time (s)	210.97	(210.04, 211.89)
		MSE	0.29	(0.20, 0.38)
	Top-7	Time (s)	410.28	(408.81, 411.75)
		MSE	0.21	(0.17, 0.24)
	Top-8	Time (s)	597.57	(596.18, 598.96)
		MSE	0.18	(0.17, 0.19)
	Top-9	Time (s)	761.45	(759.23, 763.66)
		MSE	0.16	(0.15,0.17)

Table 6.3.1: Summary table of the MSE and time taken for ten experiment runs of the MCMC algorithm with the top-k, $k = 5, \dots, 9$, rankings of the complete simulated data set of ten items and SMC algorithm with the uniform (SMC-U) and the pseudo-likelihood (SMC-P) augmentation methods when we reach the iterations where we have observed the top-k, $k = 6, \dots, 9$, rankings for all assessors.

Figure 6.3.2g. Greater certainty in posterior probabilities for some items to be allocated certain ranks in $\boldsymbol{\rho}$ with the SMC algorithm with the pseudo-likelihood augmentation method can be observed in Figure 6.3.2l. The marginal posterior probabilities for the top-4 ranked items in $\boldsymbol{\rho}$ appear to be similar across all three methods in the heat plots. In addition, the point estimate and 95% HDPIs for the scale parameter

remain relatively similar across all three methods. After observing the top-6 rankings, the SMC algorithm with the pseudo-likelihood augmentation method can obtain similar MCMC posterior estimates as MCMC by the time we have observed the top-7 rankings, whilst SMC with the uniform augmentation method can also similar MCMC posterior estimates as MCMC by the time we have observed the top-8 rankings.

6.3.2 Real data: Sushi data

We tested the SMC algorithm with a modified Sushi data set (Kamishima, 2003) of the first $M = 100$ rankings and used the same experimental set-up as the first experiment described in Section 6.3.1.

The results from this experiment are presented in Figure 6.3.3. The top row shows the output of the MCMC algorithm for each model run with the top-5,$\ldots$, top-9 rankings of the modified data set. The middle and bottom rows show the initial marginal posterior distributions for $\boldsymbol{\rho}$, which are obtained by the output of SMC with both augmentation methods when we observe the top-5 rankings in the modified data set. The analysis for this initial phase is discussed in Section 5.3.2 of Chapter 5. The remaining heat plots in Figure 6.3.3 show the estimated marginal posterior distributions of the Mallows model parameters after we have reached certain iterations in the SMC algorithm where we have observed the top-k, $k = 6, \ldots, 9$, rankings for all assessors. We notice that the trajectory of the estimation for α remains consistent across all three methods by observing the captions in each heat plot in Figure 6.3.3. The possible ranks for items in $\boldsymbol{\rho}^*$ with non-zero posterior probability are similar between the MCMC and SMC methods, but there are discrepancies in these probabilities, particularly for SMC with the uniform augmentation method. With this method, we also note that the MSE increases suddenly when we observe the top-8 rankings in Figure 6.3.3i; this is a result of some posterior probability being allocated to items that are not close to the $\boldsymbol{\rho}^*$. When we have observed the top-9 rankings we effectively know what the full data set would look like, so the marginal posteriors for $\boldsymbol{\rho}$ and α are the same across all three methods. Overall, the SMC method with the pseudo-likelihood augmentation method has achieved sequential marginal posterior estimations of $\boldsymbol{\rho}$ that are closer to the results obtained by MCMC. To perform an update with ten updated partial rankings in one iteration of SMC it can be inferred from Table 6.3.2 that this takes approximately 14 and 20 seconds with the uniform and pseudo-likelihood augmentation methods respectively whilst it takes a fraction of a second to run MCMC with each updated data set. We conclude from this experiment that the SMC methods do not perform faster than the MCMC algorithm. The computational cost, in terms of MCMC move kernels, to perform one iteration of SMC is the same as the number of iterations to run MCMC sampler.

We repeat this experiment to see whether the proposed SMC algorithm could replicate the posterior estimates obtained using MCMC with less computational effort. We reduce the number of MCMC move kernel applications across all variables and parameters in the particle set from ten to five in each iteration of SMC giving a total of 5,000 move kernels applied to each parameter in the particle set. The results are presented in Figure 6.3.4 and we note that the results obtained from MCMC are repeated from Figure 6.3.3. The potential selection of ranks for items in $\boldsymbol{\rho}$ are captured by both SMC methods, but the posterior probabilities vary greatly. Items

Method	**Observation**	**Metric**	**Mean**	**95% CI**
MCMC	Top-5	Time (s)	0.56	(0.56, 0.56)
		MSE	1.09	(1.01, 1.16)
	Top-6	Time (s)	0.53	(0.53, 0.53)
		MSE	0.96	(0.90, 1.02)
	Top-7	Time (s)	0.51	(0.51, 0.51)
		MSE	0.65	(0.62, 0.69)
	Top-8	Time (s)	0.49	(0.48, 0.49)
		MSE	0.65	(0.60, 0.71)
	Top-9	Time (s)	0.26	(0.24, 0.25)
		MSE	0.67	(0.62, 0.72)
SMC-U	Top-6	Time (s)	138.90	(138.32, 139.48)
		MSE	1.35	(0.62, 2.08)
	Top-7	Time (s)	276.99	(276.24, 277.74)
		MSE	0.76	(0.59, 0.93)
	Top-8	Time (s)	414.30	(413.48, 415.12)
		MSE	0.72	(0.55, 0.88)
	Top-9	Time (s)	546.09	(544.37, 547.81)
		MSE	0.67	(0.65, 0.69)
SMC-P	Top-6	Time (s)	210.73	(210.23, 211.22)
		MSE	1.35	(0.68, 2.03)
	Top-7	Time (s)	410.09	(409.03, 411.15)
		MSE	0.70	(0.60, 0.80)
	Top-8	Time (s)	599.11	(597.75, 600.46)
		MSE	0.74	(0.56 0.92)
	Top-9	Time (s)	763.33	(760.52, 766.16)
		MSE	0.66	(0.64, 0.68)

Table 6.3.2: Summary table of the MSE and time taken for ten experiment runs of the MCMC algorithm with the top-k, $k = 5, \dots, 9$, rankings of the complete modified Sushi data set of ten items and SMC algorithm with uniform (SMC-U) and the pseudo-likelihood (SMC-P) augmentation methods when we reach the iterations where we have observed the top-k, $k = 6, \dots, 9$, rankings for all assessors.

which have a higher posterior probability that does not align with $\boldsymbol{\rho}^*$ contribute to a larger MSE value. By the time we have observed the top-9 we effectively have the determined complete data set. At this point, using the SMC algorithm with the pseudo-likelihood augmentation method, we see that the posterior probabilities for the highest and lowest-ranked items are captured but the probabilities for middle-ranked items diverge from the marginal posterior estimate obtained by MCMC. However, the posterior estimate of α with SMC remains somewhat consistent with MCMC despite the different posterior estimate of $\boldsymbol{\rho}$. We conclude that the SMC is not able to match the MCMC method's sequential posterior estimations and we would recommend implementing MCMC when we observe updated partial rankings from existing assessors.

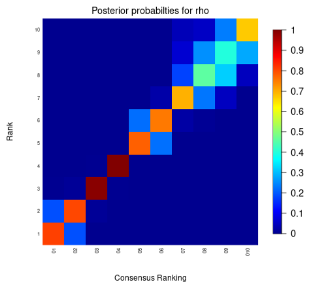
The intuition for why this behaviour occurs with SMC is because the modified Sushi data set is shown to have large variation (a low value of α indicates high variability) which makes the estimation more challenging. Each correction required for an updated partial ranking in a particle will reduce that particular particle's weight. Therefore, if several corrections are made for a particle then it is likely to be discarded during the resampling stage. The resampled particles rely heavily on more applications of the MCMC move kernel to explore the current posterior.

6.4 Conclusion

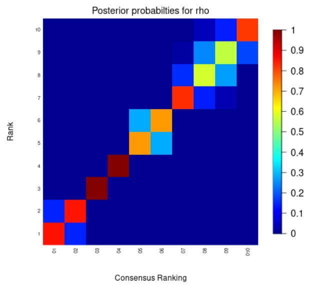
In this chapter, we applied the SMC framework for the Mallows model for partial rankings to consider an alternative scenario in the online setting. Instead of observing a new ranking from a new assessor, we receive an updated ranking from a known assessor. The update considered is a new rank to a previously unranked item by the known assessor whilst all of the remaining observed components of the ranking remain unchanged.

We considered how to correct the particle weights to account for the updated information of the observed components of the partial rankings. The proposed SMC algorithm would otherwise break because the estimated augmented rankings from the previous time step are not compatible with updated observed partial rankings. In both cases, we augment the latent components of each partial ranking using the uniform and pseudo-likelihood augmentation approaches.

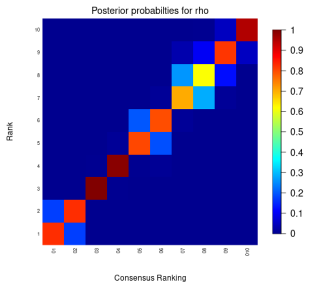
From the experiment results, we conclude that although we can correct the auxiliary components of each updated partial ranking and adjust the particle weights to account for this, currently, the proposed SMC methodology does not compute faster than if we were to rerun the MCMC algorithm with each newly updated data set. We conclude that the SMC is not able to match the MCMC's sequential posterior estimations and we would recommend that MCMC is implemented when we observe updated partial rankings from existing assessors.



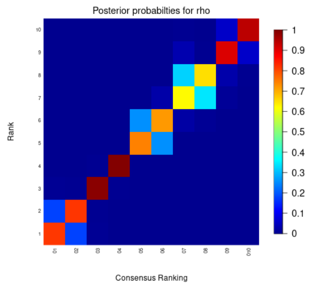
(a) MSE = 0.36, $\hat{\alpha}$ = 2.00, (1.70, 2.29).



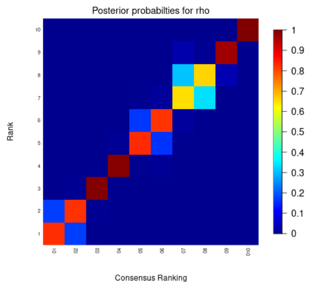
(b) MSE = 0.24, $\hat{\alpha}$ = 1.97, (1.70, 2.27).



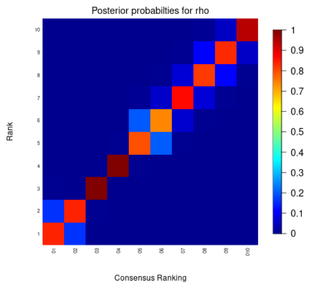
(c) MSE = 0.18, $\hat{\alpha}$ = 2.00, (1.70, 2.28).



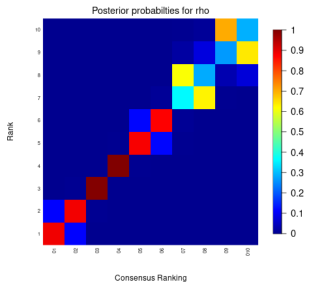
(d) MSE = 0.20, $\hat{\alpha}$ = 1.99, (1.74, 2.29).



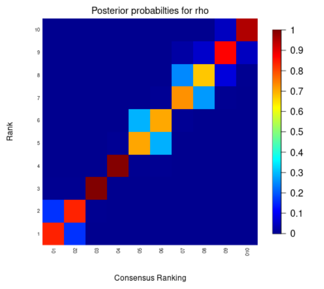
(e) MSE = 0.15, $\hat{\alpha}$ = 2.02, (1.73, 2.30).



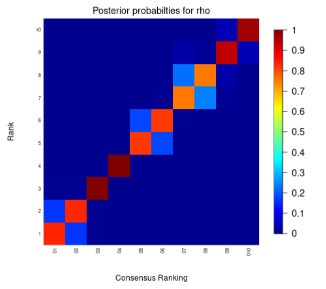
(f) MSE = 0.14, $\hat{\alpha}$ = 2.07, (1.74, 2.37).



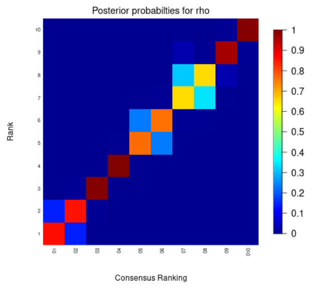
(g) MSE = 0.36, $\hat{\alpha}$ = 1.96, (1.69, 2.30).



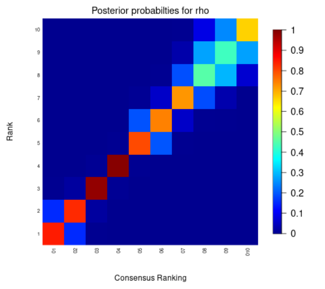
(h) MSE = 0.18, $\hat{\alpha}$ = 1.99, (1.72, 2.28).



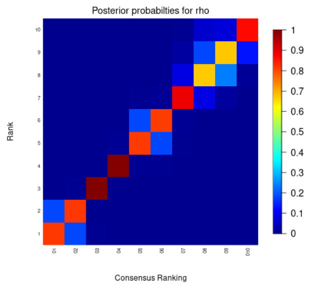
(i) MSE = 0.14, $\hat{\alpha}$ = 1.93, (1.65, 2.26).



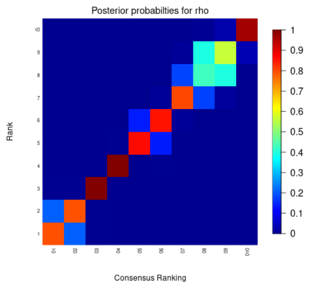
(j) MSE = 0.16, $\hat{\alpha}$ = 1.93, (1.64, 2.23).



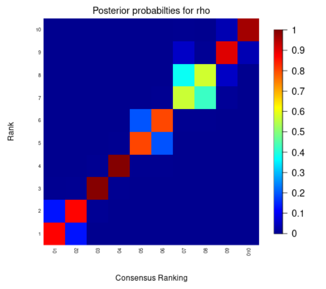
(k) MSE = 0.32, $\hat{\alpha}$ = 2.01, (1.71, 2.37).



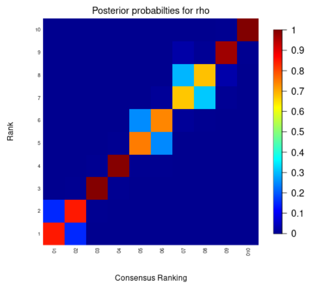
(l) MSE = 0.20, $\hat{\alpha}$ = 1.98, (1.65, 2.28).



(m) MSE = 0.20, $\hat{\alpha}$ = 1.98, (1.70, 2.27).

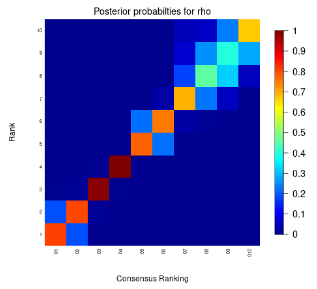


(n) MSE = 0.19, $\hat{\alpha}$ = 1.94, (1.69, 2.28).

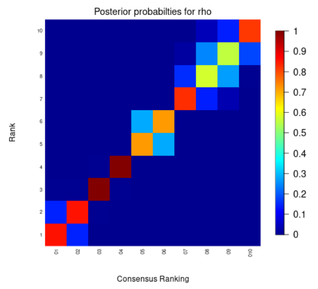


(o) MSE = 0.16, $\hat{\alpha}$ = 1.94, (1.66, 2.28).

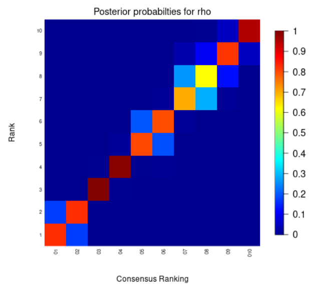
Figure 6.3.1: Heat plots of the posterior probabilities, for $n = 10$ items, for being ranked as the k^{th} most preferred, for $k = 1, \ldots, 10$ using the MCMC algorithm (top), the SMC algorithm with the uniform augmentation method (middle), and the SMC algorithm with the pseudo-likelihood augmentation method (bottom) given the top-5 (column 1) rankings, top-6 (column 2),...,top-9 (column 5) rankings. On the x-axis of each heat plot, the items are ordered according to their CP consensus. The MSE value and the mean estimate of α with its corresponding 95% HDPI are included in the captions.



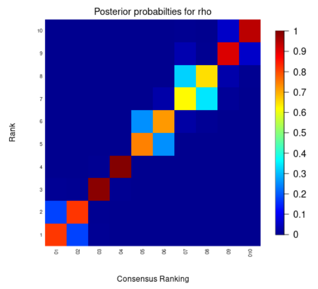
(a) MSE = 0.36, $\hat{\alpha}$ = 2.00, (1.70, 2.29).



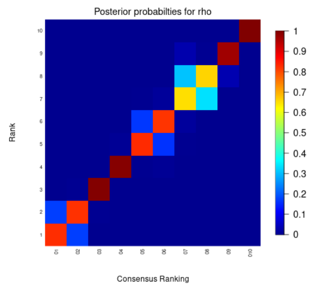
(b) MSE = 0.24, $\hat{\alpha}$ = 1.97, (1.70, 2.27)



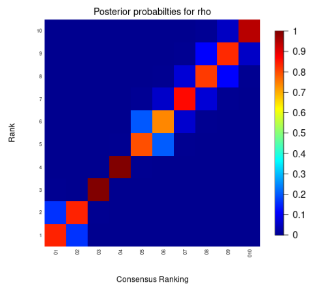
(c) MSE = 0.18, $\hat{\alpha}$ = 2.00, (1.70, 2.28).



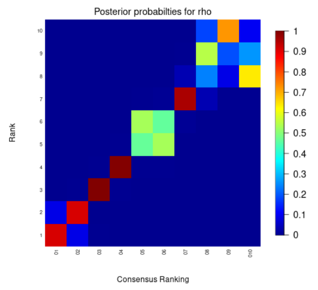
(d) MSE = 0.20, $\hat{\alpha}$ = 1.99, (1.74, 2.29).



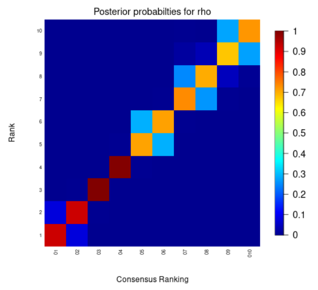
(e) MSE = 0.15, $\hat{\alpha}$ = 2.02, (1.73, 2.30).



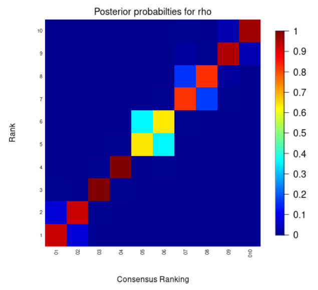
(f) MSE = 0.14, $\hat{\alpha}$ = 2.07, (1.74, 2.37).



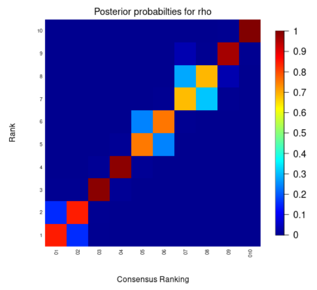
(g) MSE = 0.62, $\hat{\alpha}$ = 1.91, (1.68, 2.26).



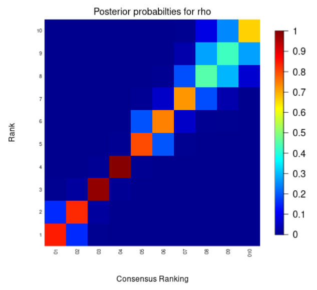
(h) MSE = 0.20, $\hat{\alpha}$ = 1.93, (1.67, 2.28).



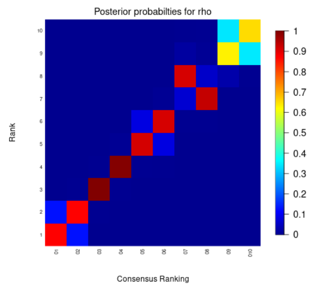
(i) MSE = 0.14, $\hat{\alpha}$ = 1.86, (1.38, 2.27)



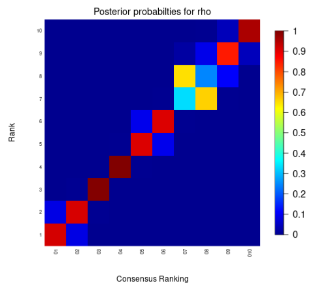
(j) MSE = 0.16, $\hat{\alpha}$ = 1.88, (1.50, 2.29).



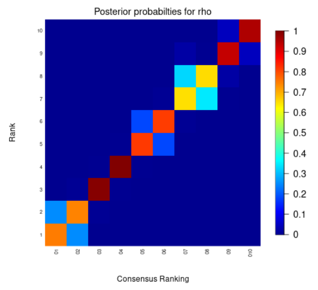
(k) MSE = 0.32, $\hat{\alpha}$ = 2.01, (1.71, 2.37).



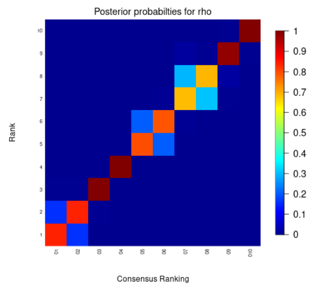
(l) MSE = 0.24, $\hat{\alpha}$ = 1.94, (1.65, 2.26).



(m) MSE = 0.20, $\hat{\alpha}$ = 1.90, (1.65, 2.22).

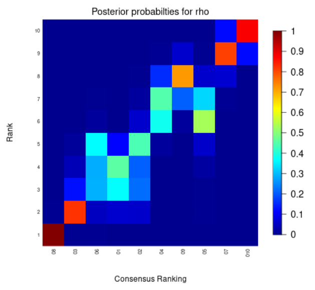


(n) MSE = 0.18, $\hat{\alpha}$ = 1.89, (1.60, 2.23).

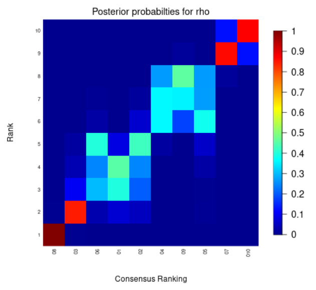


(o) MSE = 0.14, $\hat{\alpha}$ = 1.89, (1.61, 2.20).

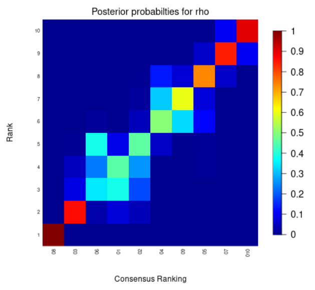
Figure 6.3.2: Heat plots of the posterior probabilities, for $n = 10$ items, for being ranked as the k^{th} most preferred, for $k = 1, ..., 10$ using the MCMC algorithm (top), the SMC algorithm with the uniform augmentation method (middle), and the SMC algorithm with the pseudo-likelihood augmentation method (bottom) given the top-5 (column 1) rankings, top-6 (column 2),...,top-9 (column 5) rankings. On the x-axis of each heat plot, the items are ordered according to their CP consensus. The MSE value and the mean estimate of α with its corresponding 95% HDPI are included in the captions.



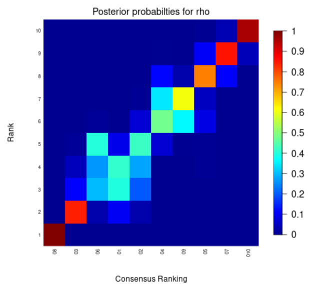
(a) MSE = 1.06, $\hat{\alpha}$ = 1.71, (1.40, 1.99).



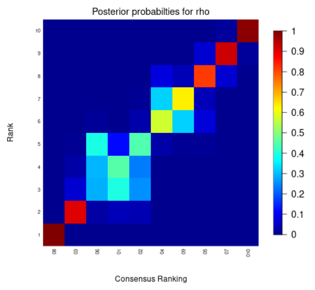
(b) MSE = 1.02, $\hat{\alpha}$ = 1.70, (1.40, 1.99).



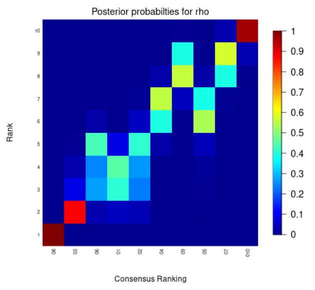
(c) MSE = 0.68, $\hat{\alpha}$ = 1.69, (1.41, 1.98).



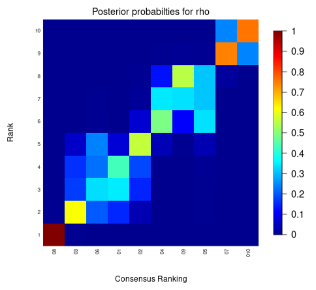
(d) MSE = 0.68, $\hat{\alpha}$ = 1.71, (1.44, 1.99).



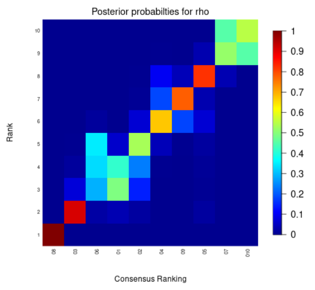
(e) MSE = 0.61, $\hat{\alpha}$ = 1.70, (1.45, 2.00).



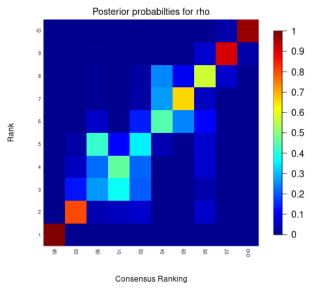
(f) MSE = 1.07, $\hat{\alpha}$ = 1.74, (1.43, 2.02).



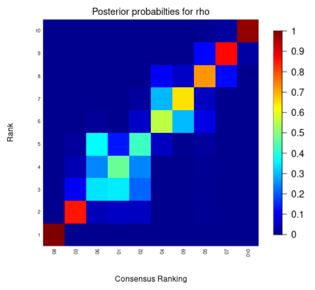
(g) MSE = 0.92, $\hat{\alpha}$ = 1.67, (1.37, 1.96).



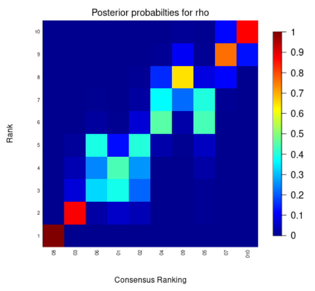
(h) MSE = 0.68, $\hat{\alpha}$ = 1.68, (1.39, 1.99).



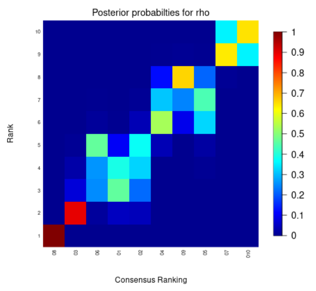
(i) MSE = 1.24, $\hat{\alpha}$ = 1.67, (1.36, 1.98).



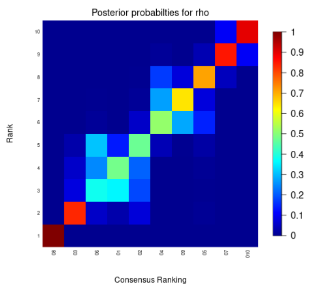
(j) MSE = 0.68, $\hat{\alpha}$ = 1.68, (1.40, 2.00).



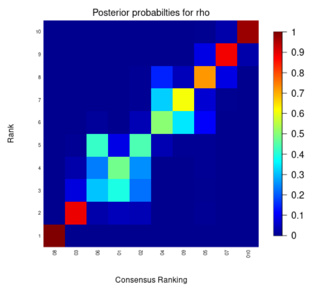
(k) MSE = 1.04, $\hat{\alpha}$ = 1.71, (1.41, 2.05).



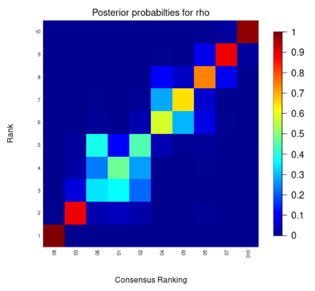
(l) MSE = 0.92, $\hat{\alpha}$ = 1.66, (1.30, 1.98).



(m) MSE = 0.74, $\hat{\alpha}$ = 1.65, (1.35, 1.94).

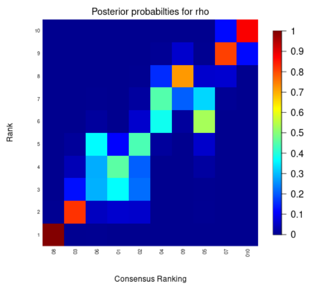


(n) MSE = 0.69, $\hat{\alpha}$ = 1.68, (1.38, 1.97).

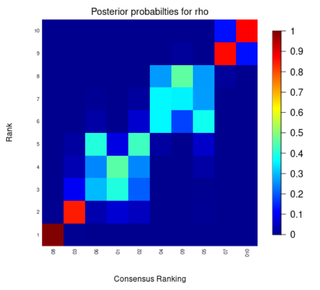


(o) MSE = 0.64, $\hat{\alpha}$ = 1.70, (1.38, 2.02).

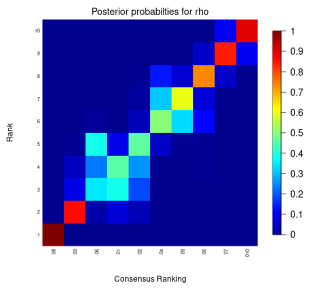
Figure 6.3.3: Heat plots of the posterior probabilities, for $n = 10$ sushi items, for being ranked as the k^{th} most preferred, for $k = 1, ..., 10$ using the MCMC algorithm (top), the SMC algorithm with the uniform augmentation method (middle), and the SMC algorithm with the pseudo-likelihood augmentation method (bottom) given the top-5 (column 1) rankings, top-6 (column 2),...,top-9 (column 5) rankings. On the x-axis of each heat plot, the items are ordered according to their CP consensus. The MSE value and the mean estimate of α with its corresponding 95% HDPI are included in the captions.



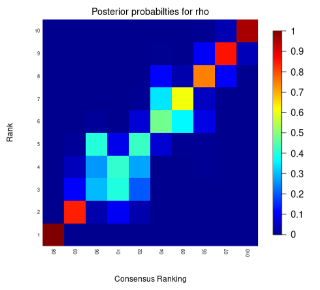
(a) MSE = 1.06, $\hat{\alpha}$ = 1.71, (1.40, 1.99).



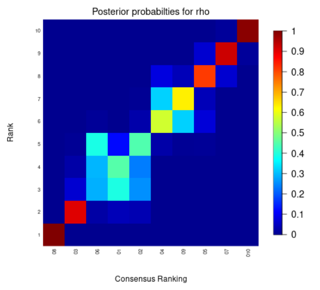
(b) MSE = 1.02, $\hat{\alpha}$ = 1.70, (1.40, 1.99).



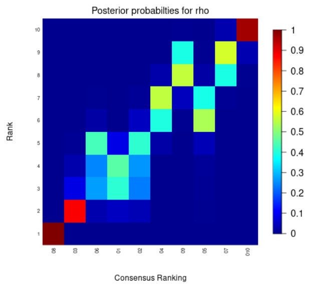
(c) MSE = 0.68, $\hat{\alpha}$ = 1.69, (1.41, 1.98)



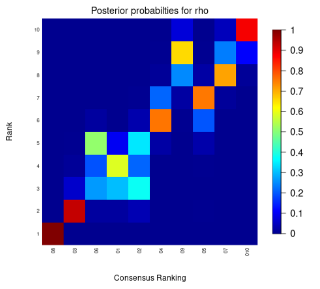
(d) MSE = 0.68, $\hat{\alpha}$ = 1.71, (1.44, 1.99).



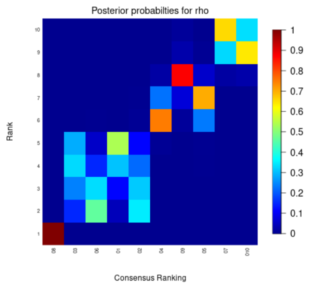
(e) MSE = 0.61, $\hat{\alpha}$ = 1.70, (1.45, 2.00).



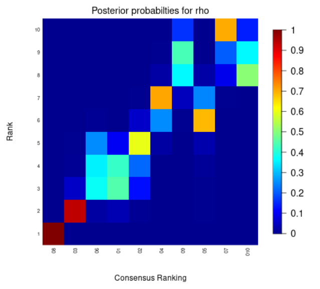
(f) MSE = 1.07, $\hat{\alpha}$ = 1.74, (1.43, 2.02).



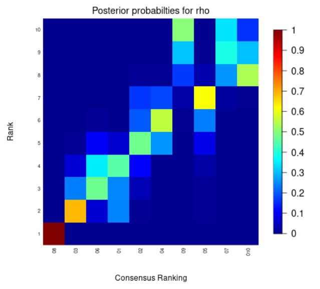
(g) MSE = 1.16, $\hat{\alpha}$ = 1.59, (1.22, 1.96).



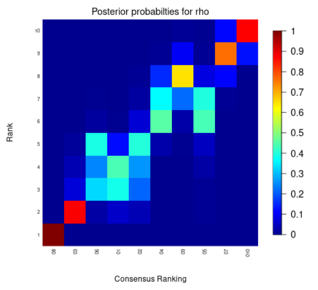
(h) MSE = 1.49, $\hat{\alpha}$ = 1.57, (1.32, 1.92).



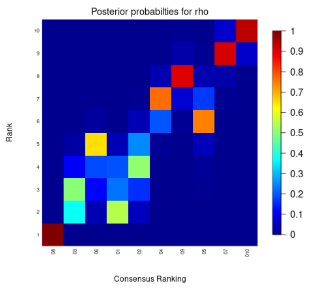
(i) MSE = 1.41, $\hat{\alpha}$ = 1.59, (1.32, 1.91)



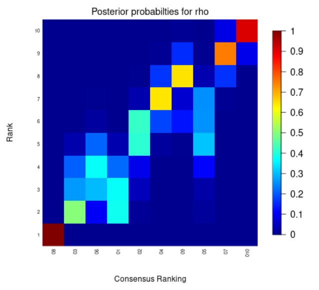
(j) MSE = 1.65, $\hat{\alpha}$ = 1.58, (1.34, 1.88).



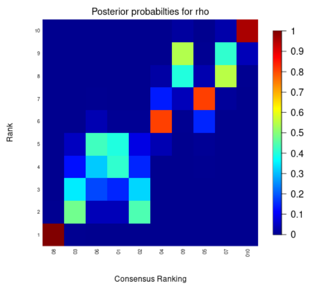
(k) MSE = 1.04, $\hat{\alpha}$ = 1.71, (1.41, 2.05).



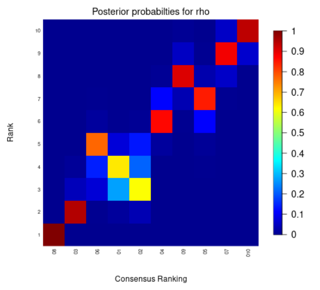
(l) MSE = 1.38, $\hat{\alpha}$ = 1.64, (1.30, 2.05).



(m) MSE = 1.56, $\hat{\alpha}$ = 1.63, (1.28, 2.00).



(n) MSE = 1.50, $\hat{\alpha}$ = 1.62, (1.31, 1.97).



(o) MSE = 0.97, $\hat{\alpha}$ = 1.62, (1.36, 2.00).

Figure 6.3.4: Heat plots of the posterior probabilities, for 10 sushi items, for being ranked as the k^{th} most preferred, for $k = 1, ..., 10$ using the MCMC algorithm (top), the SMC algorithm with the uniform augmentation method (middle), and the SMC algorithm with the pseudo-likelihood augmentation method (bottom) given the top-5 (column 1) rankings, top-6 (column 2),...,top-9 (column 5) rankings. On the x-axis of each heat plot, the items are ordered according to their CP consensus. The MSE value and the mean estimate of α with its corresponding 95% HDPI are included in the captions.

Chapter 7

Sequential Monte Carlo for the Mallows Mixture Model with Full Rankings

7.1 Introduction

In this chapter, we extend the work of Chapter 4 again to consider the sequential clustering problem with full rankings in an online setting. We assume that the collection of observed rankings is no longer generated from a single Mallows model, but from a Mallows mixture model consisting of several components, each with its own unique model parameters. Within the Bayesian framework for clustering, we are interested in estimating the posterior density of the Mallows mixture model. This includes the parameter estimation of the model parameters for each component but also the component weights and the cluster assignments for each observation.

The chapter is structured as follows. In Section 7.2, we formally describe the problem and propose the methodology to perform sequential clustering with a mixture model with a fixed number of components each time we observe a collection of new full rankings. The method is tested with synthetic data in Section 7.3.1 and with a benchmark Sushi data set in Section 7.3.2.

7.2 Proposed method

7.2.1 Problem outline

We define the Mallows mixture model consisting of C components. Each ranking is assigned a cluster label $\{z_1, \ldots, z_M\} \in \{1, \ldots, C\}$ to indicate which component it belongs to. The resulting likelihood of the Mallows mixture model is, assuming conditional independence across clusters, given M rankings is

$$p(\mathbf{R}_1, \ldots, \mathbf{R}_M | \{\boldsymbol{\rho}_c, \alpha_c\}_{c=1,\ldots,C}, z_1, \ldots, z_M) = \prod_{j=1}^{M} \frac{1_{\mathcal{P}_m}(\mathbf{R}_j)}{Z_m(\alpha_{z_j})} \exp\Big\{ -\frac{\alpha_{z_j}}{m} d(\mathbf{R}_j, \boldsymbol{\rho}_{z_j}) \Big\}.$$

The cluster labels, $z_1, \ldots, z_M$, are indicator variables that assign each ranking to one of C components and are distributed a priori according to $p(z_1, \ldots, z_M | \tau_1, \ldots, \tau_C) = \prod_{j=1}^{N} \tau_{z_j}$, where τ_c is the probability of assigning an assessor to cluster $c \in \{1, \ldots, C\}$. These assignment probabilities have a standard symmetric Dirichlet prior $\pi(\tau_1, \ldots, \tau_C) = \Gamma(\psi C)\Gamma(\psi)^{-C} \prod_{c=1}^{C} \tau_c^{\psi-1}$, where $\Gamma(\cdot)$ is the gamma function.

Similar to Chapter 4, we assume the collection of full rankings is observed over a discrete timeline $t = 1, \ldots, T$. Each ranking remains unchanged and is drawn from one of several Mallows models. We express the likelihood for M_t rankings, where M_t is the number of rankings observed up to time t, under the Mallows mixture model now as $p(\mathbf{R}_{1:M_t}|\boldsymbol{\rho}_{1:C}, \alpha_{1:C}, z_{1:M_t})$. The task is to perform sequential clustering, that is to update the posterior estimate

$$\pi_t(\boldsymbol{\rho}_{1:C}, \alpha_{1:C}|\mathbf{R}_{1:M_t}) \propto \pi(\boldsymbol{\rho}_{1:C})\pi(\alpha_{1:C})\pi(\tau_{1:C})p(z_{1:M_t}|\tau_{1:C})p(\mathbf{R}_{1:M_t}|\boldsymbol{\rho}_{1:C}, \alpha_{1:C}, z_{1:M_t}),$$

each time we receive new rankings. Here, $\pi(\boldsymbol{\rho}) = (n!)^{-1}\mathbb{1}_{\mathcal{P}_n}(\boldsymbol{\rho})$ is the uniform prior for the consensus ranking, and $\pi(\alpha|\lambda) = \lambda \exp\{-\lambda\alpha\}\mathbb{1}_{[0,\infty)}(\alpha)$ is the exponential prior distribution for the scale parameter, as specified by Vitelli et al. (2018).

7.2.2 Methodology

The proposed SMC methodology for clustering continues to follow the resample-move SMC framework (Berzuini and Gilks, 2001). If we have no initial observed rankings at time $t = 0$, that is, $M_0 = 0$, then we generate the model parameter values for each particle using specified prior distributions for $\boldsymbol{\rho}_{1:C}$, $\alpha_{1:C}$ and $\tau_{1:C}$ and assign these an equal weight. Alternatively, if we have initial observation then we can take the output of an MCMC or SMC sampler to create the particle set. At time $t-1$, each particle now contains: a set of consensus rankings, $\boldsymbol{\rho}_{1:C,t-1}$; a set of scale parameters, $\alpha_{1:C,t-1}$; a set of components weights, $\tau_{1:C,t-1}$; a set of cluster labels, $z_{1:M_{t-1}}$; and a weight W_{t-1}. This can be represented as the particle set $\{\boldsymbol{\theta}_{t-1}^{(i)} = (\boldsymbol{\rho}_{1:C,t-1}^{(i)}, \alpha_{1:C,t-1}^{(i)}, \tau_{1:C,t-1}^{(i)}), z_{1:M_t}^{(i)}, W_{t-1}^{(i)}\}_{i=1}^N$. In one iteration of the SMC sampler, we observe $(M_t - M_{t-1})$ new full rankings. We propose the cluster allocation

$z_{M_{t-1}+1:M_t}$ by calculating the contribution that each ranking $\mathbf{R}_{M_{t-1}+1:M_t}$ would contribute to the likelihood of each component of the mixture model in each particle. These contributions are normalised and we sample the assignment via a multinomial distribution using these normalised contributions as the multinomial probabilities. The proposal density in each SMC iteration is therefore defined as cluster assignment probability for a batch of $(M_t - M_{t-1})$ rankings for each particle can be expressed as

$$
\begin{aligned}
q_t(\boldsymbol{\theta}_t^{(i)}, z_{1:M_t}^{(i)} | \boldsymbol{\theta}_{t-1}^{(i)}, z_{1:M_{t-1}}^{(i)} \mathbf{R}_{1:M_t}) &= q_t(z_{M_{t-1}+1:M_t}^{(i)} | \boldsymbol{\theta}_t^{(i)}, \mathbf{R}_{M_{t-1}+1:M_t}) \\
&= \prod_{j=M_{t-1}+1}^{M_t} \frac{\frac{\tau_{z_j}^{(i)}}{Z_n(\alpha_{z_j}^{(i)})} \exp\left\{ \frac{-\alpha_{z_j}^{(i)}}{n} d(\mathbf{R}_j, \boldsymbol{\rho}_{z_j}) \right\}}{\sum_{c=1}^{C} \frac{\tau_c^{(i)}}{Z_n(\alpha_c^{(i)})} \exp\left\{ -\frac{\alpha_c^{(i)}}{n} d(\mathbf{R}_j,\ \boldsymbol{\rho}_c^{(i)}) \right\}}.
\end{aligned}
$$

After the cluster allocations are proposed, the particle weights are adjusted to account for the contribution that each new ranking has to their selected cluster component using the approximated likelihood $p(\mathbf{R}_{1:M_{t-1}} | \boldsymbol{\rho}_{1:C}, \alpha_{1:C}, z_{1:M_{t-1}})$ so that the particles are now weighted with respect to $p(\mathbf{R}_{1:M_t} | \boldsymbol{\rho}_{1:C}, \alpha_{1:C}, z_{1:M_t})$. Under importance sampling, the weight update for each particle is

$$
\begin{aligned}
\tilde{w}_t(\boldsymbol{\theta}_t^{(i)}, z_{1:M_t}^{(i)}) &= \frac{\pi_t(\boldsymbol{\theta}_t^{(i)}, z_{1:M_t}^{(i)}|\mathbf{R}_{1:M_t})}{\pi_{t-1}(\boldsymbol{\theta}_{t-1}^{(i)}, z_{1:M_{t-1}}^{(i)}|\mathbf{R}_{1:M_{t-1}})q_t(\boldsymbol{\theta}_t^{(i)}, z_{1:M_t}^{(i)}|\boldsymbol{\theta}_{t-1}^{(i)}, z_{1:M_{t-1}}^{(i)}\mathbf{R}_{1:M_t})} \\
&= \frac{\pi(\boldsymbol{\theta}_t^{(i)})p(z_{1:M_t}^{(i)}|\boldsymbol{\theta}_t^{(i)})p(\mathbf{R}_{1:M_t}|\boldsymbol{\theta}_t^{(i)}, z_{1:M_t}^{(i)})}{\pi(\boldsymbol{\theta}_{t-1}^{(i)})p(z_{1:M_{t-1}}^{(i)}|\boldsymbol{\theta}_{t-1}^{(i)})p(\mathbf{R}_{1:M_{t-1}}|\boldsymbol{\theta}_{t-1}^{(i)}, z_{1:M_{t-1}}^{(i)})} \\
&\quad \times \frac{p(\mathbf{R}_{1:M_{t-1}})}{p(\mathbf{R}_{1:M_t})} \times \frac{1}{q_t(\boldsymbol{\theta}_t^{(i)}, z_{1:M_t}^{(i)}|\boldsymbol{\theta}_{t-1}^{(i)}, z_{1:M_{t-1}}^{(i)}\mathbf{R}_{1:M_t})} \\
&= \frac{p(z_{M_{t-1}+1:M_t}^{(i)}|\boldsymbol{\theta}_t^{(i)})p(\mathbf{R}_{M_{t-1}+1:M_t}|\boldsymbol{\theta}_t^{(i)}, z_{M_{t-1}+1:M_t}^{(i)})/p(\mathbf{R}_{1:M_t})}{q_t(\boldsymbol{\theta}_t^{(i)}, z_{1:M_t}^{(i)}|\boldsymbol{\theta}_{t-1}^{(i)}, z_{1:M_{t-1}}^{(i)}\mathbf{R}_{1:M_t})/p(\mathbf{R}_{1:M_{t-1}})} \\
&\propto \frac{p(z_{M_{t-1}+1:M_t}^{(i)}|\boldsymbol{\theta}_t^{(i)})p(\mathbf{R}_{M_{t-1}+1:M_t}|\boldsymbol{\theta}_t^{(i)}, z_{M_{t-1}+1:M_t}^{(i)})}{q_t(\boldsymbol{\theta}_t^{(i)}, z_{1:M_t}^{(i)}|\boldsymbol{\theta}_{t-1}^{(i)}, z_{1:M_{t-1}}^{(i)}\mathbf{R}_{1:M_t})} \\
&= \prod_{j=M_{t-1}+1}^{M_t} \frac{\tau_{z_m}^{(i)} \exp\left\{\frac{-\alpha_{z_m}^{(i)}}{n} d(\mathbf{R}_j, \boldsymbol{\rho}_{z_m}^{(i)})\right\}}{Z_n(\alpha_{z_m}^{(i)})} \\
&\quad \times \frac{\sum_{c=1}^{C} \frac{\tau_c^{(i)}}{Z_n(\alpha_c^{(i)})} \exp\left\{ - \frac{\alpha_c^{(i)}}{n} d(\mathbf{R}_j,\ \boldsymbol{\rho}_c^{(i)})\right\}}{\frac{\tau_{z_m}^{(i)}}{Z_n(\alpha_{z_m}^{(i)})} \exp\left\{\frac{-\alpha_{z_m}^{(i)}}{n} d(\mathbf{R}_j, \boldsymbol{\rho}_{z_j^{(i)}}^{(i)})\right\}} \\
&= \prod_{j=M_{t-1}+1}^{M_t} \sum_{c=1}^{C} \frac{\tau_c^{(i)}}{Z_n(\alpha_c^{(i)})} \exp\left\{ - \frac{\alpha_c^{(i)}}{n} d(\mathbf{R}_j,\ \boldsymbol{\rho}_c^{(i)})\right\}.
\end{aligned}
$$

Multinomial resampling is employed to discard the lower-weighted particles whilst replicating the particles with greater weighting.

Finally, the moving stage consists of several steps. First, we use the proposal distributions suggested in Vitelli et al. (2018) to sample new values of $\boldsymbol{\rho}_{1:C}$ and $\alpha_{1:C}$. These updates are conditioned on the rankings that have been assigned to each cluster. Next, we perform a Gibbs update of the component weights $\tau_{1:C}$ by counting the number of rankings that are assigned to each cluster, denoted $n_c,\ c = 1, \ldots, C$ which updates the Dirichlet prior parameters to sample a new value for τ_c. Finally, we may perform a Gibbs update of allocations of the rankings to clusters. To reduce computational effort we may only reallocate the recently observed rankings. The pseudocode for the proposed SMC sampler for clustering with complete rankings is provided in Algorithm 14.

Algorithm 14: Sequential Monte Carlo for the Mallows Mixture Model with Full Rankings

```
Input: $C,\ \psi,\ \lambda,\ \sigma_\alpha,\ d(\cdot,\cdot),\ L,\ N,\ M_1,\dots,M_T;\ \mathbf{R}_1,\dots,\mathbf{R}_M;\ T,\ Z_n(\alpha)$.
Output: Posterior distributions of $\boldsymbol{\rho}_1,\dots,\boldsymbol{\rho}_C;\ \alpha_1,\dots,\alpha_C;\ \tau_1,\dots,\tau_C;\ z_1,\dots,z_M$.
Initialisation of SMC: randomly generate $\boldsymbol{\rho}_{1,0}^{(i)},\dots,\boldsymbol{\rho}_{C,0}^{(i)};\ \alpha_{1,0}^{(i)},\dots,\alpha_{C,0}^{(i)};$
  $\tau_{1,0}^{(i)},\dots,\tau_{C,0}^{(i)}$ for $i=1,\dots,N$.
for $t=1,\dots,T$ do
    Observe $\mathbf{R}_{M_{t-1}+1:M_t}$
    for $i=1:N$ do
        for $j=M_{t-1}+1:M_t$ do
            for $c=1:C$ do
                Compute $p_{cj}=\frac{\tau_{c,t}^{(i)}}{Z_n(\alpha_{c,t}^{(i)})}\exp\left\{-\frac{\alpha_{c,t}^{(i)}}{n}d(\mathbf{R}_j,\ \boldsymbol{\rho}_{c,t}^{(i)})\right\}$.
            end
            Sample $z_{j,t}^{(i)}\sim\mathcal{M}(p_{1j},\dots,p_{Cj})$.
        end
        Compute $w_t^{(i)}=\prod_{j=M_{t-1}+1}^{M_t}\sum_{c=1}^{C}\frac{\tau_c^{(i)}}{Z_n(\alpha_c^{(i)})}\exp\left\{-\frac{\alpha_c^{(i)}}{n}d(\mathbf{R}_j,\ \boldsymbol{\rho}_c^{(i)})\right\}$.
        Compute $w_t^{(i)}=W_{t-1}^{(i)}\tilde{w}_t^{(i)}$.
    end
    Resample $(k_1,\dots,k_N)\sim\mathcal{M}(W_t^{(1)},\dots,W_t^{(N)})$ and set
      $\{\boldsymbol{\theta}_t^{(1:N)},w_t^{(i)}\}\leftarrow\{\boldsymbol{\theta}_t^{(k_1:k_N)},\frac{1}{N}\}$.
    for $i=1:N$ do
        for $c=1:C$ do
            M-H step: update $\boldsymbol{\rho}_{c,t}^{(i)}$ with the leap-and-shift proposal
              $(\boldsymbol{\rho'}\sim \mathrm{L\&S}(\boldsymbol{\rho}_{c,t}^{(i)},L))$.
            M-H step: update $\alpha_{c,t}^{(i)}$ with log-normal proposal $(\alpha'\sim\log\mathcal{N}(\alpha_{c,t}^{(i)},\sigma_a^2))$.
            Gibbs step: compute $n_c=\sum_{j=1}^{M_t}1_c(z_j^{(i)})$ and sample
              $\tau_{1,t}^{(i)},\dots,\tau_{C,t}^{(i)}\sim\mathcal{D}(\psi+n_c)$.
        end
        for $j=M_{t-1}+1:M_t$ do
            for $c=1:C$ do
                Compute $p_{cj}=\frac{\tau_{c,t}^{(i)}}{Z_n(\alpha_{c,t}^{(i)})}\exp\left\{-\frac{\alpha_{c,t}^{(i)}}{n}d(\mathbf{R}_j,\ \boldsymbol{\rho}_{c,t}^{(i)})\right\}$.
            end
            Gibbs step: sample $z_{j,t}^{(i)}\sim\mathcal{M}(p_{1j},\dots,p_{Cj})$.
        end
    end
end
```

7.3 Experimental analyses

7.3.1 Simulated data: 20 items

We examine the proposed SMC methodology with a simulated data set containing $M = 1500$ rankings for $n = 20$ items. The data set is generated using the `sample_mallows` function in the `BayesMallows` R package (Sørensen et al., 2020) to create five data sets containing 500, 400, 300, 200 and 100 rankings, each with their own randomly generated consensus ranking but the same scale parameter, $\alpha_{1:5} = 5$. These data sets were concatenated and rearranged to create the final data set with an underlying distribution of a Mallows mixture model consisting of five components.

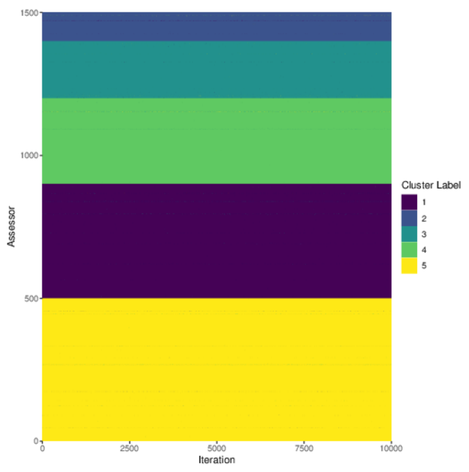
The SMC algorithm propagates $N = 2000$ particles for $T = 15$ iterations by introducing 100 new rankings at a time. In each iteration, the sample values for $\boldsymbol{\rho}_{1:5}$ and $\alpha_{1:5}$ in the particle set are updated with five applications of the MCMC move kernels. For every five SMC iterations, we update the cluster assignments $z_{1:M_t}$ for the rankings that had been observed up until that time. The MCMC algorithm of Vitelli et al. (2018) is run for 20,000 iterations with the first 10,000 iterations as burn-in. In both methods, we set the leap-size parameter in the leap-and-shift proposal for $\boldsymbol{\rho}$ to be $L = \lfloor n/5 \rfloor$, and we set the hyper-parameter and the standard deviation for the exponential proposal distribution for α to be $\lambda = 0.1$ and $\sigma_\alpha = 0.5$ respectively. We use the footrule distance throughout these experiments.

First, we inspected the cluster assignments of the rankings across the final MCMC chain and the SMC particles in their final iteration to see if label switching had occurred in Figure 7.3.1. Label switching is indicated if the cluster labels for any two rankings appear to swap between two components across either the final MCMC chain or SMC particle set. If label switching is observed, then we could perform a post-processing stage where we reallocate each ranking to its corrected cluster label based on the values of $\tau_{1:5}$ arranged in descending order. Alternatively, we could measure the distance from each of the shared consensus rankings of each component to some reference ranking and rearrange the allocations by considering the distances in descending order. There is no sign that label switching has occurred in Figure 7.3.1 and we see that nearly all of the rankings have been allocated to their matching component that was used to simulate the data set. This is likely due to the high value of the scale parameter, that was used to generate the simulated data set, so there is low variability around the consensus rankings. The cluster labels in the MCMC output need to be relabelled to match the cluster labels of the original data set and the output from the SMC algorithm.

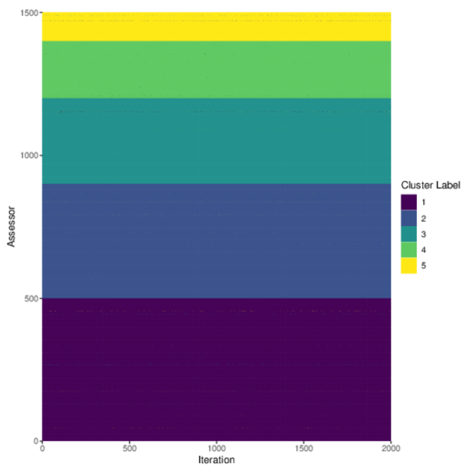
We assess the performance of the Bayesian methods in Table 7.3.1 which contains a summary of the posterior estimates for $\boldsymbol{\rho}_{1:5}$, $\alpha_{1:5}$ and $\tau_{1:5}$ from both experiments. The parameters, that were used to create the data set, are also provided. The component weights, $\tau_{1:5}$, match the posterior estimates using both methods and the estimates for $\alpha_{1:5}$ are similar and show variation from the underlying scale parameter value that was used to generate the rankings. The estimated consensus rankings in each component are also similar between both Bayesian methods with more differences in the item ranks occurring in the smallest components as we would expect.

The heat plots in Figure 7.3.2 show the posterior probabilities of assigning the rankings to each of the cluster components. Again, similarly to the heat plots in

Figure 7.3.1, there is certainty in which rankings belong to each component.



(a) MCMC.

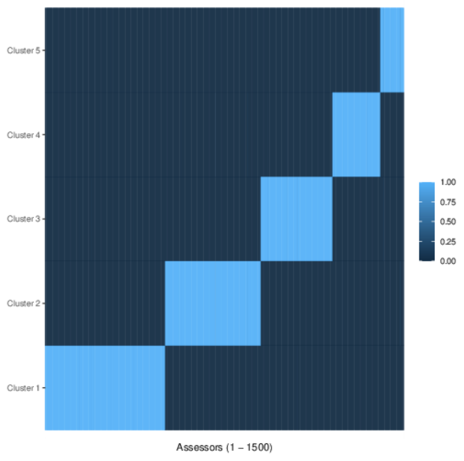


(b) SMC.

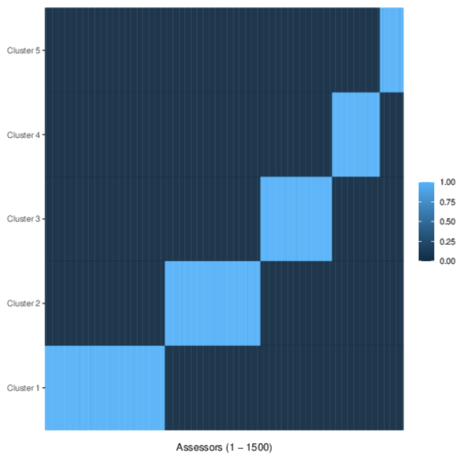
Figure 7.3.1: Heat plots of the cluster labels for each ranking in the first simulated data set across the MCMC chain after burn-in (left) and the particles in the final time step (right) respectively. It is clear that there are no signs of label switching and the cluster components between the MCMC and SMC algorithm can be paired as 5-1, 1-2, 4-3, 3-4 and 2-5.

	Cluster 1			Cluster 2			Cluster 3			Cluster 4			Cluster 5		
		MCMC	SMC		MCMC	SMC		MCMC	SMC		MCMC	SMC		MCMC	SMC
τ	0.33	0.33 (0.31, 0.35)	0.33 (0.31, 0.35)	0.26	0.26 (0.24, 0.29)	0.26 (0.24, 0.29)	0.20	0.20 (0.18, 0.22)	0.20 (0.18, 0.22)	0.14	0.14 (0.12, 0.15)	0.14 (0.12, 0.15)	0.07	0.07 (0.06, 0.08)	0.07 (0.06, 0.08)
α	5	5.10 (4.97, 5.22)	5.12 (4.95, 5.21)	5	4.84 (4.71, 4.98)	4.82 (4.64, 4.98)	5	4.91 (4.75, 5.07)	4.88 (4.72, 5.04)	5	5.10 (4.89, 5.29)	5.09 (4.92, 5.29)	5	4.75 (4.49, 5.05)	4.73 (4.45, 5.02)
ρ_1	16	16	16	3	3	3	17	17	17	13	13	13	20	20	20
ρ_2	5	5	5	4	4	4	6	6	6	3	3	3	17	17	17
ρ_3	12	12	12	5	5	5	19	19	19	18	18	18	6	6	6
ρ_4	15	15	15	2	2	2	20	20	20	6	6	6	18	18	18
ρ_5	9	9	9	14	14	14	13	13	13	12	12	12	5	5	5
ρ_6	19	19	19	15	15	15	1	1	1	4	4	4	16	16	16
ρ_7	6	6	6	8	8	8	9	9	9	9	9	9	13	14	14
ρ_8	4	4	4	11	11	11	8	8	8	7	7	7	1	1	1
ρ_9	2	2	2	19	19	19	10	10	10	19	19	19	2	3	3
ρ_{10}	7	7	7	20	20	20	15	15	15	17	17	17	19	19	19
ρ_{11}	14	14	14	7	7	7	16	16	16	5	5	5	10	10	10
ρ_{12}	10	10	10	9	9	9	18	18	18	8	8	8	12	12	12
ρ_{13}	11	11	11	13	13	13	3	3	3	14	14	14	15	15	15
ρ_{14}	20	20	20	6	6	6	11	11	11	15	15	15	3	2	2
ρ_{15}	13	13	13	12	12	12	12	12	12	1	1	1	4	4	4
ρ_{16}	8	8	8	17	17	17	7	7	7	20	20	20	11	11	11
ρ_{17}	17	17	17	16	16	16	2	2	2	16	16	16	7	7	7
ρ_{18}	1	1	1	18	18	18	14	14	14	10	10	10	9	9	9
ρ_{19}	18	18	18	1	1	1	4	4	4	11	11	11	8	8	8
ρ_{20}	3	3	3	10	10	10	5	5	5	2	2	2	14	13	13

Table 7.3.1: Results summary of the first simulated data experiment. The cluster components are labelled from left to right in descending order of the component weights. The posterior estimates for $\boldsymbol{\tau}$ and $\boldsymbol{\alpha}$, with their 95% HDPIs in parenthesis, and each $\boldsymbol{\rho}_c$, $c = 1, \ldots, 5$ that was used to generate the data set and the MAP consensus ranking estimates of the cluster components obtained from the MCMC and SMC experiments are provided.



(a) MCMC.



(b) SMC.

Figure 7.3.2: Heat plots of the posterior probabilities for each of the 1,500 assessors in the first simulated data set being assigned to each of the five clusters using the Bayesian methods.

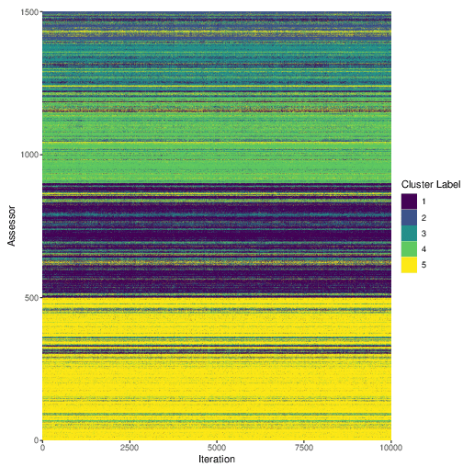
We perform the same simulated data experiment again, but this time we reduce the scale parameter for all cluster components to $\alpha_{1:5} = 2$. This will result in higher variability in the rankings generated around each of the component's "true" consensus rankings.

We check for any indication that label switching has occurred by observing the cluster assignments for each ranking from the MCMC and SMC output with the heat plots in Figure 7.3.4. Again, we do not see any sign of label switching in this instance and the cluster labels from the MCMC output must be relabelled so that they match the original labels of the simulated data set. The small scale parameter value has resulted in a significant number of rankings being allocated to more than one component or to a component that is not the same component it was generated from.

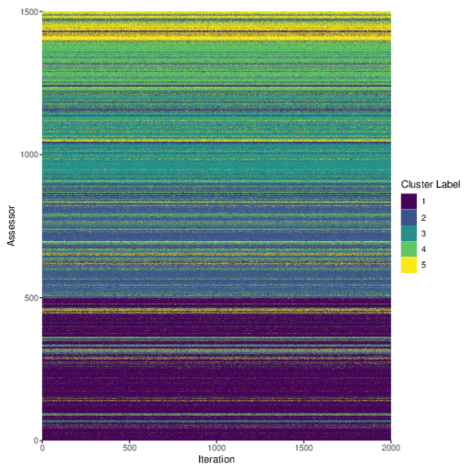
Table 7.3.2 presents the estimates of the mixture model posterior. The posterior estimates for $\tau_{1:5}$ are relatively similar, but we do see a slightly higher weight for the smaller components in the SMC output. The estimates for $\alpha_{1:5}$ vary much more between both methods for the three smallest components in the cluster. Despite this, we see that the estimates for $\boldsymbol{\rho}_{1:5}$ are a close match between both methods with only some small variation from the underlying values of $\boldsymbol{\rho}^*_{1:5}$ that were used to generate the simulated data set.

The cluster assignment plots in Figure 7.3.4 show a greater variation in the posterior probabilities for assigning rankings to clusters. We would expect this since there is a higher variance in how much the generated ranking differs from their "true" consensus ranking, so we could justify a ranking possibly belonging to more than one component. Overall, there is a reasonable amount of certainty on which cluster component a ranking belongs to.

We also created several confusion matrices to see how often SMC replicates the cluster assignments matches that MCMC achieves and with the generated data set. These results are provided in Figure 7.3.3. It appears that the number of correct matches between both methods and with each method in comparison to the cluster labels of the generated data set seems to be similar. Any misallocation is likely due to a ranking being allocated to a cluster component which has the smallest distance between its shared consensus ranking and the ranking itself. If a tie occurs between two or more components with the same distance to a ranking, then the cluster label is sampled using the component weights as the likelihood of allocations.



(a) MCMC.

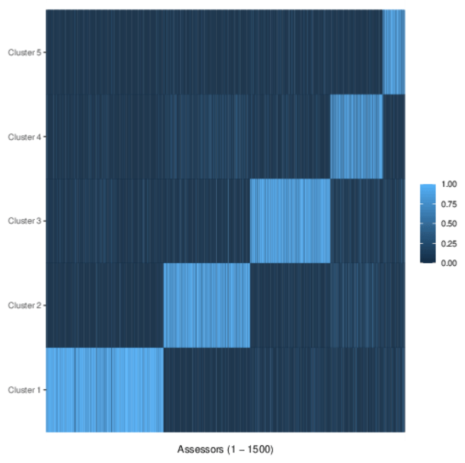


(b) SMC.

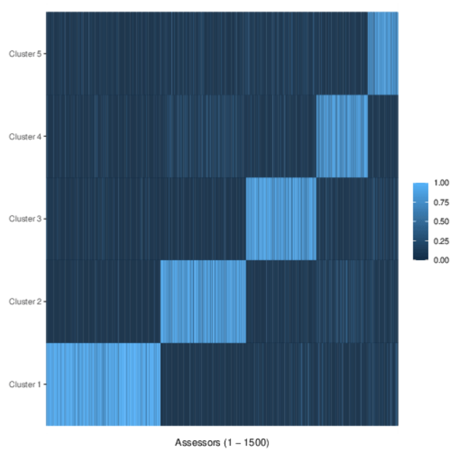
Figure 7.3.3: Heat plots of the cluster labels for each ranking in the second simulated data set across the MCMC chain after burn-in (left) and the particles in the final time step (right) respectively. It is clear that there are no signs of label switching and the cluster components between the MCMC and SMC algorithm can be paired as 5-1, 1-2, 4-3, 3-4 and 2-5.

	Cluster 1			Cluster 2			Cluster 3			Cluster 4			Cluster 5		
		MCMC	SMC		MCMC	SMC		MCMC	SMC		MCMC	SMC		MCMC	SMC
τ	0.33	0.32 (0.29, 0.35)	0.31 (0.28, 0.33)	0.26	0.23 (0.20, 0.27)	0.23 (0.21, 0.25)	0.20	0.22 (0.19, 0.26)	0.19 (0.17, 0.21)	0.14	0.15 (0.12, 0.19)	0.16 (0.14, 0.18)	0.07	0.08 (0.05, 0.11)	0.11 (0.09, 0.13)
α	2	2.20 (2.06, 2.32)	2.25 (2.18, 2.35)	2	1.85 (1.69, 2.03)	1.87 (1.76, 1.97)	2	1.92 (1.72, 2.09)	2.17 (2.03, 2.30)	2	1.86 (1.64, 2.13)	1.82 (1.64, 1.96)	2	2.20 (1.74, 2.64)	1.61 (1.44, 1.81)
ρ_1	16	16	16	3	3	3	17	18	18	13	8	9	20	20	20
ρ_2	5	5	5	4	5	4	6	5	6	3	4	3	17	15	16
ρ_3	12	12	12	5	4	5	19	17	17	18	17	17	6	6	6
ρ_4	15	15	15	2	2	2	20	20	20	6	3	5	18	18	18
ρ_5	9	9	9	14	14	14	13	13	13	12	15	13	5	5	5
ρ_6	19	19	19	15	17	16	1	2	3	4	5	4	16	16	17
ρ_7	6	6	6	8	9	10	9	11	10	9	10	12	13	14	15
ρ_8	4	3	3	11	12	13	8	7	8	7	9	7	1	1	1
ρ_9	2	2	2	19	18	19	10	8	7	19	19	19	2	3	3
ρ_{10}	7	7	7	20	20	20	15	15	15	17	18	18	19	19	19
ρ_{11}	14	14	14	7	6	7	16	16	16	5	6	6	10	9	8
ρ_{12}	10	11	10	9	8	8	18	19	19	8	7	8	12	8	11
ρ_{13}	11	10	11	13	13	12	3	3	2	14	14	16	15	14	14
ρ_{14}	20	20	20	6	7	6	11	9	9	15	16	15	3	2	2
ρ_{15}	13	13	13	12	11	10	12	12	12	1	1	1	4	7	9
ρ_{16}	8	8	8	17	15	15	7	10	11	20	20	20	11	13	10
ρ_{17}	17	17	17	16	16	17	2	1	1	16	13	14	7	4	4
ρ_{18}	1	1	1	18	19	18	14	14	14	10	11	10	9	11	12
ρ_{19}	18	18	18	1	1	1	4	4	4	11	12	11	8	10	7
ρ_{20}	3	4	4	10	10	11	5	6	5	2	2	2	14	12	13

Table 7.3.2: Results summary of the second simulated data experiment. The cluster components are labelled from left to right in descending order of the component weights. The posterior estimates for $\boldsymbol{\tau}$ and $\boldsymbol{\alpha}$, with their 95% HDPIs in parenthesis, and each $\boldsymbol{\rho}_c$, $c = 1, \ldots, 5$ that was used to generate the data set and the MAP consensus ranking estimates of the cluster components obtained from the MCMC and SMC experiments are provided.



(a) MCMC.



(b) SMC.

Figure 7.3.4: Heat matrices of the posterior probabilities for each of the 1,500 assessors in the second simulated data set being assigned to each of the five clusters using the Bayesian methods.

	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5
Cluster 1	433	18	13	23	13
Cluster 2	23	302	18	39	18
Cluster 3	14	11	243	14	18
Cluster 4	11	25	18	137	9
Cluster 5	5	10	6	7	72

(a) Simulated data vs. MCMC.

	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5
Cluster 1	433	19	20	24	4
Cluster 2	27	303	24	41	4
Cluster 3	12	11	260	11	6
Cluster 4	10	21	24	137	8
Cluster 5	7	10	7	6	70

(b) Simulated data vs. SMC.

	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5
Cluster 1	478	0	0	3	8
Cluster 2	1	349	0	3	11
Cluster 3	6	2	296	8	23
Cluster 4	0	14	2	203	0
Cluster 5	1	1	0	3	88

(c) MCMC vs. SMC.

Table 7.3.3: Confusion plots of the number of correct cluster allocation matches in the second simulated data set between the Bayesian methods and with each other.

7.3.2 Real data: Sushi data

Finally, we assess the performance of the proposed SMC framework using the benchmark Sushi data set (Kamishima, 2003) which contains $M = 5000$ rankings of $n = 10$ sushi items. This has been analysed in Vitelli et al. (2018), but these results were obtained whilst there was a coding error in the function for sampling cluster probabilities from the Dirichlet distribution (Sørensen et al., 2020). For this reason, we shall use the experiment set-up suggestions from Sørensen et al. (2020) to compare the proposed SMC method against the MCMC algorithm of Vitelli et al. (2018).

We assume that there are $C = 5$ components in the mixture model following the observation of the elbow plot in Sørensen et al. (2020). In both methods, the leap-size parameter is set to $L = 1$, and the hyper-parameter and the standard deviation for the exponential proposal distribution for α to be $\lambda = 0.1$ and $\sigma_\alpha = 0.5$ respectively. For the SMC experiment, we propagate $N = 2000$ particles over $T = 50$ iterations by introducing 100 new rankings at a time. We propose the sample values for $\boldsymbol{\rho}_{1:5}$ and $\alpha_{1:5}$ in the particle set with five applications of the MCMC move kernels in each SMC iteration during the move stage of the algorithm. After every five SMC iterations, the cluster assignments for the M_t rankings are sampled once during the move stage as well. We run the MCMC algorithm for 27,000 iterations and discard the first 25,000 as burn-in.

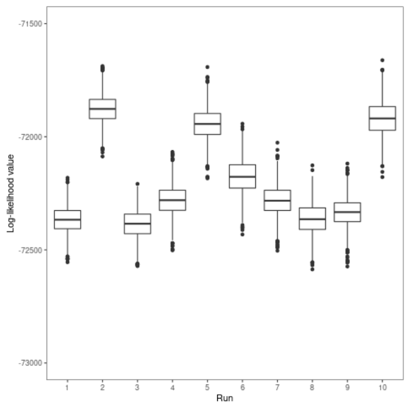
When we ran both Bayesian samplers, we observed a range of results for the posterior estimation of the Mallows mixture model. We replicated each experiment ten times and reported the log-likelihood at each run to assess if each sampler converges to different posterior modes. A summary of the log-likelihood values of the resulting posterior distributions from ten experiments is presented in Figure 7.3.5. A higher log-likelihood value indicates that a proposed method has reached a higher mode of possible solutions for the posterior distribution. In three out of ten cases, MCMC reaches a higher mode in runs 2, 5 and 10, but SMC achieves a slightly higher log-likelihood value on average compared to MCMC.

Next, we pooled the log-likelihood values from the ten experiment runs with both samplers to create two additional box plots in Figure 7.3.5c to show the overall behaviour of each Bayesian method. SMC has a higher mean and tighter quartile range, but it does obtain significantly lower values than MCMC, whereas MCMC can reach higher log-likelihood values. Overall, though MCMC can obtain solutions in higher modes of the posterior, sub-optimal solutions are found more often compared to the solutions obtained through SMC.

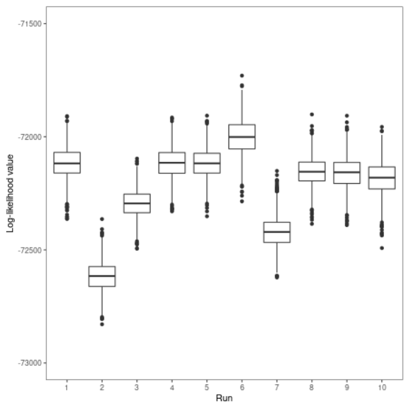
We view the mean posterior estimates of the component weights for each experiment run with each method to see which results obtain a higher mode of the posterior distribution. Table 7.3.4 shows the corresponding estimated component weights of the five components in the mixture model from each experiment run in Figure 7.3.5. Generally, there appear to be two solutions for the mixture model. The optimal solution consists of one component that accounts for approximately 40-50% of the rankings, with four smaller components that each contribute 10-20% of the remaining posterior. The sub-optimal solution consists of one component that accounts for 30-35% of the rankings, another with 20-25% and the remaining three with 10-20%.

Based on these observations we decided to inspect the remaining posterior estimates to see how similar are the outputs from the two Bayesian methods when we have a solution in the higher and lower mode of the posterior distribution. We present the results of the experiment runs, that returned the highest and lowest average log-likelihood values from each method in Table 7.3.5. In particular, Tables 7.3.5a and 7.3.5b are the results with the highest average log-likelihood values from MCMC and SMC respectively whilst Tables 7.3.5c and 7.3.5d contains the results with the lowest average log-likelihood values. We find that the pairwise comparison of the component weights, represented as a proportion, of each solution in descending order does not vary by more than 0.06. The consensus ranking for the smallest cluster is captured across all four summary tables. However, for the remaining clusters, there is variation in the estimates of $\boldsymbol{\rho}$ in both methods. In Tables, 7.3.5c and 7.3.5d we see that the likely matching component label pairs are 1-2, 2-1, 3-4 and 4-3. In Tables 7.3.5a and 7.3.5b 4-3 is also a matching component label pair. The SMC solution has more component weight assigned to cluster 3 than to clusters 4 and 5, whereas MCMC has distributed the weightings across these three clusters more evenly. This has a significant effect on the differences in parameter estimates between the methods. The overall patterns that we can see in Table 7.3.5 are that the fatty tuna sushi item is strongly preferred by the vast majority of assessors whilst the cucumber roll is strongly least preferred.

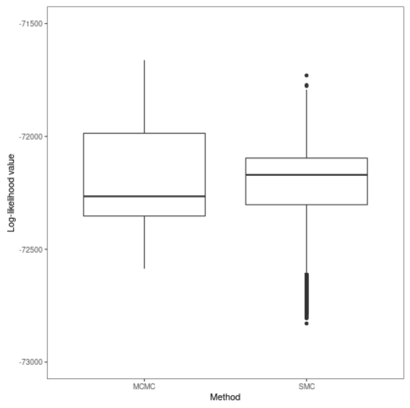
This particular data set is a challenging clustering problem because both Bayesian methods give a range of solutions to the estimated posterior and neither method is successfully sampling from the full posterior distribution. This could be because there are a small number of items in each ranking, which can make it difficult to determine the cluster labels for each ranking if the measured ranking distance between more than one component consensus ranking is similar compared to clustering rankings for a larger number of items.



(a) MCMC.



(b) SMC.



(c) Pooled results.

Figure 7.3.5: Boxplots of the log-likelihood of the posterior distributions obtained from ten experiment runs of the MCMC (top left) and SMC (top right) algorithms with the Sushi data set, and a summary box plot from each Bayesian method (bottom).

Run	τ_1	τ_2	τ_3	τ_4	τ_5
1	0.32	0.22	0.20	0.17	0.10
2	0.47	0.21	0.12	0.10	0.10
3	0.32	0.22	0.20	0.17	0.10
4	0.33	0.23	0.16	0.15	0.13
5	0.47	0.19	0.16	0.09	0.08
6	0.36	0.31	0.16	0.10	0.07
7	0.33	0.23	0.16	0.15	0.13
8	0.35	0.33	0.12	0.11	0.09
9	0.31	0.24	0.17	0.15	0.13
10	0.49	0.21	0.11	0.10	0.08

(a) MCMC.

Run	τ_1	τ_2	τ_3	τ_4	τ_5
1	0.39	0.23	0.18	0.11	0.09
2	0.33	0.24	0.19	0.16	0.08
3	0.31	0.23	0.21	0.17	0.08
4	0.39	0.25	0.16	0.11	0.09
5	0.39	0.22	0.20	0.12	0.08
6	0.33	0.24	0.18	0.09	0.06
7	0.30	0.26	0.18	0.17	0.09
8	0.38	0.24	0.19	0.11	0.08
9	0.41	0.26	0.13	0.11	0.08
10	0.39	0.23	0.18	0.11	0.09

(b) SMC.

Table 7.3.4: Summary table with the corresponding MAP estimates of the component weights $\tau_c,\ c = 1 \ldots 5$, from each of the ten experiment runs with the MCMC (left) and SMC (right) methods with the Sushi data set.

7.4 Conclusion

In this chapter, we extended the SMC algorithm for a single Mallows model which receives only full rankings as new observations to handle several Mallows models in a mixture. The proposed extension benefits by being able to process a larger batch of new observations at a time because the algorithm makes an informed initial cluster assignment for each ranking. This is done by considering the likelihood of assigning the ranking to each cluster given each particle's current parameter estimates and uniformly with these normalised probabilities. The effect of incorporating this stage is that although we need to account for these calculations for each new observed ranking in the particle weights, it reduces the number of MCMC move kernels applied to each particle after the resampling stage across all of the particles. We chose to update the cluster assignments once we processed a few SMC iterations instead of every iteration. This has greatly reduced the computational effort when we perform sequential updates of the Mallows mixture posterior for inference.

The experiments with the simulated data sets confirm that the value of the scale parameter has a greater effect on how well the MCMC and SMC algorithms can perform clustering. A larger value for α means that it is much easier to determine which component each ranking belongs to. A smaller scale parameter value will likely cause more uncertainty in a ranking's cluster label. We also found that the SMC algorithm is able to give similar posterior estimates as MCMC for less computational cost. Smaller components may vary in their estimate between both methods but this is likely due to the number of rankings allocated to a smaller component influencing the estimate of $\boldsymbol{\rho}$ and α. However, when we applied the SMC method to the benchmark Sushi data set, we discovered that both MCMC and SMC provided more than one possible solution to the clustering problem. Since both Bayesian methods find a variety of very different solutions, it is clear that neither method is successfully sampling from the full posterior distribution.

	Cluster 1	**Cluster 2**	**Cluster 3**	**Cluster 4**	**Cluster 5**
τ	0.47 (0.45, 0.49)	0.21 (0.19, 0.23)	0.12 (0.10, 0.13)	0.10 (0.09, 0.12)	0.10 (0.08, 0.11)
α	3.11 (3.03, 3.18)	2.55 (2.46, 2.65)	3.20 (3.01, 3.40)	3.45 (3.29, 3.60)	2.25 (2.07, 2.42)
ρ_1	Fatty tuna	Fatty tuna	Fatty tuna	Fatty Tuna	Shrimp
ρ_2	Sea urchin	Salmon roe	Tuna	Sea eel	Sea eel
ρ_3	Salmon roe	Tuna	Tuna roll	Tuna	Egg
ρ_4	Sea eel	Shrimp	Shrimp	Shrimp	Squid
ρ_5	Tuna	Squid	Squid	Tuna roll	Cucumber roll
ρ_6	Shrimp	Tuna roll	Tuna roll	Egg	Tuna
ρ_7	Tuna roll	Sea eel	Cucumber roll	Squid	Tuna roll
ρ_8	Squid	Egg	Sea eel	Cucumber roll	Fatty tuna
ρ_9	Egg	Cucumber roll	Salmon roe	Salmon roe	Salmon roe
ρ_{10}	Cucumber roll	Sea urchin	Sea urchin	Sea urchin	Sea urchin

(a) Results from run 2 of the MCMC experiment in Figure 7.3.5a.

	Cluster 1	**Cluster 2**	**Cluster 3**	**Cluster 4**	**Cluster 5**
τ	0.43 (0.40, 0.45)	0.24 (0.21, 0.27)	0.18 (0.16, 0.20)	0.09 (0.08, 0.10)	0.06 (0.05, 0.07)
α	3.29 (3.19, 3.39)	2.50 (2.37, 2.65)	2.94 (2.80, 3.12)	2.77 (2.58, 3.01)	2.53 (2.28, 2.77)
ρ_1	Fatty tuna	Fatty tuna	Fatty tuna	Sea urchin	Shrimp
ρ_2	Sea urchin	Tuna	Sea eel	Sea eel	Egg
ρ_3	Salmon roe	Shrimp	Tuna	Shrimp	Squid
ρ_4	Tuna	Tuna roll	Shrimp	Salmon roe	Sea eel
ρ_5	Sea eel	Squid	Tuna roll	Squid	Cucumber roll
ρ_6	Shrimp	Salmon roe	Squid	Fatty tuna	Salmon roe
ρ_7	Tuna roll	Egg	Egg	Tuna	Tuna roll
ρ_8	Squid	Sea eel	Cucumber roll	Tuna roll	Fatty tuna
ρ_9	Egg	Cucumber roll	Salmon roe	Egg	Tuna
ρ_{10}	Cucumber roll	Sea urchin	Sea urchin	Cucumber roll	Sea urchin

(b) Results from run 6 of the SMC experiment in Figure 7.3.5b.

	Cluster 1	**Cluster 2**	**Cluster 3**	**Cluster 4**	**Cluster 5**
τ	0.32 (0.30, 0.34)	0.22 (0.20, 0.24)	0.20 (0.18, 0.22)	0.17 (0.15, 0.19)	0.10 (0.09, 0.11)
α	3.41 (3.32, 3.39)	3.68 (3.47, 3.81)	3.37 (3.26, 3.56)	2.85 (2.67, 3.00)	2.07 (1.88, 2.27)
ρ_1	Fatty tuna	Fatty tuna	Fatty tuna	Fatty Tuna	Shrimp
ρ_2	Tuna	Sea urchin	Sea urchin	Sea eel	Sea eel
ρ_3	Shrimp	Salmon roe	Salmon roe	Tuna	Egg
ρ_4	Tuna roll	Sea eel	Tuna	Salmon roe	Squid
ρ_5	Squid	Shrimp	Shrimp	Sea urchin	Salmon roe
ρ_6	Sea eel	Tuna	Squid	Tuna roll	Cucumber roll
ρ_7	Egg	Squid	Tuna roll	Shrimp	Fatty tuna
ρ_8	Cucumber roll	Tuna roll	Sea eel	Egg	Tuna roll
ρ_9	Salmon roe	Egg	Cucumber roll	Squid	Tuna
ρ_{10}	Sea urchin	Cucumber roll	Egg	Cucumber roll	Sea urchin

(c) Results from run 3 of the MCMC experiment in Figure 7.3.5a.

	Cluster 1	**Cluster 2**	**Cluster 3**	**Cluster 4**	**Cluster 5**
τ	0.33 (0.31, 0.35)	0.24 (0.22, 0.26)	0.19 (0.17, 0.21)	0.16 (0.14, 0.17)	0.08 (0.07, 0.09)
α	3.41 (3.32, 3.39)	2.81 (2.70, 2.92)	2.45 (2.31, 2.58)	3.08 (3.07, 3.28)	2.25 (2.03, 2.46)
ρ_1	Fatty tuna	Fatty tuna	Fatty tuna	Fatty Tuna	Shrimp
ρ_2	Sea urchin	Tuna	Sea eel	Salmon roe	Sea eel
ρ_3	Salmon roe	Shrimp	Tuna	Sea urchin	Egg
ρ_4	Sea eel	Tuna roll	Shrimp	Tuna	Squid
ρ_5	Tuna	Squid	Salmon roe	Squid	Cucumber roll
ρ_6	Shrimp	Egg	Tuna roll	Shrimp	Salmon roe
ρ_7	Tuna roll	Sea eel	Squid	Tuna roll	Tuna roll
ρ_8	Squid	Cucumber roll	Egg	Sea eel	Fatty tuna
ρ_9	Egg	Salmon roe	Sea urchin	Egg	Tuna
ρ_{10}	Cucumber roll	Sea urchin	Cucumber roll	Cucumber roll	Sea urchin

(d) Results from run 2 of the SMC experiment in Figure 7.3.5b.

Table 7.3.5: Summary of the posterior estimates from several experiment runs in Figure 7.3.5 with the Sushi data set. Each column contains the posterior estimates of τ_c, α_c, with their corresponding 95% HDPIs, and the MAP for $\boldsymbol{\rho}_c$ for each component $c = 1, \dots, 5$.

Chapter 8

Conclusions

This thesis has focused on the development of an SMC framework for the Bayesian Mallows model. Specifically, we used the resample-move framework of Berzuini and Gilks (2001) to address the following research question: can we update the posterior distribution effectively to model the existing and new ranking data that we receive on a sequential basis? We believed that using SMC methods would mitigate the issues of increasing computational effort that typical MCMC methods for Bayesian inference would possess when addressing this problem. MCMC can be used for sequential inference, but for every observational update of rankings, we would need to rerun the MCMC chain from scratch with the existing and new rankings to estimate the new posterior. Instead, SMC methods can take the current posterior estimate given a set of existing rankings and update it based on the new rankings for a fixed computational complexity. There does not exist a previous extension of the Bayesian Mallows model that has focused on applying SMC methods as an alternative to typical MCMC sampling procedures.

Chapter 4 introduced the SMC framework for the Bayesian Mallows model to perform sequential updates of the estimated posterior distribution of a single Mallows model in an online setting where a collection of full rankings are received over a discrete timeline. The simulation studies revealed the overall behaviour of the method when various parameters and variables are adjusted. We compared how effective SMC was at performing inference against MCMC with real data sets. In this scenario, we were able to demonstrate that SMC methods can perform sequential inference and give posterior estimates similar to that of MCMC with less computational effort and time.

Chapter 5 built upon Chapter 4 by amending the proposed SMC framework to consider latent information, in the form of new partial rankings, to perform sequential updates of a single Mallows posterior. We considered two methods to perform data augmentation on the missing components of each partial ranking: a uniform augmentation, which selects the positions for the unallocated item ranks at random, and a pseudo-likelihood augmentation approach, which makes an informed allocation by considering the current Mallows model parameter estimates in each particle. We find that the pseudo-likelihood augmentation approach performs better than the uniform sampler as it can replicate the posterior estimates that are generated by the MCMC approach. Other augmentation methods could also be considered for future work. However, it was shown that the SMC algorithm performs better if we reduce the batch size of the number of new rankings to account for in each posterior update. If there are too many new partial rankings, then we reduce the chances of obtaining a particle set with decent augmentations being carried forward in the algorithm.

We also considered applying SMC to the scenario where existing assessors make additions to their partial ranking in Chapter 6. We discussed the theory of how to perform these updates when we need to account for the potential corrections for each particle's set of augmented rankings. We were able to show for small synthetic data examples how this approach can obtain a good posterior estimate. However, a large amount of computational effort is still required. We conclude that the SMC is not able to match the MCMC's sequential posterior estimations unless we rely on a large number of move kernel applications. We would recommend to use MCMC methods when we observe updated partial rankings from existing assessors.

Finally, Chapter 7 extended the SMC framework from Chapter 4 to the clustering setting where we perform sequential posterior updates for a Mallows mixture model with a known number of components. By making informed initial allocations for each new ranking, we have been able to significantly reduce the computational effort required to perform clustering compared to using MCMC. The proposed SMC methodology can provide similar inferences to MCMC for synthetic data sets with high and low variation in the generated data sets. The MCMC method was able to provide solutions considered optimal but the SMC method was able to provide slightly more consistent solutions than MCMC given their overall log-likelihood values range in repeated experiment runs. A small number of items in each ranking made it difficult to determine the cluster labels if one component has a similar ranking distance between each assessor's ranking and the consensus ranking for a particular component. We conclude that neither Bayesian method can reliably sample from the posterior because a number of different solutions are found when using each method.

8.1 Future work

Given the work conducted in this thesis, particularly in Chapter 6, we believe that further investigation should take place in this area. We outline some suggestions for future work related to this project.

8.1.1 Sequential Monte Carlo for Mallows mixture model with partial rankings

Sequential clustering could naturally be extended to handle partial rankings. This task would be similar to the problem addressed in Chapter 5, but extended to handle multiple Mallows models in Chapter 6. The particle set would contain the values for the Mallows mixture model $\{\boldsymbol{\theta}_t = (\boldsymbol{\rho}^{(i)}_{1:C,t}, \alpha^{(i)}_{1:C,t}, \tau^{(i)}_{1:C,t})\}_{i=1}^{N}$, the cluster assignments for the rankings $\{z^{(i)}_{1:M_t}\}_{i=1}^{N}$, the particle weights $\{w^{(i)}_t\}_{i=1}^{N}$ and the set of augmented rankings $\{\tilde{\mathbf{R}}^{(i)}_{1:M_t}\}_{i=1}^{N}$ which represents the target posterior distribution $p_t(\boldsymbol{\rho}_{1:C}, \alpha_{1:C}|\tilde{\mathbf{R}}_{1:M_t})$.

We would need to adapt the SMC clustering algorithm to determine how the missing components of the partial ranking are augmented to obtain a full ranking as well as the component label. Also, we would need to decide when data augmentation is performed: we can propose a component label before performing augmentation, or an augmentation before proposing a component label for the assessor's ranking.

In Chapter 6, we made an informed proposal for the cluster allocation for each new full ranking by considering the likelihood of allocating the ranking to each of the cluster components. Ideally, this stage would be incorporated into the proposed framework for clustering with partial rankings. One solution is to extend each particle multiple times by proposing all possible cluster assignments for the new observation, as in Fearnhead (2004). That is, we perform data augmentation on the new ranking for each cluster component before we then sample a component label and the matching auxiliary complete ranking. The component label is sampled by calculating the contribution of each augmented ranking to the likelihood of its respective component of the mixture. Then we sample the assigned component label multinomially using the normalised contributions as the multinomial probabilities. However, this means that when calculating the assignment probability, we would have to marginalise all of the possible augmentations, given the current estimated values of $\boldsymbol{\rho}$ and α for each cluster. This is computationally challenging.

8.1.2 Clustering with an unknown number of components

It is quite common for a multi-modal data set to not necessarily have a known number of components. Typically, the number of components is predetermined using the complete data set before performing clustering. However, within an SMC framework, in theory, we do not know the size of the "complete" data set as we are in an online setting. It would be appropriate to update the number of components on a sequential basis as well as the model parameter estimates of the mixture model posterior. At several points in time, we may find that it is appropriate to propose a new cluster component for the posterior given that a new observation may be better represented in the new component rather than be assigned to an existing cluster component.

Proposed model

We apply the DPM model to the SMC framework for the Mallows mixture model in Chapter 6. We consider for now the task of clustering full rankings on a sequential basis to update the number of components as well as the model parameter values in the SMC particles.

To account for the variable number of components across the particle, let $c_t^{(i)}$ be the total number of components in a particle $i \in \{1, \dots, N\}$ at time t. In this model, we assume that we now observe one new full ranking at a time, so the rankings and cluster assignment labels are also indexed by t. At iteration $t-1$, each one of N particles now contains: a set of the cluster components, $\boldsymbol{\theta}^{(i)} = \{\boldsymbol{\rho}^{(i)}_{1:c_{t-1}}, \alpha^{(i)}_{1:c_{t-1}}\}$, a set of cluster labels $z^{(i)}_{1:t-1}$, and particle weights $w^{(i)}_{t-1}$. At iteration t, we observe one new ranking at a time and the probability of assigning a cluster label to the observation given all previous cluster labels for each particle $p(z_t^{(i)}|z^{(i)}_{1:t-1})$ is defined by the recursion

$$p(z_t^{(i)} = j|z^{(i)}_{1:t-1}) = \begin{cases} n_j/(c^{(i)}_{t-1} + \alpha_{DPM}) & \text{for } j = 1, \dots, c^{(i)}_{t-1} \\ \alpha_{DPM}/(c^{(i)}_{t-1} + \alpha_{DPM}) & \text{for } j = c^{(i)}_{t-1} + 1 \end{cases},$$

where: n_j is the number of observations that $z_{1:t}$ assigns to cluster j and α_{DPM} is the concentration parameter in the DPM.

The SMC-DPM model extends the particle to each existing component and a potential new component. The probability of proposing the cluster label has the same likelihood,

$$\pi(z_t^{(i)}|z^{(i)}_{1:t-1}) = \frac{1}{c^{(i)}_{t-1} + 1}.$$

If the ranking is assigned to an existing cluster, then the incremental weight is simply

$$\tilde{w}_t^{(i)} = \frac{p(\mathbf{R}_t|\boldsymbol{\theta}_t^{(i)}, z_t^{(i)})}{c_{t-1}^{(i)} + 1}.$$

If, however, the ranking is assigned to a new component, then we need to select a distribution that proposes the model parameters of the new component. This would require further consideration as the choice of the proposal distribution would affect the incremental weight calculation for a particle.

The cluster label is sampled multinomially using the normalised likelihood contributions of proposing the ranking to each component. We can iterate between sampling allocations and resampling particles if we have a batch of new rankings to process before we can incorporate the move stage of the SMC algorithm by using MCMC move kernels to propose new values for each $\{\boldsymbol{\rho}_{1:c_t}^{(i)}, \alpha_{1:c_t}^{(i)}\}_{i=1}^{N}$. Jump parameters can also be incorporated into the move stage of the algorithm to reduce computational effort.

When using the DPM model for clustering, the number of components after T iterations will likely be higher than the number of components used to generate the multi-model data set. In addition, the final number of components could vary between particles. Further work will be required to decide how best to post-process the experiment output so that the clustering analysis remains meaningful.

Bibliography

C. E. Antoniak. Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems. *The Annals of Statistics*, 2:1152–1174, 1974.

D. Asfaw, V. Vitelli, Ø. Sørensen, E. Arjas, and A. Frigessi. Time-varying rankings with the Bayesian Mallows model. *The ISIs Journal for the Rapid Dissemination of Statistics Research*, 6(1):14–30, 2017.

P. Awasthi, A. Blum, O. Sheffet, and A. Vijayaraghavan. Learning mixtures of ranking models. *Advances in Neural Information Processing Systems*, 27:2609–2617, 2014.

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

C. Berzuini and W. Gilks. Resample-move filtering with cross-model jumps. In *Sequential Monte Carlo Methods in Practice*, pages 117–138. Springer-Verlag, New York, NY, USA, 2001.

C. Berzuini and W. R. Gilks. Particle filtering methods for dynamic and static Bayesian problems. In *Highly Structured Stochastic Systems*, pages 207–227. Oxford University Press, Oxford, UK, 2003.

C. Biernacki, G. Celeux, and G. Govaert. Assessing a mixture model for clustering with the integrated completed likelihood. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(7):719–725, 2000.

D. M. Blei and M. I. Jordan. Variational inference for Dirichlet process mixtures. *Journal of Bayesian Analysis*, 1(1):121–143, 2006.

R. A. Bradley and M. E. Terry. Rank analysis of incomplete block designs: I. The method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

R. Busa-Fekete, E. Hüllermeier, and B. Szörényi. Preference-based rank elicitation using statistical models: The case of Mallows. In *31st International Conference on Machine Learning*, pages 1071–1079. PMLR, 2014.

L. M. Busse, P. Orbanz, and J. M. Buhmann. Cluster analysis of heterogeneous rank data. In *Proceedings of the 24th International Conference on Machine learning*, pages 113–120, ACM, 2007. Corvallis, OR, USA.

O. Cappé, S. J. Godsill, and E. Moulines. An overview of existing methods and recent advances in sequential Monte Carlo. *Proceedings of the IEEE*, 95(5):899–924, 2007.

F. Caron and A. Doucet. Efficient Bayesian inference for generalized Bradley-Terry models. *Journal of Computational and Graphical Statistics*, 21(1):174–196, 2012.

J. Carpenter, P. Clifford, and P. Fearnhead. Improved particle filter for nonlinear problems. *IEE Proceedings - Radar, Sonar and Navigation*, 146(1):2–7, 1999.

C. M. Carvalho, H. F. Lopes, N. G. Polson, and M. A. Taddy. Particle learning for general mixtures. *Bayesian Analysis*, 5(4):709–740, 2010.

F. Chierichetti, A. Dasgupta, R. Kumar, and S. Lattanzi. On learning mixture models for permutations. In *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science*, pages 85–92. Association for Computing Machinery, 2015.

F. Chierichetti, A. Dasgupta, S. Haddadan, R. Kumar, and S. Lattanzi. Mallows models for top-k lists. *Advances in Neural Information Processing Systems*, 31: 4387–4397, 2018.

N. Chopin. A sequential particle filter method for static models. *Biometrika*, 89(3): 539–552, 2002.

M. Crispino and I. Antoniano-Villalobos. Informative priors for the consensus ranking in the bayesian mallows model. *Bayesian Analysis*, 1(1):1–24, 2022.

M. Crispino, E. Arjas, V. Vitelli, and A. Frigessi. Recommendation from intransitive pairwise comparisons. In *Poster Proceedings of the ACM Recommender Systems Conference*, ACM, 2016. Corvallis, OR, USA.

M. Crispino, E. Arjas, V. Vitelli, N. Barrett, and A. Frigessi. A Bayesian Mallows approach to nontransitive pair comparison data: How human are sounds? *The Annals of Applied Statistics*, 13(1):492–519, 2019.

D. E. Critchlow, M. A. Fligner, and J. S. Verducci. Probability models on rankings. *Journal of Mathematical Psychology*, 35(3):294–318, 1991.

P. Del Moral, A. Doucet, and A. Jasra. Sequential Monte Carlo samplers. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(3):411–436, 2006.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22, 1977.

P. Diaconis. *Group representations in probability and statistics*, volume 11. Institute of Mathematical Statistics, Hayward, CA, USA, 1988.

J.-P. Doignon, A. Pekeč, and M. Regenwetter. The repeated insertion model for rankings: Missing link between two subset choice models. *Psychometrika*, 69(1): 33–54, 2004.

R. Douc and O. Cappé. Comparison of resampling schemes for particle filtering. In *Proceedings of the 4th International Symposium on Image and Signal Processing and Analysis (ISPA)*, pages 64–69. IEEE, 2005.

A. Doucet and A. M. Johansen. A tutorial on particle filtering and smoothing: Fifteen years later. *Handbook of nonlinear filtering*, 12(3):656–704, 2009.

A. Doucet, N. De Freitas, and N. Gordon. An introduction to sequential Monte Carlo methods. In *Sequential Monte Carlo Methods in Practice*, pages 3–14. Springer-Verlag, New York, NY, USA, 2001.

P. Fearnhead. Markov chain Monte Carlo, sufficient statistics, and particle filters. *Journal of Computational and Graphical Statistics*, 11(4):848–862, 2002.

P. Fearnhead. Particle filters for mixture models with an unknown number of components. *Statistics and Computing*, 14(1):11–21, 2004.

M. A. Fligner and J. S. Verducci. Distance based ranking models. *Journal of the Royal Statistical Society: Series B (Methodological)*, 48(3):359–369, 1986.

C. Fraley and A. E. Raftery. How many clusters? Which clustering method? Answers via model-based cluster analysis. *The Computer Journal*, 41(8):578–588, 1998.

B. J. Frey and D. Dueck. Clustering by passing messages between data points. *Science*, 315(5814):972–976, 2007.

S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:721–741, 1984.

J. Geweke. Bayesian inference in Econometric models using Monte Carlo integration. *Econometrica: Journal of the Econometric Society*, 57:1317–1339, 1989.

S. Godsill and T. Clapp. Improvement strategies for Monte Carlo particle filters. In *Sequential Monte Carlo Methods in Practice*, pages 139–158. Springer-Verlag, New York, NY, USA, 2001.

P. K. Goel, L. Lang, and B. Bakshi. Sequential Monte Carlo in Bayesian inference for dynamic models: An overview. *Bayesian Statistics and Its Applications, Co-sponsored by International Society for Bayesian Analysis*, 2005.

N. J. Gordon, D. J. Salmond, and A. F. Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Proceedings - F (Radar and Signal Processing)*, 140(2):107–113, 1993.

I. C. Gormley and T. B. Murphy. Analysis of Irish third-level college applications data. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, 169 (2):361–379, 2006.

E. Gregory. RMallow: Fit multi-modal Mallows models to ranking data. `https://CRAN.R-project.org/package=RMallow`, 2012. Accessed: 2022-11-24.

GroupLens. Movielens dataset. `https://grouplens.org/datasets/movielens/`. Accessed: 2022-11-24.

J. Guiver and E. Snelson. Bayesian inference for Plackett-Luce ranking models. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 377–384. ACM, 2009.

W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57:97–109, 1970.

D. R. Hunter. MM algorithms for generalized Bradley-Terry models. *The Annals of Statistics*, 32(1):384–406, 2004.

E. Irurozki, B. Calvo Molinos, and J. A. Lozano Alonso. Sampling and learning the Mallows and weighted Mallows models under the Hamming distance. Technical report, University of the Basque Country, San Sebastian, Spain, January 2014a.

E. Irurozki, J. Ceberio Uribe, B. Calvo Molinos, and J. A. Lozano Alonso. Sampling and learning the Mallows model under the Ulam distance. Technical report, University of the Basque Country, San Sebastian, Spain, January 2014b.

E. Irurozki, B. Calvo Molinos, and J. Lozano Alonso. Permallows: An R package for Mallows and generalized Mallows models. *Journal of Statistical Software*, 71(12): 1–30, 2016.

E. Irurozki, B. Calvo Molinos, and J. Lozano Alonso. Sampling and learning Mallows and generalized Mallows models under the Cayley distance. *Methodology and Computing in Applied Probability*, 20(1):1–35, 2018.

A. Jasra, C. C. Holmes, and D. A. Stephens. Markov chain Monte Carlo methods and the label switching problem in Bayesian mixture modeling. *Statistical Science*, 20(1):50–67, 2005.

R. E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45, 1960.

T. Kamishima. Nantonac collaborative filtering: Recommendation based on order responses. In *Proceedings of the Ninth International Conference on Knowledge Discovery and Data Mining*, pages 583–588, 2003.

N. Kantas, A. Doucet, S. S. Singh, and J. M. Maciejowski. An overview of sequential Monte Carlo methods for parameter estimation in general state-space models. *IFAC Proceedings Volumes*, 42(10):774–785, 2009.

N. Kantas, A. Doucet, S. Singh, J. Maciejowski, and N. Chopin. On particle methods for parameter estimation in state-space models. *Statistical Science*, 30(3):328–351, 2015.

G. Kitagawa. Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. *Journal of Computational and Graphical Statistics*, 5(1):1–25, 1996.

A. Kong, J. S. Liu, and W. H. Wong. Sequential imputations and Bayesian missing data problems. *Journal of the American Statistical Association*, 89(425):278–288, 1994.

Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

G. Lebanon and Y. Mao. Non-parametric modeling of partially ranked data. *Journal of Machine Learning Research*, 9(79):2401–2429, 2008.

P. H. Lee and P. L. Yu. An R package for analyzing and modeling ranking data. *BMC Medical Research Methodology*, 13(1):1–11, 2013.

A. Liu and A. Moitra. Efficiently learning mixtures of Mallows models. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science*, pages 627–638, 2018.

J. Liu and M. West. Combined parameter and state estimation in simulation-based filtering. In *Sequential Monte Carlo Methods in Practice*, pages 197–223. Springer-Verlag, New York, NY, USA, 2001.

J. S. Liu. Nonparametric hierarchical Bayes via sequential imputations. *The Annals of Statistics*, 24(3):911–930, 1996.

J. S. Liu. *Monte Carlo Strategies in Scientific Computing*. Springer-Verlag, New York, NY, USA, 2001.

J. S. Liu and R. Chen. Sequential Monte Carlo methods for dynamic systems. *Journal of the American Statistical Association*, 93(443):1032–1044, 1998.

Q. Liu, M. Crispino, I. Scheel, V. Vitelli, and A. Frigessi. Model-based learning from preference data. *Annual Reviews of Statistics and Its Application*, 6(1):329–354, 2019a.

Q. Liu, A. H. Reiner, A. Frigessi, and I. Scheel. Diverse personalized recommendations with uncertainty from implicit preference data with the Bayesian Mallows model. *Knowledge-Based Systems*, 186:104960, 2019b.

T. Lu and C. Boutilier. Effective sampling and learning for Mallows models with pairwise-preference data. *The Journal of Machine Learning Research*, 15(1):3783–3829, 2014.

R. D. Luce. *Individual choice behaviour: A theoretical analysis.* Wiley, New York, NY, USA, 1959.

S. N. MacEachern, M. Clyde, and J. S. Liu. Sequential importance sampling for nonparametric Bayes models: The next generation. *Canadian Journal of Statistics*, 27(2):251–267, 1999.

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

B. Mandhani and M. Meila. Tractable search for learning exponential models of rankings. *Journal of Machine Learning Research*, 5:392–399, 2009.

J. Marden. *Analyzing and Modeling Rank Data*. Chapman & Hall, Cambridge, MA, USA, 1995.

G. J. McLachlan, S. X. Lee, and S. I. Rathnayake. Finite mixture models. *Annual Review of Statistics and Its Application*, 6:355–378, 2019.

M. Meilă and L. Bao. An exponential model for infinite rankings. *Journal of Machine Learning Research*, 11:3481–3518, 2010.

M. Meilă and H. Chen. Dirichlet process mixtures of generalized Mallows models. pages 358–367, Corvallis, OR, USA, 2010. AUAI Press.

N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953.

T. Minka. Power EP. Technical Report MSR-TR-2004-149, Microsoft Research, Cambridge, UK, October 2004.

C. Mollica and L. Tardella. PLMIX: An R package for modelling and clustering partially ranked data. *Journal of Statistical Computation and Simulation*, 90(5): 925–959, 2020.

S. Mukherjee. Estimation in exponential families on permutations. *The Annals of Statistics*, 44(2):853–875, 2016.

T. B. Murphy and D. Martin. Mixtures of distance-based models for ranking data. *Computational Statistics & Data Analysis*, 41(3-4):645–655, 2003.

C. A. Naesseth, F. Lindsten, and T. B. Schn. Elements of sequential Monte Carlo. *Foundations and Trends in Machine Learning*, 12(3):307–392, 2019.

R. M. Neal. Annealed importance sampling. *Statistics and Computing*, 11(2):125–139, 2001.

P. Papastamoulis. label.switching: An R package for dealing with the label switching problem in MCMC outputs. *Journal of Statistical Software*, 69(1), 2016.

D. Peel and G. MacLahlan. *Finite Mixture Models*. Wiley, New York, NY, USA, 2000.

M. K. Pitt and N. Shephard. Filtering via simulation: Auxiliary particle filters. *Journal of the American Statistical Association*, 94(446):590–599, 1999.

R. L. Plackett. The analysis of permutations. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 24(2):193–202, 1975.

Z. Qian and L. Philip. Weighted distance-based models for ranking data using the R package rankdist. *Journal of Statistical Software*, 90(1):1–31, 2019.

R. A. Redner and H. F. Walker. Mixture densities, maximum likelihood and the EM algorithm. *SIAM Review*, 26(2):195–239, 1984.

C. Robert and G. Casella. *Monte Carlo statistical methods.* Springer Science & Business Media, New York, NY, USA, 2004.

A. Sawadogo, D. Lafon, and S. Dossou-Gbété. Combining MM-algorithms and MCMC procedures for maximum likelihood estimation in Mallows-Bradley-Terry models. *Journal of Mathematics and Statistical Science*, 5(5):106–128, 2019.

B. B. Smith. Discussion of Professor Ross's paper. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 12(1):41–59, 1950.

Ø. Sørensen, M. Crispino, Q. Liu, and V. Vitelli. BayesMallows: An R package for the Bayesian Mallows model. *The R Journal*, 12(1):324–342, 2020.

M. Sperrin, T. Jaki, and E. Wit. Probabilistic relabelling strategies for the label switching problem in Bayesian mixture models. *Statistics and Computing*, 20(3): 357–366, 2010.

M. Stephens. Dealing with label switching in mixture models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 62(4):795–809, 2000.

H. Stern. Models for distributions on permutations. *Journal of the American Statistical Association*, 85(410):558–564, 1990.

G. Storvik. Particle filters for state-space models with the presence of unknown static parameters. *IEEE Transactions on Signal Processing*, 50(2):281–289, 2002.

J. Stoyanovich, L. Ilijasic, and H. Ping. Workload-driven learning of Mallows mixtures with pairwise preference data. In *Proceedings of the 19th International Workshop on Web and Databases*, pages 1–6, San Francisco, CA, USA, 2016. ACM.

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

Y. Ulker, B. Günsel, and T. Cemgil. Sequential Monte Carlo samplers for Dirichlet process mixtures. *Journal of Machine Learning Research*, 9:876–883, 2010.

V. Vitelli, Ø. Sørensen, M. Crispino, A. Frigessi, and E. Arjas. Probabilistic preference learning with the Mallows rank model. *Journal of Machine Learning Research*, 18 (1):5796–5844, 2018.

G. Yao and U. Böckenholt. Bayesian estimation of Thurstonian ranking models based on the Gibbs sampler. *British Journal of Mathematical and Statistical Psychology*, 52(1):79–92, 1999.

P. L. Yu, J. Gu, and H. Xu. Analysis of ranking data. *Wiley Interdisciplinary Reviews: Computational Statistics*, 11(6):e1483, 2019.

E. Zermelo. Die berechnung der turnier-ergebnisse als ein maximumproblem der wahrscheinlichkeitsrechnung. *Mathematische Zeitschrift*, 29(1):436–460, 1929.

A. Ziegler, E. Christiansen, D. Kriegman, and S. Belongie. Locally uniform comparison image descriptor. *Advances in Neural Information Processing Systems*, 25, 2012.