NONPARAMETRIC BAYESIAN CLUSTERING UNDER STRUCTURAL RESTRICTIONS

by

Hanxi Sun

A Dissertation

Submitted to the Faculty of Purdue University

In Partial Fulfillment of the Requirements for the degree of

Doctor of Philosophy

Department of Statistics

West Lafayette, Indiana

August 2021

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF COMMITTEE APPROVAL

Dr. Vinayak Rao, Chair

Department of Statistics

Dr. Heejung Shim

School of Mathematics and Statistics, University of Melbourne

Dr. Hao Zhang

Department of Statistics

Dr. Xiao Wang

Department of Statistics

Dr. Qifan Song

Department of Statistics

Approved by:

Dr. Jun Xie

To my parents, Huixia Han and Bo Sun.

ACKNOWLEDGMENTS

First and foremost, I want to express my sincere gratitude towards my advisor, Dr. Vinayak Rao, who guided me into the beautiful world of nonparametric Bayesian analysis. He has always been knowledgeable and supportive during my time at Purdue. I have learned so much from him. He taught me how to think critically and independently, as well as present my ideas accurately and confidently.

I have been fortunate to work with Dr. Heejung Shim. She introduced me to biostatistics, and her dedication to our collaborative work is greatly appreciated. Without her help, my research would not have gone this far. I am also grateful to my other committee members, Dr. Hao Zhang, Dr. Xiao Wang, and Dr. Qifan Song, for their insightful comments on this dissertation and supports during my graduate study.

My gratitude extends beyond my committee to my other collaborators. Dr. Guang Cheng and Tianyang Hu brought me into the exciting field of deep learning and generative modeling. Dr. Thibaud Coroller, Dr. Mark Baillie, and Jason Plawinski accompanied me to explore the potential of these models in privacy protection. Although this line of work is not included in the dissertation, I very much enjoyed it.

I appreciate the chance to serve as a consultant in the department. Dr. Bruce Craig, Dr. Arman Sabbaghi, and Dr. Gu Chong helped me with many projects, and I am thankful for being trusted by all my clients.

I will always remember the quality time I have spent with many wonderful people in the past five years, especially during the last year and a half under the pandemic. I appreciate the help from the department staff, Ce-Ce, Doug, Patti, Mary, and Holly. Thank you to all of my fellow students, Cheng Li, Boqian Zhang, Jiasen Yang, Sophie Sun, Yao Chen, Jincheng Bai, Jiapeng Liu, Qi Wang, Yumin Zhang, Meng Deng, Wenbin Zhu, Fan Wu, Wei Hao, Hakeem Wahab, and Zhanyu Wang, Ryan Murphy, Bingjing Tang, Xinyi Pei, Imon Banerjee, and Kent Gauen. In particular, I am thankful to Jin Fang for her encouragement during challenging times, and to Tianyang Hu, thank you for being my best friend throughout the journey.

Finally, I would like to thank my family for their unconditional love and support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

i.i.d.	independently and identically distributed
MCMC	Markov chain Monte Carlo
MRMM	Matérn repulsive mixture model
CRM	completely random measure
DP	Dirichlet process
HDP	hierarchical Dirichlet process
HPYP	hierarchical Pitman-Yor process
CRP	Chinese restaurant process
CRF	Chinese restaurant franchise
ESS	Effective Sample Size
TV	Total variation
ROC	receiver operating characteristic
AUC	area under curve
CTL	cytotoxic T-lymphocyte
HLA	human leukocyte antigen

ABSTRACT

Model-based clustering, with its flexibility and solid statistical foundations, is an important tool for unsupervised learning, and has numerous applications in a variety of fields. This dissertation focuses on nonparametric Bayesian approaches to model-based clustering under *structural restrictions*. These are additional constraints on the model that embody prior knowledge, either to regularize the model structure to encourage interpretability and parsimony or to encourage statistical sharing through underlying tree or network structure.

The first part in the dissertation focuses on the most commonly used model-based clustering models, mixture models. Current approaches typically model the parameters of the mixture components as independent variables, which can lead to overfitting that produces poorly separated clusters, and can also be sensitive to model misspecification. To address this problem, we propose a novel Bayesian mixture model with the structural restriction being that the clusters repel each other. The repulsion is induced by the generalized Matérn type-III repulsive point process. We derive an efficient Markov chain Monte Carlo (MCMC) algorithm for posterior inference, and demonstrate its utility on a number of synthetic and real-world problems.

The second part of the dissertation focuses on clustering populations with a hierarchical dependency structure that can be described by a tree. A classic example of such problems, which is also the focus of our work, is the phylogenetic tree with nodes often representing biological species. The structure of this problem refers to the hierarchical structure of the populations. Clustering of the populations in this problem is equivalent to identify branches in the tree where the populations at the parent and child node have significantly different distributions. We construct a nonparametric Bayesian model based on hierarchical Pitman-Yor and Poisson processes to exploit this, and develop an efficient particle MCMC algorithm to address this problem. We illustrate the efficacy of our proposed approach on both synthetic and real-world problems.

1. INTRODUCTION

1.1 Model-based clustering

Cluster analysis or clustering is a fundamental unsupervised learning tool for data exploration and analysis that aims to identify groups of homogeneous objects within the data. It has been extensively studied for decades [1]–[4] and has found wide applications in multiple disciplines, including topic modeling [5], [6], genetics [7]–[9], computer vision [10], and pattern recognition [11]. Early clustering approaches, such as the K-means algorithm [2], [12], are mainly based on heuristic or geometric procedures that rely heavily on similarity measures and symmetry assumptions. For modern clustering problems, the model-free nature can make such methods extremely limited. Further, these often do not possess the capacity to handle data with complicated underlying structures, and it is hard to incorporate prior knowledge and missing data into the clustering process. Lastly, there are no mathematically concrete model assessment and model selection criteria available.

The development of probabilistic models in cluster analysis dates back at least to the work of Wolfe [13] in 1963. In the literature, such approaches are often referred to as model-based clustering methods. With enhanced capacity, interpretability, and the ability to provide statistical insights into the model fit, such approaches have since become increasingly popular in both practice and academic research [14]–[19]. In particular, thanks to recent advancements in technology, computationally-intensive Bayesian clustering methods, especially those based on finite and infinite mixture models, have shown promising results in a wide range of applications (see, for instance, Yeung, Fraley, Murua, *et al.* [20], Fraley and Raftery [21], and Melnykov, Maitra, *et al.* [22]).

1.2 Structural restrictions in clustering problems

Model-based clustering through mixture models is a powerful class of models with great flexibility in characterizing sub-populations in the data, capable of approximating increasingly complex distributions as the number of mixture components increases. However, the flexibility of mixture models often comes at the cost of interpretability and parsimony. For

computational tractability, the parameters of the mixture components (the cluster parameters) are typically modeled as independent and identically distributed draws from some "base distribution". This can result in overlapping clusters, i.e. unless the clusters are very widely separated, the posterior will typically assign probability to multiple clusters in some neighborhood. The nearly identical locations of these clusters leads to redundancy, and lack of interpretability, as large interpretable clusters are broken into smaller meaningless groups. Since mixture models are typically composed of simple parametric components, even if the data exhibits clear clustered structure, any deviation of individual clusters from the parametric form will again result in overlapping components. Rousseau and Mengersen [23] discussed the overfitting behavior in asymptotics for finite mixture models, and Miller and Harrison [24] and Miller and Harrison [25] demonstrated the inconsistency in number of components for the most commonly used infinite mixture model, Dirichlet process mixtures [26], [27], and its extension, Pitman-Yor process [28], [29] mixtures.

To directly address the lack of interpretability, we propose to introduce a *structural restriction* to the model. Rather than being sampled independently from the base measure, cluster locations are jointly sampled from a prior distribution that penalizes realizations where clusters are situated too close to each other. We refer to such restrictions as *structural restrictions*, and the corresponding priors as *repulsive priors*. This serves as a means of regularization, to enforce separability between clusters and hence improve interpretability.

In the first part of the dissertation, we focus on the problem of clustering observations with a repulsive mixture model. To be specific, motivated by the work of Rao, Adams, and Dunson [30], we proposed a nonparametric Bayesian framework that introduces repulsion between clusters with a generalized Matérn type-III repulsive point process model [31], [32], obtained through a dependent sequential thinning scheme on a primary Poisson point process [33]. We develop a novel and efficient Gibbs sampler for posterior inference of the proposed Matérn repulsive mixture model (MRMM). We provide a collection of synthetic and real data studies to demonstrate the flexibility of MRMM and the superiority of our proposed algorithm. Our algorithm achieves comparable performance (goodness-of-fit) with fewer mixture components and is proven to be robust to model misspecifications. We also updated

the derivation of an essential Gibbs update step in Rao, Adams, and Dunson [30] using Campbell's theorem [33]. This part of work is included in Sun, Zhang, and Rao [34].

Most of the cluster analysis tools focus on clustering of individual instance/observation. However, there is another type of application where the goal is to group *populations* with common features (see, for instance, Nielsen, Nock, and Amari [35] and Henderson, Gallagher, and Eliassi-Rad [36]). Take topic modeling as an example. Documents are often regarded as collections of words (bag-of-word assumption) [5], and the quantity of interest is document-level similarities rather than that among words. If we extend the definition and in turn treat documents as populations with different word frequencies, clustering documents will then become a problem of clustering populations, a problem that seeks to group those with similar word distributions together.

In the second part of the dissertation, we extend our analysis to population clustering with structural constraints. Modern datasets often possess rich underlying structures, originating from their mechanistic, spatio-temporal generative processes. Trees are a widely used structures, representing a hierarchical organization of observations into partially overlapping sets at multiple granularities. A classic example, and one that is the focus of our work, are phylogenetic trees, showing relationships between various entities evolving from a common ancestor. The entities in a phylogenetic tree are typically biological species, though we take a broader view, and also consider evolving languages and other social norms. Internal and leaf nodes of the tree represent different populations (distributions), and one or multiple i.i.d. observations are obtained from some nodes in the tree. In this work, we focus on the case when the populations are characterized by discrete distributions, and the goal is to cluster populations with similar distributions. In this problem, the *structural restriction* refers to the natural dependency structure described by the tree. Naturally, the clustering problem can be rephrased into detecting branches where the distributions at the parent and child node significantly differ form each other. Motivated by the work of Ansari and Didelot [37], we develop a nonparametric Bayesian model with hierarchical Pitman-Yor process (HPYP) that takes advantages of a convenient marginalization property that it possess to enable efficient computation. A novel particle MCMC [38] algorithm is developed for the posterior

inference, and a number of synthetic and read empirical experiments are conducted to show the efficacy of our proposed approach.

In summary, *structural restrictions* in clustering problems can either serve as a mean of regularization to encourage interpretability and parsimony, or reflect the prior knowledge or underlying structure of the problem. This dissertation focuses on both of these.

1.3 Dissertation Organization

We organize the rest of the dissertation as follows. Chapter 2 provides a brief review of a few preliminary topics, including mixture models, Dirichlet processes, hierarchical Dirichlet processes (HDPs), Chinese restaurant processes (CRPs), Chinese restaurant franchises (CRFs), and Pitman-Yor processes.

In chapter 3, we start with an introduction to Matérn type-III repulsive point processes. Then we describe our first main contribution, the Matérn repulsive mixture model (MRMM), and the inference algorithm in section 3.2 and 3.3, respectively. For simplicity, when no confusion can be raised, we will use MRMM to refer to the model and the inference method interchangeably. Chapter 4 evaluates the performance of the Python3 package `mrmm` we developed for MRMM inference. We study the effect of hyperparameters in the model, and also apply MRMM to various real-world tasks, including clustering on a torus (protein structural data) to show the flexibility of the model and two additional comparisons with existing repulsive mixture models to illustrate the superior performance of our model.

Starting from chapter 5, we switch to our second main contribution, clustering populations with a hierarchical structure. The proposed model is described in detail in section 5.1, and section 5.2 fully outlines the particle MCMC inference algorithm (algorithm 5). Chapter 6 includes a collection of empirical results, including synthetic studies on the identifiability and robustness to model misspecification of our approach, and real data analyses to demonstrate the efficacy and practicality of our algorithm.

We conclude in chapter 7 with a summary of the dissertation and a discussion of potential future directions.

2. PRELIMINARIES

2.1 Mixture models

Mixture model is a powerful and flexible class of models, capable of approximating increasingly complex distributions as the number of mixture components increases. Such models are useful both in density modeling applications [39], as well as in clustering applications [14], [16], [40]. In this section, we will briefly introduce finite mixture models mainly as a reference to our discussions later in the dissertation. A more comprehensive review can be found in McLachlan, Lee, and Rathnayake [41].

Denote n random samples from a finite mixture model with C components as $\boldsymbol{X} = (x_1, \ldots, x_n)$. For an observation x_i, there is a latent cluster assignment $z_\mathrm{i} \in \{1, \ldots, C\}$ associating with it. Assume the mixture components come from a family of probabilistic distributions $p_{\mathcal{X}}(\cdot\,; \theta)$ parameterized by $\theta \in \Theta$. Write $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_C)^\top$ for the collection of parameters for mixture components, and let $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_C)^\top$ be the corresponding mixture weights, such that $\pi_\mathrm{j} \in [0, 1]$ and $\sum_\mathrm{j} \pi_\mathrm{j} = 1$. Then the generative model of the data $\boldsymbol{X}$ is given by

$$\begin{aligned} z_\mathrm{i} \,|\, \boldsymbol{\pi} &\overset{\text{i.i.d.}}{\sim} \text{Multinomial}\,(\pi_1, \ldots, \pi_C) \qquad \mathrm{i} = 1, \ldots, n \\ x_\mathrm{i} \,|\, z_\mathrm{i}, \boldsymbol{\theta} &\overset{\text{i.i.d.}}{\sim} p_{\mathcal{X}}\,(\cdot\,; \theta_{z_\mathrm{i}}) \end{aligned} \tag{2.1}$$

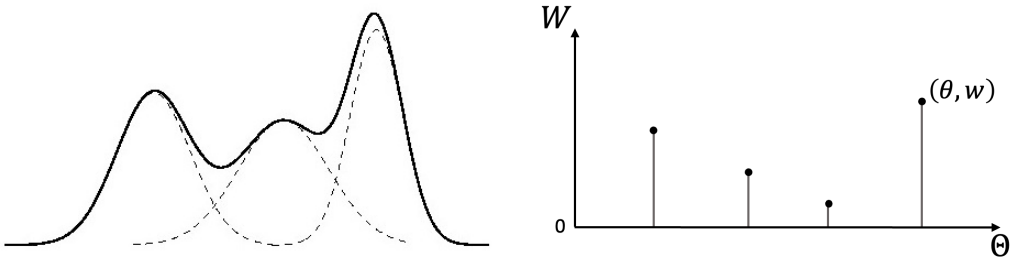

Figure 2.1. (left) An illustration of a Gaussian mixture model with three mixture components. **(right)** Parameters for mixture components θ and mixture weights w as a collection of points in the product space $\Theta \times \mathcal{W}$. Although the parameter space Θ is illustrated as one dimensional, it can represent a higher dimensional space. The mixture weights, on the other hand, are positive (non-negative) real numbers, i.e., $\mathcal{W} = \mathbb{R}^+$.

In Bayesian framework, priors are placed on $\boldsymbol{\pi}$ and $\boldsymbol{\theta}$. For the mixture weights, a common choice of the prior is the conjugate flat Dirichlet distribution $\boldsymbol{\pi} \sim \text{Dirichlet}(\alpha/C, \ldots, \alpha/C)$. For the mixture parameter θ_{j}'s, conjugate prior, if exists, are often preferred. For instance, in a Gaussian mixture model where $p_{\mathcal{X}}(\cdot\,; \theta)$ represents a unit Gaussian distribution centered at θ, the conjugate prior of θ is also a Gaussian distribution. With conjugate priors, the posterior inference is straightforward.

Let us further explore the prior on the mixture weights. Notice that Dirichlet distribution can be constructed from normalizing independent Gamma random variables [42]. This suggests that instead of using the normalized weights $\boldsymbol{\pi}$, we could instead use a collection of unnormalized weights $\boldsymbol{w} = (w_1, \ldots, w_C)^\top$ where $w_{\text{j}} \geq 0$ and $\pi_{\text{j}} = w_{\text{j}} / \sum_{\text{j}} w_{\text{j}}$. Then the flat Dirichlet prior placed over the mixture proportions is equivalent to independent Gamma$(\alpha, 1)$ priors on the unnormalized weight w_{j}'s. This leads to an i.i.d. prior on the parameter and weights pairs $(\theta_{\text{j}}, w_{\text{j}})$ of mixture components. Figure 2.1 (right) illustrates how to represent a mixture model with a collections of points in the product space $\Theta \times \mathcal{W}$, where $\mathcal{W}$ denotes the positive half of the real line.

When the number of mixture components C is unknown, one might place a Poisson distribution over the number of components. As mixture models can be defined with a collection of points on $\Theta \times \mathcal{W}$, this prior is equivalent to have a Poisson point process [33] prior over the collection of pairs $\{(\theta_1, w_1), \ldots, (\theta_C, w_C))\}$. This specifies a *completely random measure* [43]. Let $\mathcal{X}$ be a complete and separable metric space endowed with the Borel σ-field $\mathcal{B}(\mathcal{X})$. A completely random measure (CRM) μ is a random element taking values on the space of boundedly finite measures on $\mathcal{X}$ such that, for any disjoint measurable subsets $A_1, \ldots, A_n$ in $\mathcal{B}(\mathcal{X})$, with $A_{\text{i}} \cap A_{\text{j}} = \emptyset$ for $\text{i} \neq \text{j}$, the random variables $\mu(A_1), \ldots, \mu(A_n)$ are mutually independent [44]. This nature of Poisson point processes motivates our work on repulsive mixture models in section 3.2.

2.2 Dirichlet Processes

Dirichlet processes (DPs) [26] are widely used as a nonparametric model that provides a measure on all (discrete) distributions. It is parameterized by a concentration parameter

$\alpha > 0$ and a base distribution H on some space $\mathcal{X}$. We write G for a realization of DP, $G \sim \text{DirichletProcess}(\alpha, H)$, then G is almost surely a discrete distribution with infinite number of components on $\mathcal{X}$. The base distribution H serves as the "center" for generating random measure G, and the concentration parameter α describes how close the sample G would be to the "center". Specifically, for any H-measurable set A,

$$\begin{aligned}\mathbb{E}[G(A)] &= H(A)\\ \text{Var}(G(A)) &= \frac{H(A)(1-H(A))}{1+\alpha}.\end{aligned}$$

In practice, DP is often used to model the underlying distribution of observations $x_{\text{i}} \mid G \sim G$, $\text{i} = 1, 2, \ldots$. There are two approaches to obtain i.i.d. sample x_{i}'s from the generative process: simulating from the stick-breaking representation of G [45] or sequentially producing observations with G marginalized out through the Chinese restaurant process (CRP).

Stick-breaking representation

An instantiation G from $\text{DirichletProcess}(\alpha, H)$ can be constructed with the following stick-breaking process.

$$\begin{aligned}\tilde{\beta}_{\text{j}} &\overset{\text{i.i.d.}}{\sim} \text{Beta}(1, \alpha) \qquad \text{j} = 1, 2, \ldots\\ \beta_{\text{j}} &:= \tilde{\beta}_k \prod_{\text{j}=1}^{\text{j}-1} \left(1 - \tilde{\beta}_{\text{j}}\right)\\ y_{\text{j}} &\overset{\text{i.i.d.}}{\sim} H\\ G &:= \sum_{\text{j}=1}^{\infty} \beta_{\text{j}} \delta_{y_{\text{j}}}\end{aligned} \tag{2.2}$$

where δ_y is the Dirac distribution that has probability 1 to be y. Truncated stick-breaking representation of G serves as an approximation to G, and simulating observations $x \sim G$ is straightforward, $\mathbb{P}\{x = y_{\text{j}}\} = \beta_{\text{j}}$.

Chinese restaurant process

A sequence of i.i.d. samples $x_{\mathrm{j}} \sim G$, $\mathrm{j} = 1, 2, \ldots$, can be generated from the Chinese restaurant process (CRP) associated with the DP [46] with the distribution G marginalized out. To start the process, the first sample x_1 is simulated directly from the base measure H. At step j when j samples $\{x_1, \ldots, x_{\mathrm{j}}\}$ have been generated, the next sample $x_{\mathrm{j}+1}$ is going to be simulated from the following distribution,

$$x_{\mathrm{j}+1} \,|\, \{x_1, \ldots, x_{\mathrm{j}}\},\ \alpha,\ H \sim \sum_{\tilde{\mathrm{j}}=1}^{\mathrm{j}} \frac{1}{\alpha + \mathrm{j}} \delta_{x_{\tilde{\mathrm{j}}}} + \frac{\alpha}{\alpha + \mathrm{j}} H. \tag{2.3}$$

With probability $\mathrm{j}/(\alpha + \mathrm{j})$ it will randomly take the value of an existing sample, and with probability $\alpha/(\alpha + \mathrm{j})$, it will take a new value directly from the base measure. In the "restaurant" context, we use "tables" to refer to the clusters where samples take value from the same existing sample. The table assignment of sample x_{j} is denoted by t_{j}, and it is defined as the index of the first sample in the cluster x_{j} joins. We say that the $(\mathrm{j}+1)$-st sample joins an existing table with probability proportional to the table size (the number of samples assigned to the table), and joins a new (empty) table with probability proportional to α. With more and more samples being generated, there is a "rich gets richer" effect that reinforces the larger clusters (tables). The more a sample is drawn, the more likely it will be drawn in the future.

Furthermore, although the observations are generated sequentially, CRP is actually invariant under permutations, which is formally known as the *exchangeability*. This property, along with the fact that CRP marginalizes out instantiation of the distribution G, makes CRP a computational efficient generative process for the observations.

2.3 Hierarchical Dirichlet process (HDP)

Teh, Jordan, Beal, *et al.* [47] proposes the hierarchical Dirichlet process (HDP) to extend DPs to model a collection of distributions with a hierarchical dependency structure, such as a tree. The idea is to model the child distribution with a DP centered at the distribution its parent. Figure 2.2 shows a simple hierarchical relationship between three populations

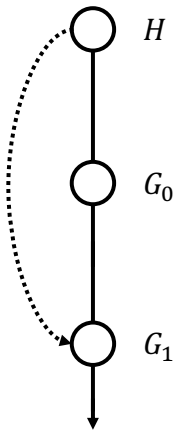

Figure 2.2. Illustration of an example of HDP model.

(distributions), the base measure H, and its offspring G_0 and G_1. The corresponding HDP model is given by

$$\begin{aligned} G_1 \,|\, H &\sim \text{DirichletProcess}\,(\alpha, H) \\ G_2 \,|\, G_1 &\sim \text{DirichletProcess}\,(\alpha, G_1) \end{aligned} \tag{2.4}$$

Following from the construction of CRP in equation (2.3), Teh, Jordan, Beal, *et al.* [47] develops the Chinese restaurant franchise (CRF) as coupling of CRPs, which is also a exchangeable process that integrates out all the distributions in the generative process and sequentially obtains i.i.d. observations from the HDP model. The idea of CRF is to construct observations at the child node iteratively as a CRP from a sequence of i.i.d. observations from the parent node. Consider the HDP model described in figure 2.2 and equation (2.4) as an example. Let $x_{(\mathrm{i,j})}$ denote the j-th sample of G_i, $\mathrm{i} = 0, 1$. Due to the exchangeability of samples, the actual sequence of generating those samples does not affect the correctness of the process. Let us start with $x_{(1,1)}$, the first sample of the child node G_1. According to CRP, it is generated directly from the base measure of G_1, i.e., $x_{(1,1)} \sim G_0$. As there is no sample at G_0 at this moment, according to CRP, simulating the first sample from G_0 ($x_{(0,1)}$) is equivalent to generate a sample from the base H. Therefore, the initial step of this

generative process simulates $x_{(2,1)} = x_{(1,1)} \sim H$. Then consider the distribution of $x_{(1,\mathrm{j}+1)}$ conditioning on the previous samples $x_{(1,1)}, \ldots, x_{(1,\mathrm{j})}$ and G_0. Following equation (2.3),

$$x_{(1,\mathrm{j}+1)} \,\Big|\, \{x_{(1,1)}, \ldots, x_{(1,\mathrm{j})}\},\ \alpha,\ G_0 \sim \sum_{\tilde{\mathrm{j}}=1}^{\mathrm{j}} \frac{1}{\alpha + \mathrm{j}} \delta_{x_{(1,\tilde{\mathrm{j}})}} + \frac{\alpha}{\alpha + \mathrm{j}} G_0. \tag{2.5}$$

This is a mixture between reinforcing existing samples and requesting new sample from G_0. Here G_0 can further be integrated out by reusing the CRP stated in equation (2.3) again.

The CRF process provides a simplification of the generative process of HDP. When generating samples from it, there is no need to simulate any of the distributions as long as we keep track of all the samples being created along the way. However, even with CRF, HDP still may not be the ideal model certain tasks. In the example of figure 2.2, if the intermediate distribution G_0 is marginalized out, the distribution of the child G_1 given the base H is no longer a Dirichlet process. Therefore, if only samples from certain nodes are of interest, we still need to simulate all the samples associated with their ancestors. Lacking of a nice marginalization property make the inference with HDP model computationally difficult in this kind of questions.

This is a brief introduction to HDP and CRF with information tailored to our need in the dissertation. For more detailed and comprehensive description of this model, including the extension of stick breaking process, we will direct the readers to the work of Teh, Jordan, Beal, *et al.* [47].

2.4 Pitman-Yor process and hierarchical Pitman-Yor process (HPYP)

Pitman-Yor process Pitman and Yor [48] is an extension of Dirichlet processes, and is also a nonparametric prior that can be used in model distributions. Besides a concentration parameter α and the base probability measure H on the space $\mathcal{X}$, specifying a Pitman-Yor process requires an additional parameter, the discount parameter $d \in (0, 1)$. A realization from the Pitman-Yor is a discrete probability measure G on $\mathcal{X}$, and we write it as

$$G \,|\, \alpha, d, H \sim \text{Pitman-Yor}\,(\alpha, d, H)\,. \tag{2.6}$$

The same as DPs, the base distribution H serves as the center of the Pitman-Yor process prior, with $\mathbb{E}[G] = H$. The concentration and discount parameter control the shape and spread of this prior around H, where $\alpha \geq -d$. When the discount $d = 0$, this process degenerates to a Dirichlet process [26].

As with the DPs, we can also associating a Chinese restaurant process (CRP) with the Pitman-Yor process. The generating rule of the $(\mathrm{j}+1)$-st observation $x_{\mathrm{j}+1}$ conditioning on previous observations is

$$x_{\mathrm{j}+1} \,|\, \{x_1, \ldots, x_{\mathrm{j}}\},\ \alpha,\ d,\ H \sim \sum_{k=1}^{K} \frac{n_k - d}{\alpha + \mathrm{j}} \delta_{y_k} + \frac{\alpha + K \cdot d}{\alpha + \mathrm{j}} H, \tag{2.7}$$

where $K = K(\mathrm{j})$ denotes the total number of clusters (tables) formed by the first j observations, and n_k and y_k are the cluster size and the first observation of cluster k, respectively.

Similar to DPs, Pitman-Yor processes can also be used in modeling distributions with a hierarchical dependency structure, which results in the hierarchical Pitman-Yor process (HPYP). The reason why this is particularly interesting to us is that unlike the DPs, Pitman-Yor processes have a convenient marginalization property when the concentration parameter is zero. Specifically, consider the hierarchical model in figure 2.2:

$$\begin{aligned} G_0 \,|\, H &\sim \text{Pitman-Yor}\,(0, d, H) \\ G_1 \,|\, G_0 &\sim \text{Pitman-Yor}\,(0, d, G_0)\,. \end{aligned} \tag{2.8}$$

It turns out that marginalizing out the intermediate distribution G_0, the distribution G_1 continues to follow a Pitman-Yor process, albeit with different parameters [49]–[51]:

$$G_1 \,|\, H \sim \text{Pitman-Yor}\left(0, d^2, H\right) \tag{2.9}$$

This marginalization property can result in significant savings in computation when not all intermediate distributions are of interest. With the concentration being zero, the discount parameter itself controls the spread of the distribution around the base measure. To be specific, a large discount parameter (close to one) results amplifies the similarity between a realizations of the process G and the base measure H.

3. REPULSIVE CLUSTERING WITH MATÉRN POINT PROCESSES

As statistics and machine learning methods find wide application in real world problems, practitioners are increasingly seeking to balance statistical fidelity and predictive accuracy with interpretability, parsimony and fairness. Popular instances of these trade-offs include introducing smoothness, sparsity or low-dimensional structure into statistical models. In this work, we focus on interpretability and diversity, through the use of *repulsive priors* in mixture modeling applications. Such models are useful in both density modeling and clustering applications, with goals for the latter typically including data exploration, visualization, and summarization (see section 2.1 for a brief review).

As stated in chapter 1, the flexibility of mixture models often comes at the cost of interpretability and parsimony. One approach towards addressing this problem is to control the number of clusters through an appropriate prior. However, trying to induce interpretability in this indirect fashion can make model specification quite challenging, especially in nonparametric applications where the number of clusters depends on the dataset size. Further, this approach is still sensitive to any misspecification of the form of the individual components. Another approach is to use more flexible (e.g. nonparametric) densities for each component of a mixture model [52], though once again this raises problems with model specification, identifiability and computation. A more modern approach is to directly address the problem of overlapping clusters, enforcing diversity through *repulsive priors*. Here, rather than being sampled independently from the base measure, cluster locations are jointly sampled from a prior distribution that penalizes realizations where clusters are situated too close to each other. Such priors typically draw from the point process literature, examples including Gibbs point processes [53] and determinantal point processes [54], [55]. Mixture models built on such priors have been shown to provide simpler, clearer and more interpretable results, often without too much loss of predictive performance [56]–[58]. Nevertheless, they present computational challenges, since the the repulsive models often involve normalization constants that are intractable to evaluate. Our work replaces the Gibbs point process with the Matérn type-III process, though one can use other underdispersed point processes. In [58], the au-

thors use a determinantal point processes (DPP) [54], [55], [59]. While mathematically and computationally elegant, DPPs are not as intuitive and mechanistic as Gibbs-type models, or our thinning mechanism. In our experiments, we compare with the models of Xie and Xu [60] and Bianchini, Guglielmi, Quintana, *et al.* [58].

Work on repulsive mixture models dates back to at least Dasgupta [61]. While that work did not propose a new model for repulsion, it demonstrated the importance of separated components for learning mixture models. An early Bayesian mixture model with repulsion was proposed in [56]. Here, repulsion was induced through a Gibbs point process mechanism: specifically, the prior probability of any configuration of cluster locations was proportional to the product of individual cluster probabilities multiplied by a term that penalizes nearby components. The authors there considered two types of penalties, one corresponding to a product of penalty terms for each pair of components, and one depending on the minimum separation between components, before deriving an MCMC sampler, and proving posterior consistency. Xie and Xu [60] and Quinlan, Page, and Quintana [62] generalized this model slightly, and also derived posterior rates of convergence. Fúquene, Steel, and Rossell [63] considered a similar approach to Petralia, Rao, and Dunson [56], though they framed their work in the more general setting of *non-local priors*. Here, given a collection of nested models, parameter configurations in a more complex model that result in an identical density to some configuration in a simpler model are given zero probability. All these works however face computational challenges: the Gibbs interaction term results in intractable normalization constants. This is especially severe when trying to infer parameters of the repulsive penalty, or switch between models with different numbers of components.

Finally, another line of work takes a post-processing approach, deliberately using over-fitted mixtures with a large number of components, and then discarding unoccupied clusters [64], [65], and merging nearby clusters together [66]. Unlike model-based approaches like ours, these are a bit ad hoc, making it difficult to coherently calibrate uncertainty, especially in more complicated hierarchical models.

In this work, we propose a new class of repulsive priors based on the Matérn type-III point process. Matérn point processes are a class of repulsive point processes first studied in Matérn [31], [32]. More recently, Rao, Adams, and Dunson [30] developed a simple and

efficient Markov chain Monte Carlo (MCMC) sampling algorithm for a generalized Matérn type-III process (see section 3.1). In this work, we bring this process to the setting of mixture models, using them as a repulsive prior over the number of clusters and their locations. Treating the Matérn realization as a latent, rather than a fully observed point process, raises computational challenges that the algorithm from Rao, Adams, and Dunson [30] does not handle. We develop an efficient MCMC sampler for our model and demonstrate the practicality and flexibility of our proposed repulsive mixture model on a variety of datasets. Our proposed algorithm is also useful in Matérn point process applications with missing observations, as well as for mixture models without repulsion, as an alternative to often hard-to-tune reversible jump MCMC methods [67] to sample the unknown number of components.

3.1 Introduction to Matérn repulsive point processes

The Poisson process [33] is a *completely random* point process, where events in disjoint sets are independent of each other. To incorporate repulsion between events, Matérn [31], [32] introduced three spatial point process models that build on the Poisson process. The three models, called the Matérn hardcore point process of type I, II and III, only allow point process realizations with pairs of events separated by at least some fixed distance η, where η is a parameter of the model. The three models are constructed by applying different dependent thinning schemes on a *primary* homogeneous Poisson point process. Despite being theoretically more challenging than the other two processes, the type-III process has the most natural thinning mechanism, and supports higher densities of points. [30] showed how this can easily be generalized to include probabilistic thinning and spatial inhomogeneity. Furthermore, Rao, Adams, and Dunson [30] showed that posterior inference for a completely observed type-III process can be carried out in a relatively straightforward manner. These advantages make the generalized Matérn type-III process superior to the other two as a prior for mixture models. For simplicity, we will refer to this process as the Matérn process in the rest of this work.

Formally, the Matérn process is a finite point process defined on a space Θ, parameterized by a thinning kernel $\mathcal{K}_\eta : \Theta \times \Theta \to [0, 1]$ and a nonnegative intensity function $\lambda_\Theta : \Theta \to$

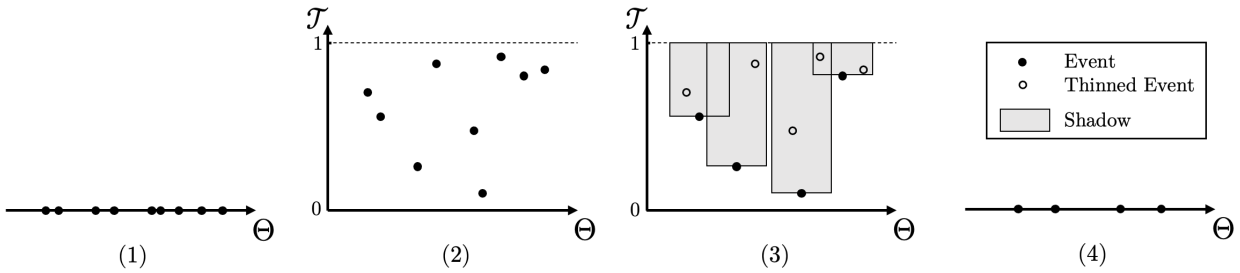

Figure 3.1. The generative process of a one-dimensional Matérn process with a hardcore thinning kernel $\mathcal{K}_\eta$. **(1)** A primary Poisson point process F_Θ with intensity $\lambda_\Theta(\theta)$ is simulated on Θ, **(2)** Events in F_Θ are assigned random birth times uniformly from $\mathcal{T} = [0, 1]$, **(3)** Events in the shadow (i.e. within horizontal distance η) of earlier surviving events are thinned, **(4)** Surviving events are projected onto Θ to form the Matérn realization G_Θ.

$[0, \infty)$. We will find it convenient to decompose the function $\lambda_\Theta(\theta)$ as $\lambda_\Theta(\theta) = \bar{\lambda} \cdot p_\Theta(\theta)$, for a finite normalizing constant $\bar{\lambda} > 0$ and some probability density $p_\Theta(\theta)$ on Θ. Simulating this process proceeds in four steps: (1) Simulate the primary process $F_\Theta = \left\{\theta_1, \ldots, \theta_{|F_\Theta|}\right\}$ from a Poisson process with intensity $\lambda_\Theta(\cdot)$ on Θ. (2) Assign each event θ_j in F_Θ an independent random *birth-time* uniformly on the interval $\mathcal{T} = [0, 1]$. (3) Sequentially visit events in the primary process according to their birth-times (from the oldest to the youngest) and attempt to thin (delete) them. Specifically, at step j, the jth oldest event (θ, t) is thinned by each surviving older primary event $(\theta, t), t < t$ with probability $\mathcal{K}_\eta(\theta, \theta)$. (4) Write G_Θ and $\tilde{G}_\Theta$ for the elements of F_Θ that survive and are thinned from the previous step, respectively. The set G_Θ forms the Matérn process realization.

For a hardcore Matérn process (figure 3.1), the thinning kernel satisfies $\mathcal{K}_\eta(\theta, \theta_\mathrm{j}) = \mathbb{1}_{\|\theta - \theta_\mathrm{j}\| < \eta}$, where η is the thinning radius, so that thinning is deterministic: newer events within distance η of a previously survived event are thinned with probability 1. Other approaches are probabilistic thinning [30], where $\mathcal{K}_\eta(\theta, \theta_\mathrm{j}) = \eta_1 \mathbb{1}_{\|\theta - \theta_\mathrm{j}\| < \eta_2}$ (with $\eta_1 \in [0, 1]$), or the smoother squared-exponential thinning, where $\mathcal{K}_\eta(\theta, \theta_\mathrm{j}) = \exp(-\frac{\|\theta - \theta_\mathrm{j}\|^2}{2\eta})$. Huber and Wolpert [68] propose soft-core thinning, where each event θ_j has its own thinning radius η_j drawn from some distribution, and $\mathcal{K}_\eta(\theta, \theta_\mathrm{j}) = \mathbb{1}_{\|\theta - \theta_\mathrm{j}\| < \eta_\mathrm{j}}$.

Observe that since each event θ_j has an independently and uniformly distributed birth-time t_j associated with it, the set of pairs $\left\{(\theta_1, t_1), \ldots, (\theta_{|F_\Theta|}, t_{|F_\Theta|})\right\}$ is itself distributed as

a Poisson process on $\Theta \times \mathcal{T}$, with intensity $\lambda(\theta, t) = \lambda_\Theta(\theta)\mathbb{1}_{[0,1]}(t)$. We write this extended primary process as F. Consistent with our use of F_Θ to represent the set of locations of each point in F, we will use $F_\mathcal{T}$ to represent the set of birth-times. Similarly, we will use G for the extended Matérn events, and $G_\mathcal{T}$ for the associated birth-times (and $\tilde{G}$ and $\tilde{G}_\mathcal{T}$ for their thinned counterparts).

Following Rao, Adams, and Dunson [30], we will specify the thinning process through a *shadow function* $\mathcal{H}_\eta : \Theta \times \mathcal{T} \to [0, 1]$ parameterized by a possibly vector-valued η. This gives the probability that an event $(\theta^*, t^*) \in \Theta \times \mathcal{T}$ is thinned by a collection of events G as

$$\mathcal{H}_\eta\left((\theta^*, t^*)\,; G\right) = 1 - \prod_{g \in G}\left[1 - \mathcal{H}_\eta\left((\theta^*, t^*)\,; g\right)\right], \tag{3.1}$$

where for a single event $g = (\theta, t)$, $\mathcal{H}_\eta\left((\theta^*, t^*)\,; (\theta, t)\right) = \mathbb{1}_{[t,1]}(t^*)\mathcal{K}_\eta\left(\theta^*, \theta\right)$. Note that the $\mathbb{1}_{[t,1]}(t^*)$ formalizes the fact that an event (θ^*, t^*) can only be thinned by earlier events. We will write $\text{MatérnThin}_\mathcal{K}\left(F, \eta\right)$ for the sequential thinning process that assigns elements of F to one of G or $\tilde{G}$ according to thinning kernel $\mathcal{K}_\eta$ (algorithm 1), and $\text{Proj}_\mathcal{A}(\cdot)$ for the operator that projects elements of a set on to some subspace $\mathcal{A}$. The generative process of $G_\Theta \sim \text{MatérnProcess}_\mathcal{K}\left(\lambda, \eta\right)$ can be written as

$$\begin{aligned} F \,|\, \lambda &\sim \text{PoissonProcess}\left(\lambda(\cdot, \cdot)\right), \\ G, \tilde{G} \,\Big|\, F, \mathcal{K}_\eta &\sim \text{MatérnThin}_\mathcal{K}\left(F, \eta\right), \quad G_\Theta = \text{Proj}_\Theta(G). \end{aligned}$$

With a Matérn model of point pattern data, one seeks to infer the intensity function $\lambda_\Theta(\theta)$ and the thinning parameters η from a realization G_Θ. Observe that for the Matérn type-III process, an event can only be thinned by a surviving event, so that the probability of thinning at any location depends only on the set G. Rao, Adams, and Dunson [30] showed that in fact, conditioned on G, the events in $\tilde{G}$ are distributed as an inhomogeneous Poisson process with intensity $\lambda_\Theta(\theta)\mathcal{H}_\eta\left((\theta, t)\,; G\right)$. This allowed them to develop an efficient Gibbs sampler when Matérn events G_Θ are fully observed. This proceeds by sequentially updating the thinned events $\tilde{G}$, the Matérn birth times $G_\mathcal{T}$, the Poisson intensity λ_Θ and thinning kernel parameter η, each conditioned on the rest. The fact that the thinned events $\tilde{G}$ can

be jointly sampled avoids the need for any birth-death steps in the MCMC algorithm, both simplifying the algorithm and improving its efficiency. We adapt this sampler for our MCMC algorithm, where the fact that the Matérn events are hidden or partially observed will create new challenges. In the next section, we first describe our model that uses the Matérn process to impose repulsion between clusters of a finite mixture model.

Algorithm 1: Details of the function $\text{MatérnThin}_{\mathcal{K}}(F, \eta)$

Function $\text{MatérnThin}_{\mathcal{K}}(F, \eta)$**:**
- **Input :** Extended primary Poisson process F and thinning kernel $\mathcal{K}_\eta$
- **Output:** Extended Matérn events G and thinned events $\tilde{G}$

Write $\overrightarrow{F} = \left(f_1, \ldots, f_{|F|}\right)$ for F sorted in ascending order of birth times (so that $\text{Proj}_{\mathcal{T}}(f_{\text{j}}) < \text{Proj}_{\mathcal{T}}(f_{\text{j}})$ if $\text{j} < \text{j}$).
for $\text{j} \leftarrow 1$ **to** $|F|$ **do**
  Set $(\theta, t) \leftarrow (\text{Proj}_{\Theta}(f_{\text{j}}), \text{Proj}_{\mathcal{T}}(f_{\text{j}}))$
  Draw $u \sim \text{Unif}[0, 1]$
  if $u < \mathcal{H}_\eta((\theta, t); G)$ **then** `// Assign` f_{j} `to` G `w.p.` $\mathcal{H}_\eta((\theta, t); G)$
    $G \leftarrow G \cup f_{\text{j}}$
  else
    $\tilde{G} \leftarrow \tilde{G} \cup f_{\text{j}}$

return G, $\tilde{G}$

3.2 Matérn repulsive mixture model (MRMM)

We start with a primary Poisson process F that includes mixture weights, defining it on an extended space $\Theta \times \mathcal{W} \times \mathcal{T}$ with $\mathcal{T} = [0, 1], \mathcal{W} = [0, \infty)$. Write its intensity function as

$$\lambda(\theta, w, t) = \bar{\lambda} \cdot p_\Theta(\theta) \cdot p_{\mathcal{W}}(w) \cdot \mathbb{1}_{[0,1]}(t). \tag{3.2}$$

We set $p_{\mathcal{W}}(w) = \text{Gamma}(w; \alpha, 1)$, while $p_\Theta(\theta)$ is a problem-specific prior over cluster parameters. Unlike the Matérn process, we model F as a Poisson process conditioned to have at least one event. Given F, we will produce a Matérn realization $G = \left\{(\theta_1, w_1, t_1), \ldots, (\theta_{|G|}, w_{|G|}, t_{|G|})\right\}$ by applying the function $\text{MatérnThin}_{\mathcal{K}}(F, \eta)$ for some kernel $\mathcal{K}$ on Θ with parameter η. Each element $(\theta, w, t) \in G$ will form a component of a mixture model, with θ and w representing

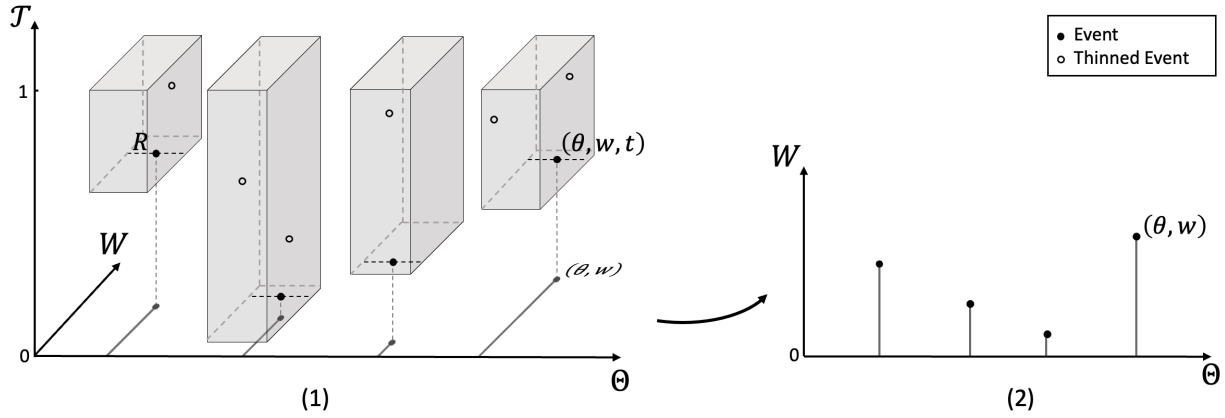

Figure 3.2. Illustration of the Matérn prior for mixture models. **(1)** Primary Poisson events $F = \left\{(\theta_1, w_1, t_1), \ldots, (\theta_{|F|}, w_{|F|}, t_{|F|})\right\}$ thinned by a hardcore thinning kernel with thinning radius R. The surviving events are projected to the parameter space of the mixture model $\Theta \times \mathcal{W}$. **(2)** The resulting mixture model, consisting of a collection of mixture component parameters $\theta \in \Theta$ and their corresponding unnormalized mixture weights $w \in \mathcal{W}$.

the parameter and unnormalized weight of that component; see also figure 3.2. Our model thus serves as a prior over both the number of components in a mixture model, as well as the component weights and locations. Since events in F can only be thinned by surviving events, our modified Matérn prior on F ensures the mixture model has at least one component.

For a set A, write $\sum A$ for the sum of its elements. Consistent with the notation of G_Θ and $G_\mathcal{T}$, we write $G_\mathcal{W}$ for $\mathrm{Proj}_\mathcal{W}(G)$. Then, given G, the observed data $\boldsymbol{X} = \{x_\mathrm{i}, \mathrm{i} = 1, \ldots, n\}$ is modeled as follows:

$$x_\mathrm{i} \,|\, G \sim \sum_{(\theta,w,t)\in G} \frac{w}{\sum G_\mathcal{W}} p_\mathcal{X}(\cdot\,;\theta), \qquad \mathrm{i} = 1, \ldots, n. \tag{3.3}$$

Here, $p_\mathcal{X}(\cdot\,;\theta)$ represents some family of probability densities parameterized by $\theta \in \Theta$. As an example, if the observations lie on a Euclidean space, $p_\mathcal{X}(\cdot\,;\theta)$ could be a normal distribution, with θ representing the location and variance of a cluster in a Gaussian mixture model. In this case, the density $p_\Theta(\theta)$ might be a Normal-Inverse-Wishart distribution.

Note that when the w's are independent Gamma$(\alpha, 1)$ variables, the vector of normalized weights $\left(w_1 \big/ \sum G_\mathcal{W}, \ldots, w_{|G|} \big/ \sum G_\mathcal{W}\right)$ follows a symmetric Dirichlet distribution with con-

centration parameter α [42]. If the thinning kernel $\mathcal{K}_\eta$ equals 0, our model then reduces to a standard mixture model, with i.i.d. cluster parameters, Dirichlet-distributed cluster weights, and a conditional Poisson distribution on the number of clusters. Different settings of $\mathcal{K}_\eta$ (hardcore, probabilistic or squared-exponential thinning) allow different kinds of repulsion between the cluster parameters. Observe that repulsion is only between the cluster parameters θ (and not the cluster weights w). Further, in many settings we allow $\mathcal{K}_\eta$ to only depend on a subset of the components of θ. For instance, writing $\theta = (\theta^\mu, \theta^\sigma)$ where θ^μ is the cluster location and θ^σ is the cluster variance, a common requirement is to enforce repulsion only between the cluster locations, but not their variances. This can easily be achieved by setting $\mathcal{K}_\eta$ to depend only on θ^μ.

All that is left to complete a Bayesian model is to specify hyperpriors on the hyperparameters $\bar{\lambda}$ and η, as well as any hyperparameters of the density $p_\Theta(\theta)$. The last is problem specific, and is no different from models without repulsion. A natural prior for $\bar{\lambda}$ is the Gamma distribution, while the choice of the hyperprior on η will depend on the type of thinning kernel. In general, we recommend at least a mildly informative prior on the thinning parameter, as otherwise, the posterior can settle on a model without any repulsion. For the hardcore process, where η is the thinning radius, or for the squared-exponential thinning kernel, where η is the lengthscale parameter, we can use a Gamma hyperprior. For probabilistic thinning, where $\eta = (R, p)$, we can use a Beta prior on the thinning probability p, and a Gamma prior on the thinning radius R. We include further discussion of the choice of hyperpriors in chapter 4.

Write $\boldsymbol{z} = (z_1, \ldots, z_n)$ for the collection of latent cluster assignments of the data in equation (3.3), with $z_\mathrm{i} \in \{1, \ldots, |G|\}$. Following notation in section 3.1, with hyperpriors omitted for simplicity, the generative process of MRMM is

$$\begin{aligned} F \,|\, \lambda &\sim \text{PoissonProcess}\left(\lambda(\cdot)\right) \Big| |F| > 0, \\ G, \tilde{G} \,\Big|\, F, \mathcal{K}_\eta &\sim \text{MatérnThin}_\mathcal{K}\left(F, \eta\right), \\ z_\mathrm{i} \,|\, G &\sim \text{Multinomial}\left(\frac{w_1}{\sum G_\mathcal{W}}, \ldots, \frac{w_{|G|}}{\sum G_\mathcal{W}}\right), \\ x_\mathrm{i} \,|\, z_\mathrm{i}, G &\sim p_\mathcal{X}\left(\cdot\,; \theta_{z_\mathrm{i}}\right), \qquad \mathrm{i} = 1, \ldots, n. \end{aligned} \tag{3.4}$$

Note that for convenience, in the last line above we have imposed an arbitrary ordering on the elements of G, and thus the cluster identities, though the cluster indicators z_{i} really take values in some categorical space. The proposition below gives the joint density of all variables, and will be useful for deriving our posterior sampling algorithm.

Proposition 3.2.1. *Write $\mathscr{P}_\lambda$ for the measure of a rate-$\lambda(\cdot)$ Poisson process on $\Theta \times \mathcal{W} \times \mathcal{T}$. Then the tuple $\boldsymbol{X}$, G, $\tilde{G}$ has joint density with respect to $\mathscr{P}_\lambda \times dx^n$ given by*

$$p\left(\boldsymbol{X}, G, \tilde{G} \,\middle|\, \lambda, \eta\right) = \frac{\mathbb{1}\left(|G \cup \tilde{G}| > 0\right)}{1 - \mathrm{e}^{\int_{\Theta\times\mathcal{W}\times\mathcal{T}} -\lambda(\theta,w,t)\,\mathrm{d}\theta\,\mathrm{d}w\,\mathrm{d}t}} \prod_{g\in G}\left[1 - \mathcal{H}_\eta\left(g\,;G\right)\right] \prod_{\tilde{g}\in\tilde{G}} \mathcal{H}_\eta\left(\tilde{g}\,;G\right) \prod_{\mathrm{i}=1}^{n} \sum_{(\theta,w,t)\in G} \frac{w}{\sum G_{\mathcal{W}}} p_{\mathcal{X}}\left(x_{\mathrm{i}}\,;\theta\right). \tag{3.5}$$

3.3 Posterior inference for MRMM

Given a dataset $\boldsymbol{X} = \{x_1, \ldots, x_n\}$ from MRMM, we are interested in the posterior distribution $p\left(G, \boldsymbol{z}, \bar{\lambda}, \eta \,\middle|\, \boldsymbol{X}\right)$, summarizing information about the cluster weights and locations (through G), and the cluster assignments (through $\boldsymbol{z}$). We construct a Markov chain Monte Carlo (MCMC) sampler to simulate from it. Our sampler is an auxiliary variable Gibbs sampler, that for computational reasons, also imputes the thinned events $\tilde{G}$. The sampler proceeds by sequentially updating the latent variables $\bar{\lambda}, \eta, G, \tilde{G}$ and $\boldsymbol{z}$ according to their conditional posterior distributions. Among these, the most challenging steps are updating G and $\tilde{G}$: both of these are variable-dimension objects, where not just the values but also the cardinality of the sets must be sampled. Given Matérn events G, sampling $\tilde{G}$ resembles the sampling problem from Rao, Adams, and Dunson [30], where the Matérn realization was completely observed. However, our modified prior on F (where $|F|$ must be greater than 0) requires some care, and we provide a different and cleaner derivation of this update step using Campbell's theorem [33]. Updating G given the rest is more challenging, and we further augment our MCMC state space with an independent Poisson process $\tilde{F}$, and then update the triplet $(G, \tilde{G}, \tilde{F})$ to $(G^*, \tilde{G}^*, \tilde{F}^*)$ using a 'relabeling' process that keeps the union unchanged. This approach is simpler than reversible-jump or birth-death approaches, with

the augmented Poisson intensity trading-off mixing and computation, and forming the only new parameter. Below, we present full details of the Gibbs steps.

1) Updating thinned events $\tilde{G}$

Given G, the thinned events $\tilde{G}$ are independent of the observations: $p(\tilde{G} \mid \bar{\lambda}, \eta, G, \boldsymbol{z}, \boldsymbol{X}) = p(\tilde{G} \mid \bar{\lambda}, \eta, G)$. Furthermore, events in $\tilde{G}$ can only be thinned by events in G, suggesting that conditioned on $G, \bar{\lambda}, \eta$, the events within $\tilde{G}$ do not interact with each other, and form a Poisson process. The result below formalizes this:

Proposition 3.3.1. *Given all other variables, the conditional distribution of the thinned events $\tilde{G}$ is a Poisson process with intensity $\lambda(\cdot)\mathcal{H}_\eta(\cdot\,; G)$.*

This result resembles that of Rao, Adams, and Dunson [30], though our derivation in section 3.4 exploits proposition 3.2.1 and works with densities with respect to the rate-λ Poisson measure, and is simpler and cleaner. Simulating such a Poisson process is straightforward: simulate a Poisson process with intensity $\lambda(\cdot)$ on the whole space $\Theta \times \mathcal{W} \times \mathcal{T}$, and then keep each event $\tilde{g}$ in it with probability $\mathcal{H}_\eta(\tilde{g}\,; G)$ [69]. This makes jointly updating the entire set $\tilde{G}$ easy and efficient, without any tuning parameters.

2) Updating the Matérn events G

This step is significantly more challenging, since unlike the thinned events, the Matérn events interact with each other, and with the clustering structure of the data. Consequently, we cannot simply discard G and sample a new realization. Instead, we produce a dependent update of G, through a Markov kernel that targets this conditional distribution.

We first discard the cluster assignments $\boldsymbol{z}$; note these can easily be resampled (see step 3 below). A naive approach is then to make a pass through the elements of $G \cup \tilde{G}$, reassigning each to either G or $\tilde{G}$ based on the appropriate conditional. This forms a standard sequence of Gibbs updates, and does not involve any reversible jump or stochastic process simulation. At the end of this pass, we have an updated pair $(G^*, \tilde{G}^*)$, with G^* possibly having different number of elements from G. While this keeps the union $G \cup \tilde{G}$ unchanged, our ability to efficiently update $\tilde{G}$ might suggest fast mixing.

In our experiments however, we observed poor mixing, especially with hardcore thinning. The latter setting forbids elements of G^* from lying within each others' shadow, and also requires $\tilde{G}^*$ to lie in the shadow of G^*, making it hard to switch an event from the Matérn set to thinned set, or vice versa. To address this, we begin this step by augmenting our MCMC state-space with an independent rate-$\gamma\lambda(\cdot)$ Poisson process $\tilde{F} \subset \Theta \times \mathcal{W} \times \mathcal{T}$:

$$\tilde{F} \,\Big|\, \gamma, \lambda \sim \text{PoissonProcess}\left(\gamma\lambda(\cdot)\right). \tag{3.6}$$

We call $\gamma > 0$ the augmentation factor, which forms a parameter of our MCMC algorithm. Since $\tilde{F}$ is simulated independently of all other variables, the joint distribution of $(G, \tilde{G}, \tilde{F})$ conditioned on all other variables has the conditional distribution of $(G, \tilde{G})$ as its marginal. Having simulated $\tilde{F}$, we cycle through the elements of $G \cup \tilde{G} \cup \tilde{F}$, sequentially relabeling each event as "survived", "thinned" or "augmented" to produce a new triplet $G^* \cup \tilde{G}^* \cup \tilde{F}^*$. This relabeling is carried to preserve the joint conditional of $G^*, \tilde{G}^*, \tilde{F}^*$, so that after discarding $\tilde{F}^*$, we have updated $(G, \tilde{G})$ while maintaining their conditional distribution.

The augmented Poisson process $\tilde{F}$ more easily allows events to be introduced into, and removed from G, especially in the hardcore setting. Each relabeling step is straightforward, and requires computing a 3-component probability. For each $\mathrm{e} \in G \cup \tilde{G} \cup \tilde{F}$, write $G^{\setminus \mathrm{e}}, \tilde{G}^{\setminus \mathrm{e}}$ and $\tilde{F}^{\setminus \mathrm{e}}$ for the sets resulting from removing e (only one of these will change). Write $S^{\setminus \mathrm{e}}$ for the sum of the unnormalized mixture weights after removing e: $S^{\setminus \mathrm{e}} = \sum \mathrm{Proj}_{\mathcal{W}}(G^{\setminus \mathrm{e}})$. For an observation $x_{\mathrm{i}} \in \boldsymbol{X}$ and event $g = (\theta, w, t) \in G$, write $l_{\mathrm{i}}^g = w p_{\mathcal{X}}\left(x_{\mathrm{i}}\,; \theta\right)$. Write $L_{\mathrm{i}}^{\setminus \mathrm{e}} = \sum_{g \in G^{\setminus \mathrm{e}}} l_{\mathrm{i}}^g$, this is the unnormalized likelihood of observation i with event e taken out,

and with its cluster assignment marginalized out. Then, following proposition 3.2.1, and with $\mathrm{e}_{\mathcal{W}} = \mathrm{Proj}_{\mathcal{W}}(\mathrm{e})$, the probabilities of "survived", "thinned" or "augmented" are

$$
\begin{aligned}
P(\mathrm{e} \in G|-) &\propto \prod_{\mathrm{i}=1}^{n} \frac{l_{\mathrm{i}}^{\mathrm{e}} + L_{\mathrm{i}}^{\backslash \mathrm{e}}}{S^{\backslash \mathrm{e}} + \mathrm{e}_{\mathcal{W}}} \prod_{g \in G^{\backslash \mathrm{e}} \cup \{\mathrm{e}\}} \left[1 - \mathcal{H}_{\eta}(g\,; G^{\backslash \mathrm{e}} \cup \{\mathrm{e}\})\right] \prod_{\tilde{g} \in \tilde{G}} \mathcal{H}_{\eta}(\tilde{g}\,; G^{\backslash \mathrm{e}} \cup \{\mathrm{e}\}), \\
P(\mathrm{e} \in \tilde{G}|-) &\propto \prod_{\mathrm{i}=1}^{n} \frac{L_{\mathrm{i}}^{\backslash \mathrm{e}}}{S^{\backslash \mathrm{e}}} \prod_{g \in G^{\backslash \mathrm{e}}} \left[1 - \mathcal{H}_{\eta}(g\,; G^{\backslash \mathrm{e}})\right] \prod_{\tilde{g} \in \tilde{G}^{\backslash \mathrm{e}} \cup \{\mathrm{e}\}} \mathcal{H}_{\eta}(\tilde{g}\,; G^{\backslash \mathrm{e}}), \\
P(\mathrm{e} \in \tilde{F}|-) &\propto \gamma \prod_{\mathrm{i}=1}^{n} \frac{L_{\mathrm{i}}^{\backslash \mathrm{e}}}{S^{\backslash \mathrm{e}}} \prod_{g \in G^{\backslash \mathrm{e}}} \left[1 - \mathcal{H}_{\eta}(g\,; G^{\backslash \mathrm{e}})\right] \prod_{\tilde{g} \in \tilde{G}^{\backslash \mathrm{e}}} \mathcal{H}_{\eta}(\tilde{g}\,; G^{\backslash \mathrm{e}}).
\end{aligned}
\tag{3.7}
$$

Having cycled through all elements of $G \cup \tilde{G} \cup \tilde{F}$, we have a new partition $(G^*, \tilde{G}^*, \tilde{F}^*)$, after which the augmented Poisson events $\tilde{F}^*$ are discarded; see also algorithm 2 and figure 3.3. The augmented factor γ in this procedure governs the cardinality of augmented events $\tilde{F}$. A larger γ results in faster mixing, but higher computational cost. Our experiments suggests that a moderate augmentation factor (somewhere between 5 to 10) adequately balances mixing and computation.

3) Updating cluster assignments z and cluster weights $G_{\mathcal{W}}$

Given $\boldsymbol{X}$ and mixture parameters G_{Θ} and $G_{\mathcal{W}}$, we can easily resample the cluster assignments $\boldsymbol{z}$ that were discarded at the start of the previous step. This is no different from standard mixture models; for observation i: $p(z_{\mathrm{i}} = g|-) \propto l_{\mathrm{i}}^{g},\ \forall g \in G$. Clusters assignments for all observations are conditionally independent, so that these assignments can be carried out in parallel.

Given cluster assignments $\boldsymbol{z}$ and the number of mixture components $|G|$, the mixture weights $G_{\mathcal{W}} = \{w_{\mathrm{j}}, \mathrm{j} = 1, \ldots, |G|\}$ are independent of the other variables. A priori, the w_{j}'s are independent Gamma$(\alpha, 1)$ random variables, or equivalently, are obtained by multiplying a sample from a Dirichlet$(\alpha, \ldots, \alpha)$ distribution (the normalized weights) with an independent sample from a Gamma$(|G|\alpha, 1)$ distribution (the sum of the weights) [42]. We work with the latter representation, and seek to simulate from the posterior distribution of the normalized weights and the sum of the weights. It is easy to see that these continue to be

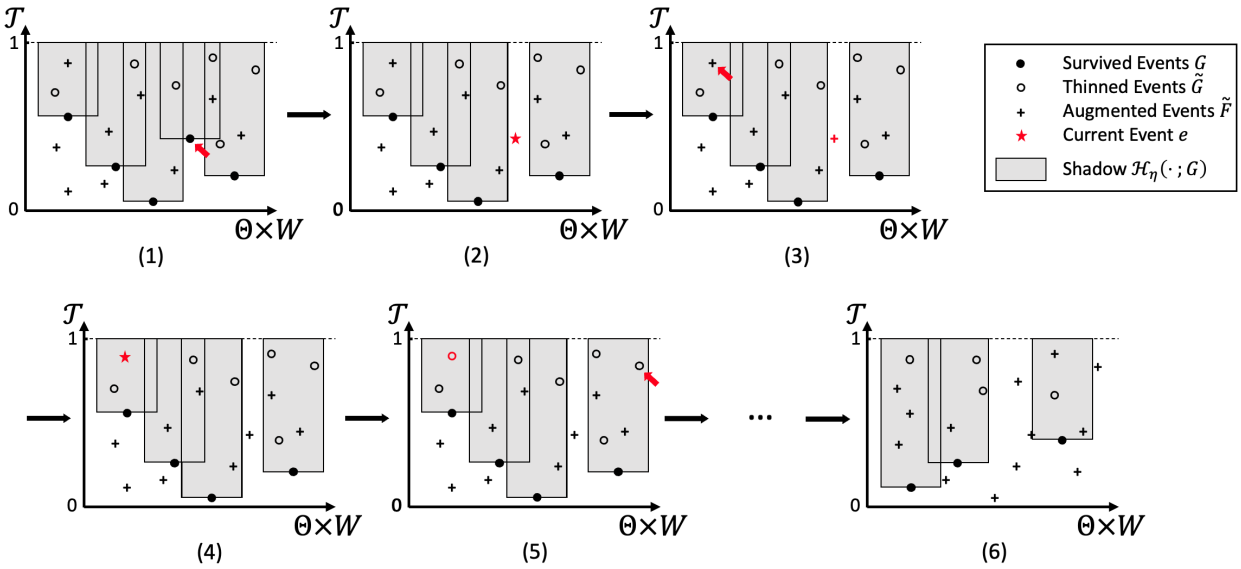

Figure 3.3. Illustration of the relabeling step. **(1)** Before relabeling, the state of the surviving events G, thinned events $\tilde{G}$ and auxiliary events $\tilde{F}$, and the shadow cast by G, $\mathcal{H}_\eta(\cdot\,; G)$. **(2-3)** The first event (After random shuffling of all events in $G \cup \tilde{G} \cup \tilde{F}$) is relabeled as "auxiliary". The event is first removed from its original set G (and the shadow is affected accordingly) in **(2)**. Then, in **(3)**, it is relabeled as "auxiliary" according to the posterior conditional probabilities in equation (3.7). Notice that with the hardcore thinning kernel, it is impossible for the event to be relabeled to "thinned", as it is not under the shadow of a previously surviving event. **(4-5)** The second event is relabeled as "thinned". Similarly, the event is removed from the collection of augmented events F in **(4)** and then relabeled as "thinned" in **(5)**. Notice that it is under the shadow of a surviving event, and hence, with the hardcore thinning kernel, it can only be labeled as "thinned" or "auxiliary". **(6)** The final state for G, $\tilde{G}$, $\tilde{F}$, after all events are relabeled.

independent under the posterior. The sum of the weights plays no role in the likelihood, and continues to follow a Gamma($|G|\alpha, 1$) distribution, while the Dirichlet-multinomial conjugacy implies that the normalized weights follow a Dirichlet($\alpha + n_1, \ldots, \alpha + n_{|G|}$), with n_j the number of observations in cluster j.

4) Updating cluster locations G_Θ and Matérn birth-times $G_\mathcal{T}$

Updating the cluster locations G_Θ and birth-times $G_\mathcal{T}$ is not strictly necessary given the relabeling step, nevertheless, we find doing it improves mixing. With the number of Matérn events G and thinned Matérn events G determined, updating these is straightforward, if a little tedious. Unlike standard mixture models, because of repulsion, clusters locations are not conditionally independent. Write θ_j for the location of j-th cluster, and write $\boldsymbol{X}_\mathrm{j}$ for the observations assigned to this cluster. Then, the conditional distribution of θ_j is

$$p\left(\theta_\mathrm{j} \,\middle|\, G_\Theta^{-\mathrm{j}}, G_\mathcal{T}, \tilde{G}, \lambda, \eta, \boldsymbol{X}_\mathrm{j}\right) \propto p\left(G, \tilde{G} \,\middle|\, \lambda, \eta\right) p_\Theta\left(\theta_\mathrm{j}\right) \prod_{x \in \boldsymbol{X}_\mathrm{j}} p_\mathcal{X}\left(x\,; \theta_\mathrm{j}\right). \tag{3.8}$$

The term $p(G, \tilde{G} \,|\, \lambda, \eta)$ accounts for how changing the jth event's location changes the shadow, and therefore the probability of the current Matérn and thinned events. The other two terms are the prior and likelihood of θ_j under a mixture model without repulsion. A simple way to simulate from this is with a Metropolis-Hastings step, and when the prior p_θ is conjugate to the likelihood $p(x \,|\, \theta)$, a natural choice for the proposal distribution is the posterior distribution if there were no repulsion: $q_\mathrm{j}\left(\theta_\mathrm{j}\right) \propto p_\Theta\left(\theta_\mathrm{j}\right) \prod_{x \in \boldsymbol{X}_\mathrm{j}} p_\mathcal{X}\left(x\,; \theta_\mathrm{j}\right)$.

Like the cluster locations, the birth-times $G_\mathcal{T}$ of the Matérn events can also be updated one at a time. Given the cluster locations, $G_\mathcal{T}$ is independent of the observations or their cluster assignments, and one only needs to consider their impact on the shadow (proposition 3.2.1). Specifically, if t_j is the birth time of the j-the event, then

$$p\left(t_\mathrm{j} \,|\, -\right) \propto p\Big(G, \tilde{G} \,\Big|\, \lambda, \eta\Big) \propto \prod_{g \in G} \Big[1 - \mathcal{H}_\eta\left(g\,; G\right)\Big] \prod_{\tilde{g} \in \tilde{G}} \mathcal{H}_\eta\left(\tilde{g}\,; G\right).$$

Since $t_{\mathrm{j}} \in [0,1]$, simulating from this is straightforward, though it is possible to simplify this further. When the thinning kernel is symmetric, this first product term does not depend on t_{j}, and can be dropped. Next, the birth-times of the thinned events $\tilde{g}_{\mathrm{j}} = (\tilde{\theta}_{\mathrm{j}}, \tilde{w}_{\mathrm{j}}, \tilde{t}_{\mathrm{j}}) \in \tilde{G}$ can be used to partition the interval $\mathcal{T} = [0,1]$ into segments $[\tilde{t}_{\mathrm{j}}, \tilde{t}_{\mathrm{j}+1})$, $\mathrm{j} = 1, \ldots, |\tilde{G}| - 1$. If the thinning probability is a function only of separation in space (as is the case with all kernels we have considered), then the probability of t_{j} within each segment is constant, depending only on the number of thinned events born before and after the interval $[\tilde{t}_{\mathrm{j}}, \tilde{t}_{\mathrm{j}+1})$. Specifically, for any time t, define $\tilde{G}^{\leq t}$ as the subset of events in $\tilde{G}$ that were born before or at t, and define $\tilde{G}^{>t}$ similarly. Then

$$p\left(t_{\mathrm{j}} \in [\tilde{t}_{\mathrm{j}}, \tilde{t}_{\mathrm{j}+1}) \,\middle|\, -\right) \propto \prod_{\tilde{g} \in \tilde{G}^{\leq t_{\mathrm{j}}}} \mathcal{H}_\eta\left(\tilde{g}\,; G^{-\mathrm{j}}\right) \prod_{\tilde{g} \in \tilde{G}^{>t_{\mathrm{j}}}} \mathcal{H}_\eta\left(\tilde{g}\,; G\right). \tag{3.9}$$

Having picked a segment, the exact value of t_{j} is drawn uniformly within the segment.

5) Updating hyperparameters

Hyperparameters include the primary Poisson process intensity, and those in the thinning kernel. The mean intensity $\bar{\lambda}$ controls the cardinality of the primary process F, and it is easy to show that with a Gamma(a, b) prior, and with the constraint $|F| > 0$, the conditional posterior is $p\left(\bar{\lambda} \,\middle|\, -\right) \propto \frac{1}{1-\mathrm{e}^{-\bar{\lambda}}} \mathrm{Gamma}(\bar{\lambda}\,; a + |F|, b + 1)$.

Next, write ν for any parameters of the normalized Poisson intensity $p_\Theta(\theta \,|\, \nu) = \lambda_\Theta(\theta)/\bar{\lambda}$. For a prior $p_\nu(\nu)$, the conditional distribution simplifies as $p(\nu \,|\, -) \propto p_\nu(\nu) \prod_{\theta \in F_\Theta} p_\Theta(\theta \,|\, \nu)$. Finally, writing p_η for the prior for the thinning parameter η, the posterior is $p\left(\eta \,|\, -\right) \propto p_\eta(\eta) \prod_{g \in G} \left[1 - \mathcal{H}_\eta\left(g\,; G\right)\right] \prod_{\tilde{g} \in \tilde{G}} \mathcal{H}_\eta\left(\tilde{g}\,; G\right)$. All three distributions above can be updated using any standard MCMC kernel.

Algorithm 2: The relabeling step to update Matérn events G

Function `Relabel(`λ, γ, G, $\tilde{G}$, $\boldsymbol{X}$`):`

Input : Primary Poisson intensity λ, augmentation factor γ, current state of the surviving events G and the thinned events $\tilde{G}$, the data $\boldsymbol{X}$.

Output: Updated Matérn events G and thinned events $\tilde{G}$.

Sample augmented $\tilde{F} \sim \text{PoissonProcess}\left(\gamma\lambda(\cdot)\right)$

Impute non-locational parameters of $\tilde{G}$ from the prior (if presents in the model)

Obtain shuffled indices $J =$ `RandomShuffle(`$\{1, \ldots, |G \cup \tilde{G} \cup \tilde{F}|\}$`)`

Compute likelihood related objects: $n \times |J|$ matrix $L = \left(w_{\mathrm{j}} p_{\mathcal{X}}\left(x_{\mathrm{i}}\,; \theta_{\mathrm{j}}\right) : \mathrm{i}, \mathrm{j}\right)$ and n-dim vector $\boldsymbol{l} = \left(\sum_{g \in G} l_1^g, \ldots, \sum_{g \in G} l_n^g\right)$

Compute the normalizing constant $S = \sum G_{\mathcal{W}}$

foreach j *in* J **do**

  if *event* j *in* G **then**

    if $|G| = 1$ **then** `// G contains only event j`

      next

    else

      $S \leftarrow S - w_{\mathrm{j}}$

      $\boldsymbol{l} \leftarrow \boldsymbol{l} - L_{\cdot \mathrm{j}}$

  Remove event j from its original event set

  Assign event j to G, $\tilde{G}$ or $\tilde{F}$ with probability $P(\mathrm{e} \in G|-)$, $P(\mathrm{e} \in \tilde{G}|-)$ and $P(\mathrm{e} \in \tilde{F}|-)$ in equation (3.7), respectively,

return G, $\tilde{G}$

Algorithm 3: Bayesian inference of MRMM

Input : Data $\boldsymbol{X} = \{x_1, \ldots, x_n\}$, number of MCMC iterations M, model of cluster components $p_{\mathcal{X}}(\cdot\,; \theta)$, augmentation factor γ, prior on cluster locations p_θ, shape parameter of the Gamma prior on weights α, shape and rate parameter of the Gamma prior on mean intensity (a, b), and prior on thinning kernel parameter p_η.

Output: Posterior samples of mean intensity $\bar{\lambda}$, thinning parameter η, Matérn events G_Θ, $G_\mathcal{T}$, $G_\mathcal{W}$, thinned events $\tilde{G}$, and cluster assignments $\boldsymbol{z}$.

Initialize $\bar{\lambda} \sim \text{Gamma}(a, b)$, $\eta \sim p_\eta$

Initialize $G, \tilde{G} \sim \text{MatérnProcess}_\mathcal{K}(\lambda, \mathcal{K}_\eta)$

Initialize $\boldsymbol{z}$ from $\boldsymbol{z} \,|\, \boldsymbol{X}, G$: $z_\text{i} \sim \text{Multinomial}\left(w_\text{j} \cdot p_\mathcal{X}(X_\text{i}\,; \theta_\text{j}),\ \text{j} = 1, \ldots, |G|\right)$

for $m \leftarrow 1$ **to** M **do**

  Update $\bar{\lambda}$ according to $\frac{1}{1-\text{e}^{-\bar{\lambda}}}\text{Gamma}(a + |F|, b + 1)$ using Metropolis-Hastings

  Update η according to $p(\eta \,|\, G, \tilde{G})$

  Update $\tilde{G}$: (Poisson thinning) simulate from $\text{PoissonProcess}(\lambda)$ and discard event $\tilde{g}$ with probability $1 - \mathcal{H}_\eta(\tilde{g}\,; G)$

  Update $G_\mathcal{T}$ one at a time according to equation (3.9)

  Update $G_\mathcal{W} \leftarrow S \cdot \overline{G_\mathcal{W}}$ where $S \sim \text{Gamma}(|G|\alpha, 1)$ and $\overline{G_\mathcal{W}} \sim \text{Dirichlet}\left(\alpha + n_1, \ldots, \alpha + n_{|G|}\right)$ $(n_\text{j} = \sum_{\text{i}=1}^{n} \mathbb{1}(z_\text{i} = \text{j}))$

  Update G_Θ one at a time according to equation (3.8) using Metropolis-Hastings

  $G, \tilde{G} \leftarrow$ `Relabel(`λ, γ, G, $\tilde{G}$, $\boldsymbol{X}$`)`

  Update $\boldsymbol{z}$ one at a time: $z_\text{i} \sim \text{Multinomial}\left(w_\text{j} \cdot p_\mathcal{X}(X_\text{i}\,; \theta_\text{j}), \text{j} = 1, \ldots, |G|\right)$

return *Posterior MCMC samples of* $\bar{\lambda}$, η, G, $\tilde{G}$ *and* $\boldsymbol{z}$

3.4 Proofs

Proposition (3.2.1)**.** *Write $\mathscr{P}_\lambda$ for the measure of a Poisson process on $\Theta \times \mathcal{W} \times \mathcal{T}$ with intensity $\lambda(\theta, w, t)$. Then the tuple $\boldsymbol{X}$, G, $\tilde{G}$ has joint density with respect to $\mathscr{P}_\lambda \times \mathrm{d}x^n$ given by*

$$p\left(\boldsymbol{X}, G, \tilde{G} \,\middle|\, \lambda, \eta\right) = \left(\frac{\mathbb{1}\left(|G \cup \tilde{G}| > 0\right)}{1 - \mathrm{e}^{\int_{\Theta\times\mathcal{W}\times\mathcal{T}} -\lambda(\theta,w,t)\ \mathrm{d}\theta\,\mathrm{d}w\,\mathrm{d}t}}\right) \left(\prod_{g\in G}\left[1 - \mathcal{H}_\eta\left(g\,;G\right)\right] \prod_{\tilde{g}\in\tilde{G}} \mathcal{H}_\eta\left(\tilde{g}\,;G\right)\right) \left(\prod_{\mathrm{i}=1}^{n} \sum_{(\theta,w,t)\in G} \frac{w}{\sum G_{\mathcal{W}}} p_{\mathcal{X}}\left(x_{\mathrm{i}}\,;\theta\right)\right).$$

Proof. First note that the set $F = G \cup \tilde{G}$ follows a Poisson process with rate $\lambda(\theta, w, t)$, conditioned to have at least 1 event. The probability that such a Poisson process produces 1 or more events is $1 - \exp(-\int \lambda(\theta, w, t) d\theta dw dt)$. It follows that conditioning on this event, F has density with respect to $\mathscr{P}_\lambda$ given by the ratio in the first parentheses. Each element f of F is assigned to either G or $\tilde{G}$, with probability $1 - \mathcal{H}_\eta\left(f\,;G\right)$ or $\mathcal{H}_\eta\left(f\,;G\right)$ respectively. This gives the terms in the second parentheses. Finally, the ith observation is assigned to cluster $(\theta, w, t) \in G$ with probability $w/G_{\mathcal{W}}$, with its value having density $p_{\mathcal{X}}\left(x_{\mathrm{i}}\,;\theta\right)$ with respect to dx. Marginalizing over cluster assignments, and considering all n observations, we get the final terms. The result then follows easily from Lemma 3.4.1. □

To prove proposition 3.3.1, we start with the following useful (and not new) result:

Lemma 3.4.1. *Consider two Poisson processes on some space $\mathcal{Y}$, with intensities $\lambda(y)$ and $\mu(y)$. Then the former has density with respect to the latter given by*

$$\frac{d\mathscr{P}_\lambda}{d\mathscr{P}_\mu}(M) := p_\mu(M|\lambda) = \mathrm{e}^{\int_{\mathcal{Y}} \mu(y) - \lambda(y)\,\mathrm{d}y} \prod_{m\in M} \frac{\lambda(m)}{\mu(m)} \tag{3.10}$$

Proof. Consider a function $h : \mathcal{Y} \to \mathrm{Re}$. For a point process M on $\mathcal{Y}$, we overload notation, and define the linear functional $h(M) = \sum_{m\in M} h(m)$. Write $\mathbb{E}_{\mathscr{M}}[h(M)]$ for the expectation of $h(M)$ when M is distributed as a point process with measure $\mathscr{M}$. Recall that $\mathscr{P}_\lambda$ corresponds

to a rate-$\lambda(\cdot)$ Poisson process on $\mathcal{Y}$, and $\mathscr{P}_\mu$, to a rate-$\mu(\cdot)$ Poisson process.We first note that from Campbell's theorem [33], for a rate-$\mu(\cdot)$ Poisson process, we have

$$\mathbb{E}_{\mathscr{P}_\mu}[\exp(h(M))] = \mathbb{E}_{\mathscr{P}_\mu}\left[\exp\left(\sum_{m\in M} h(m)\right)\right] = \exp\left(\int (\mathrm{e}^{h(y)}-1)\mu(y)\,\mathrm{d}y\right). \tag{3.11}$$

Now write $\mathscr{M}_\mu^\lambda$ for the probability measure of a point process with density $p_\mu(M|\lambda)$ with respect to a rate-$\mu(\cdot)$ Poisson process. Then

$$\begin{aligned}
\mathbb{E}_{\mathscr{M}_\mu^\lambda}[\exp(h(M))] &= \mathbb{E}_{\mathscr{P}_\mu}\left[p_\mu(M|\lambda)\exp(h(M))\right] \\
&= \mathbb{E}_{\mathscr{P}_\mu}\left[\mathrm{e}^{\int_{\mathcal{Y}}(\mu(y)-\lambda(y))\,\mathrm{d}y}\left(\prod_{m\in M}\frac{\lambda(m)}{\mu(m)}\right)\exp(h(M))\right] \\
&= \mathrm{e}^{\int_{\mathcal{Y}}(\mu(y)-\lambda(y))\,\mathrm{d}y}\,\mathbb{E}_{\mathscr{P}_\mu}\left[\exp\sum_{m\in M}\left(h(m)+\log\lambda(m)-\log\mu(m)\right)\right] \\
&= \exp\left(\int_{\mathcal{Y}}(\mathrm{e}^{h(y)}-1)\lambda(y)\,\mathrm{d}y\right) \quad \text{(from equation (3.11))} \\
&= \mathbb{E}_{\mathscr{P}_\lambda}[\exp(h(M))]. \qquad (3.12)
\end{aligned}$$

This confirms that $\mathscr{M}_\mu^\lambda$ equals $\mathscr{P}_\lambda$ a.e., proving our result. □

Proposition (3.3.1)**.** *Given all other variables, the conditional distribution of the thinned events $\tilde{G}$ is a Poisson process with intensity $\lambda(\cdot)\mathcal{H}_\eta(\cdot\,;G)$.*

Proof. With respect to a rate-$\lambda(\cdot)$ Poisson process,

$$\begin{aligned}
p(\tilde{G}|-) &\propto p\left(G,\tilde{G},\boldsymbol{X}\,\middle|\,\lambda,\eta\right) \\
&= \left(\frac{\mathbb{1}(|G\cup\tilde{G}|>0)}{1-\mathrm{e}^{\int_{\Theta\times\mathcal{W}\times\mathcal{T}}-\lambda(\theta,w,t)\,\mathrm{d}\theta\,\mathrm{d}w\,\mathrm{d}t}}\right)\prod_{g\in G}\left[1-\mathcal{H}_\eta(g\,;G)\right]\prod_{\tilde{g}\in\tilde{G}}\mathcal{H}_\eta(\tilde{g}\,;G) \\
&\propto \prod_{\tilde{g}\in\tilde{G}}\mathcal{H}_\eta(\tilde{g}\,;G).
\end{aligned}$$

In the last equation, we dropped all terms that do not depend on $\tilde{G}$, and used the fact that since $|G|>0$, $\mathbb{1}(|G\cup\tilde{G}|>0)$. The result now follows from Lemma 3.4.1. □

4. EMPIRICAL RESULTS OF MRMM

In this chapter we evaluate different settings of our MRMM model and MCMC algorithm described in chapter 3, and compare with two other repulsive models: the DPP-based method of Bianchini, Guglielmi, Quintana, *et al.* [58] and the repulsive Gaussian mixture model of Xie and Xu [60]. We implemented our method as a Python3 package `mrmm`. An `R` implementation of the method of Bianchini, Guglielmi, Quintana, *et al.* [58] was acquired directly from the authors, while a `MATLAB` implementation of the method of Xie and Xu [60] was obtained from their supplementary material.

For all experiments, we placed a Gamma$(1, 1)$ prior on the unnormalized mixture weights w_{j} in our model, resulting in a flat Dirichlet prior on the mixing proportions. Unless otherwise specified, we placed a Gamma$(1, 0.1)$ prior on the mean intensity of the primary Poisson process, $\bar{\lambda}$. We considered three Matérn thinning kernels, the hardcore, probabilistic and squared-exponential kernel (see table 4.1 for details). Recall that with zero repulsion, MRMM reduces to an independent mixture model with a prior on the number of components. As stated earlier, it is important to have a relatively informative hyperprior on the parameters of the repulsive kernel, otherwise the model can revert to no repulsion. For most experiments, the thinning strength (thinning radius in the hardcore and probabilistic kernels, and lengthscale in the squared-exponential kernel) had a Gamma$(4, 2)$ prior, which had mean 2 and variance 1.

We evaluated model and sampler performance along three dimensions: computational efficiency, goodness-of-fit and parsimony. For computational efficiency, we first computed the effective sample size (ESS) of a number of posterior statistics (for simplicity, we reported only one of them, the number of clusters). ESS estimates the number of uncorrelated samples that a sequence of dependent MCMC samples corresponds to, and to compute this, we used the `effectiveSize` function from the `R` package `coda` [70]. Dividing this by the total run CPU runtime of the MCMC sampler, we get the ESS per second (ESS/s), an estimate of the cost of producing one independent sample. We use this as our measure of sampler efficiency. To evaluate the goodness-of-fit and predictive accuracy, we reported the predictive likelihood $\ln p\left(\boldsymbol{X}_{\text{test}} \mid \boldsymbol{X}\right)$ of a held-out test dataset $\boldsymbol{X}_{test}$, as well as the log pseudo-marginal likelihood

Table 4.1. Thinning kernels used in experiments

Thinning Kernel	Thinning Parameter		Expression
Hardcore	$\eta = R$	Radius $R > 0$	$\mathcal{K}_R(\theta, \theta) = \mathbb{1}_{\|\theta - \theta\| < R}$
Probabilistic	$\eta = (R, p)$	Radius $R > 0$ Probability $p \in [0, 1]$	$\mathcal{K}_{(R,p)}(\theta, \theta) = p\mathbb{1}_{\|\theta - \theta\| < R}$
Squared-exponential	$\eta = l$	Lengthscale $l > 0$	$\mathcal{K}_l(\theta, \theta) = \exp\left\{-\frac{\|\theta - \theta_j\|^2}{2l}\right\}$

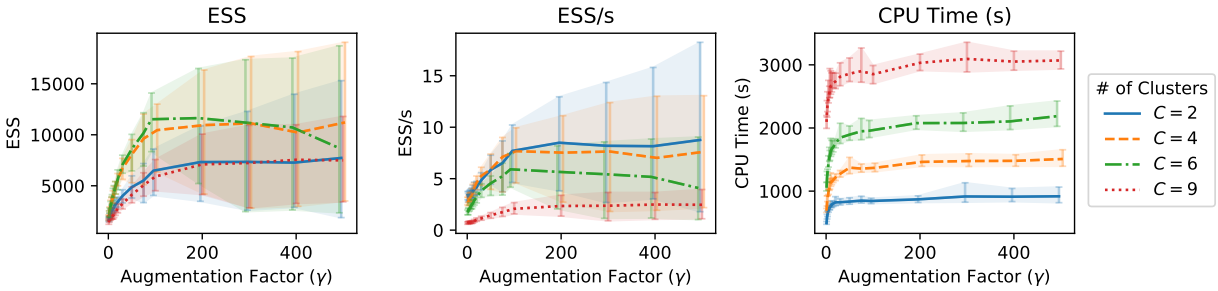

Figure 4.1. The impact of augmentation factor on **(left)** MCMC mixing (ESS out of 20,000 iterations), **(middle)** MCMC mixing rate (ESS/s) and **(right)** computational cost (CPU time). A tiny perturbation is added to γ's to ensure visibility.

LPML $= \sum_i \log p(x_i \mid \boldsymbol{X}^{-i})$ where $\boldsymbol{X}^{-i}$ denotes the dataset without the i-th observation [see 58]. To assess the parsimony and interpretability of inferred model, we reported the posterior mean and variance of the number of clusters ($\mathbb{E}[C \mid \boldsymbol{X}]$ and $\mathrm{Var}(C \mid \boldsymbol{X})$), as well as a central estimate of the posterior clustering structure (a 'median' posterior clustering). The latter was obtained by minimizing the posterior expectation of Binder's loss function under equal misclassification costs [58], [71]. We denote the number of clusters in this estimate as $\hat{C}_B$.

4.1 Synthetic studies

4.1.1 Study of augmentation factor

We focus here on MRMM with hardcore thinning, the most challenging setting for MCMC mixing. We applied MRMM to synthetic data generated from two-dimensional Gaussian mixture models, with minimum cluster separation of 4.0 and with varying number of clusters. The models to generate the datasets are illustrated in figure 4.2.

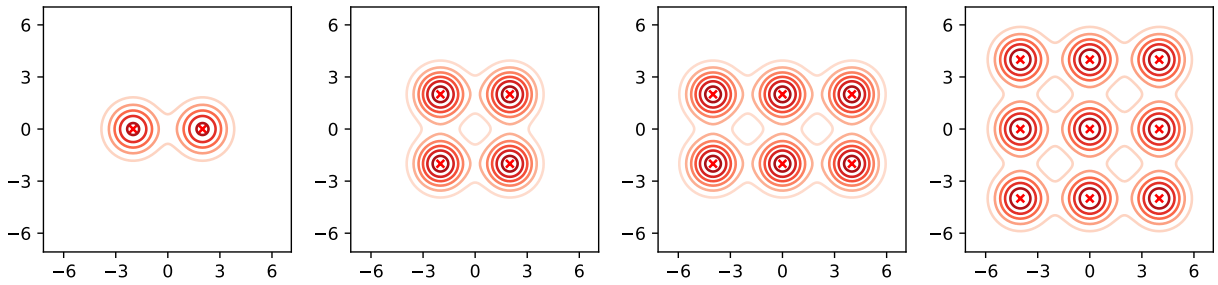

Figure 4.2. Mixtures of equally weighted Gaussian distributions for the study of augmentation factor γ in section 4.1.1. From left to right, number of clusters $C = 2, 4, 6, 9$, respectively. Each cluster is a standard bivariate Gaussian with covariance being the 2×2 identity matrix I_2. The minimum distances between cluster centers is 4.

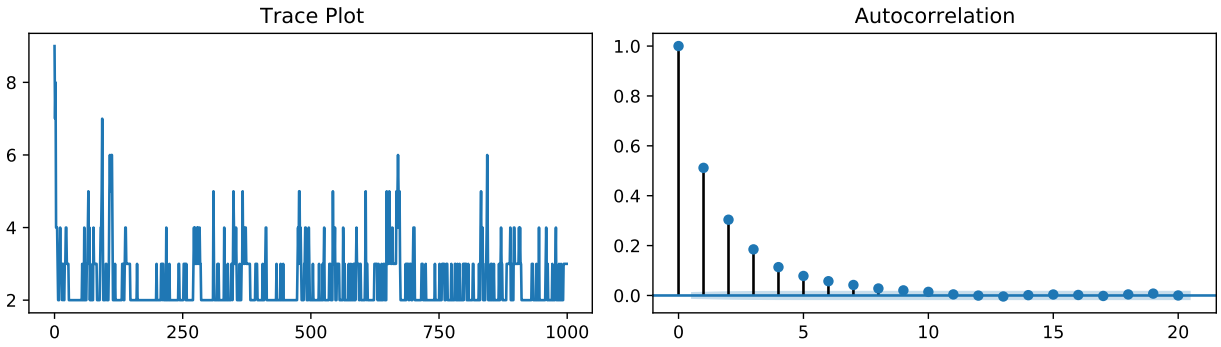

Figure 4.3. Visualization for assessing mixing of posterior number of clusters ($|G|$) in one run with augmentation factor $\gamma = 5$ on the dataset with two clusters as illustrated in figure 4.2. In this run, ESS = 5624; ESS/s = 6.97; CPU Time (s) = 806.80. **(Left)** The trace plot of the first 1,000 updates of $|G|$. **(Right)** The autocorrelation function of posterior samples of $|G|$.

For each model, we simulated 50 training datasets, each consisting of 20 observations per cluster. The number of clusters C thus quantifies both model complexity and dataset size. We modeled each dataset as a hardcore MRMM with the thinning radius fixed to 2. The covariance of each cluster was set to the 2×2 identity matrix I_2, and the normalized intensity $p_\Theta(\theta)$ was set to $N(\mathbf{0}, 10I_2)$. For each dataset, we ran our MCMC sampler for 20,000 iterations, with γ ranging from 1 to 500.

Figure 4.1 plots the raw ESS (left), ESS/s (center) and CPU run-time (right) against the augmentation factor γ, with each curve representing a different generative model. The right panel shows that, as expected, increasing γ results in an increase in CPU time, as the number of events in the augmentation Poisson process increases. At the same time, the leftmost panel shows that this added computational cost comes with the benefit of faster mixing, as more augmented Poisson events more easily allows events to be switched into and out of the Matérn events G. For small values for γ, this improvement is significant, before plateauing out as γ crosses 50. The middle panel shows that this improvement easily compensates for the added computational burden. We see similar results for other thinning kernels, but do not include them. In practice, based on these results, we recommend setting γ somewhere in the range of 5 to 10. In the rest of our experiments, we fix it to 5. Figure 4.3 visualizes assessments for the mixing of one run.

4.1.2 Study of thinning kernels and thinning strengths

Having established that our MCMC sampler mixes well, we now proceed to study the effect of different thinning kernels and thinning strengths on MRMM inferences. Table 4.1 lists all thinning kernels are corresponding parameters used in this study, specifically, for the probabilistic thinning kernel, the thinning probability $p = 0.95$.

We consider a series of two-dimensional Gaussian mixture models shown in figure 4.4. Each model consists of four equally weighted, unit-variance Gaussian components, located at $(-d/2, 3d/2)$, $(d/2, d)$, $(d, -d)$, $(-3d/2, -3d/2)$, where $d = 1, 2, 3, 4$ quantifies the separation level. A training dataset of size 200 and a test data with 100 observations were simulated independently for each model.

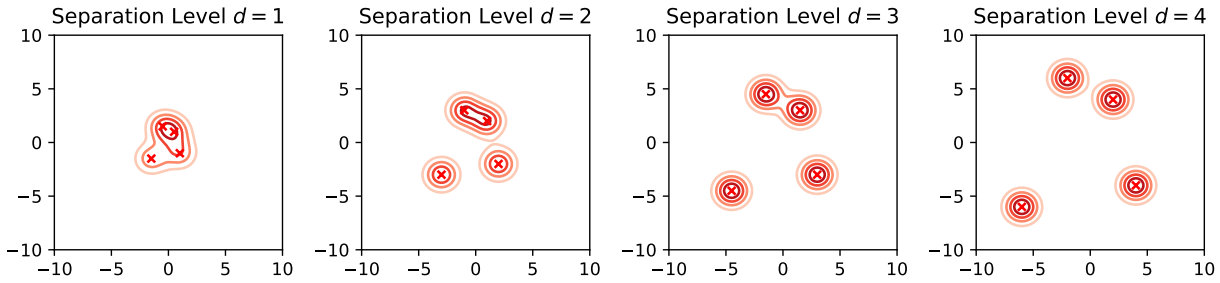

Figure 4.4. Synthetic study in section 4.1.2: The ground truth model M_0 with different separation levels.

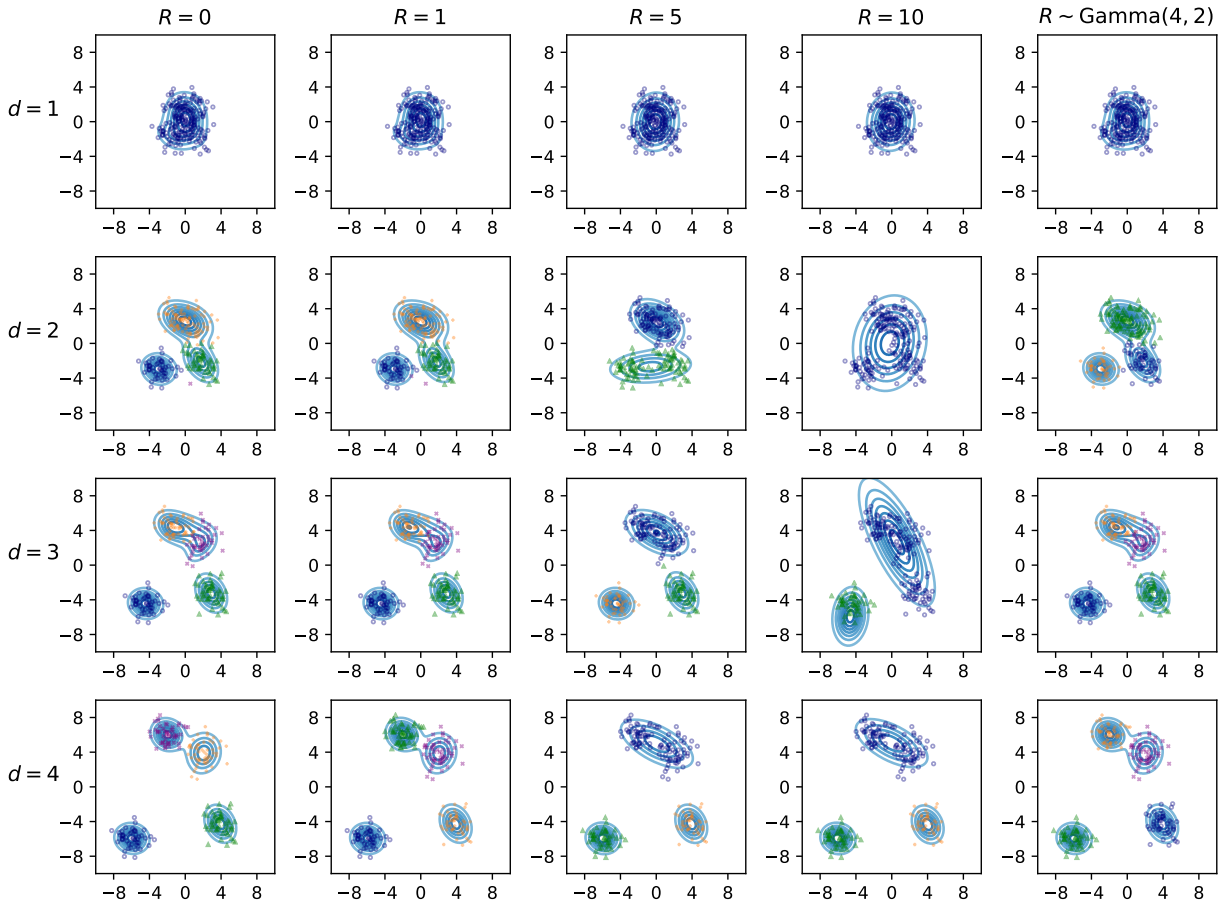

Figure 4.5. Contour and cluster assignments of the synthetic study in section 4.1.2 with hardcore MRMM.

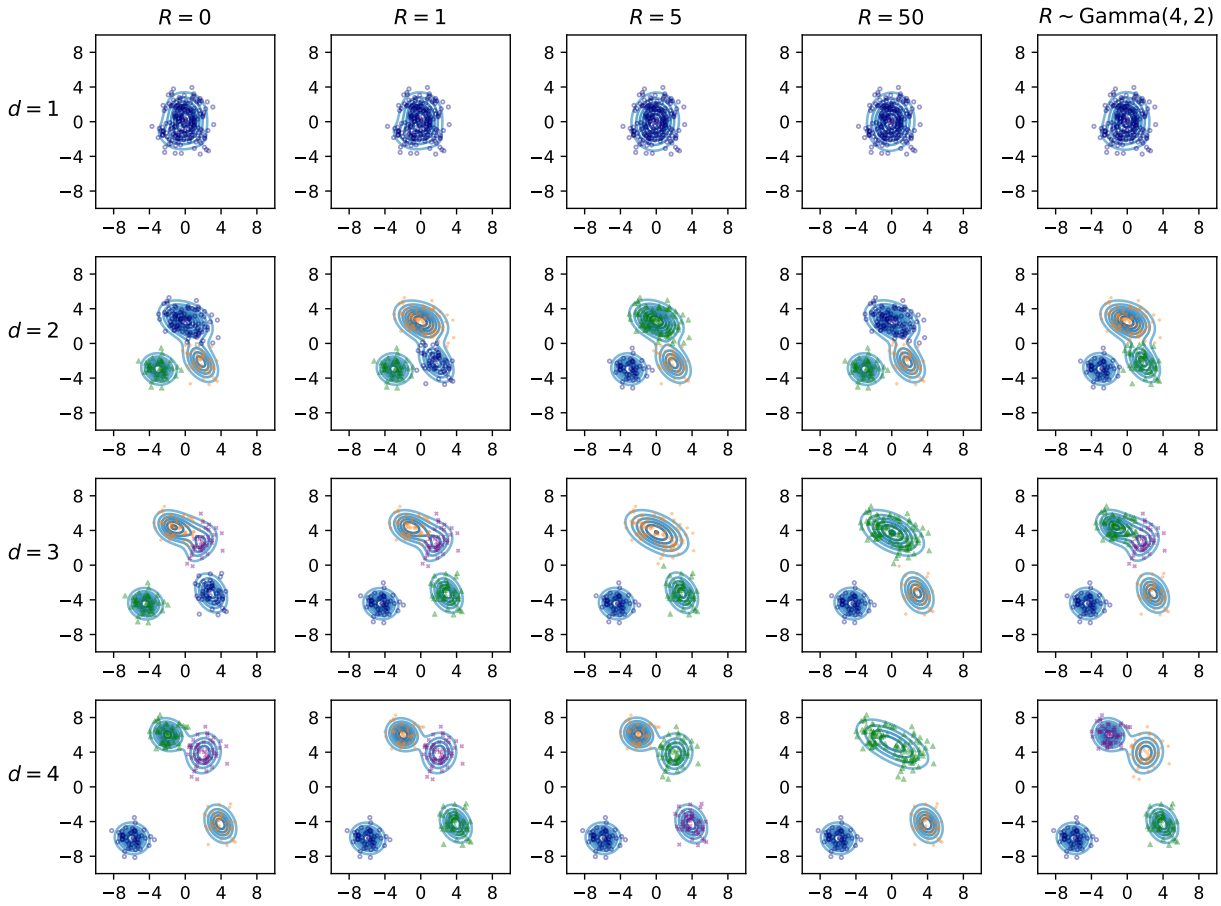

Figure 4.6. Contour and cluster assignments of the synthetic study in section 4.1.2 with probabilistic MRMM.

For MRMM, we set the prior $p_\Theta(\theta)$ to a Gaussian with mean zero and covariance $10I_2$. We placed an inverse-Wishart prior with 2 degrees of freedom and a scale matrix I_2 on the covariances. When learning the thinning strength (thinning radius R for both hardcore and probabilistic MRMM, or the lengthscale l for the squared-exponential MRMM), we placed a Gamma$(4, 2)$ prior with mean 2.0 and variance 1.0. All results were obtained from 2,000 iterations of MRMM after discarding the first 1,000 samples as burn-in.

Figure 4.5, 4.6 and 4.7 are the inferred posterior contours and the 'median' clustering results obtained with the three kernels. Heatmaps in figures 4.8 to 4.12 compare the parsimony and the goodness-of-fit of different thinning kernels with a variety of thinning strengths. As expected, increasing repulsion strength results in greater parsimony, with both the posterior

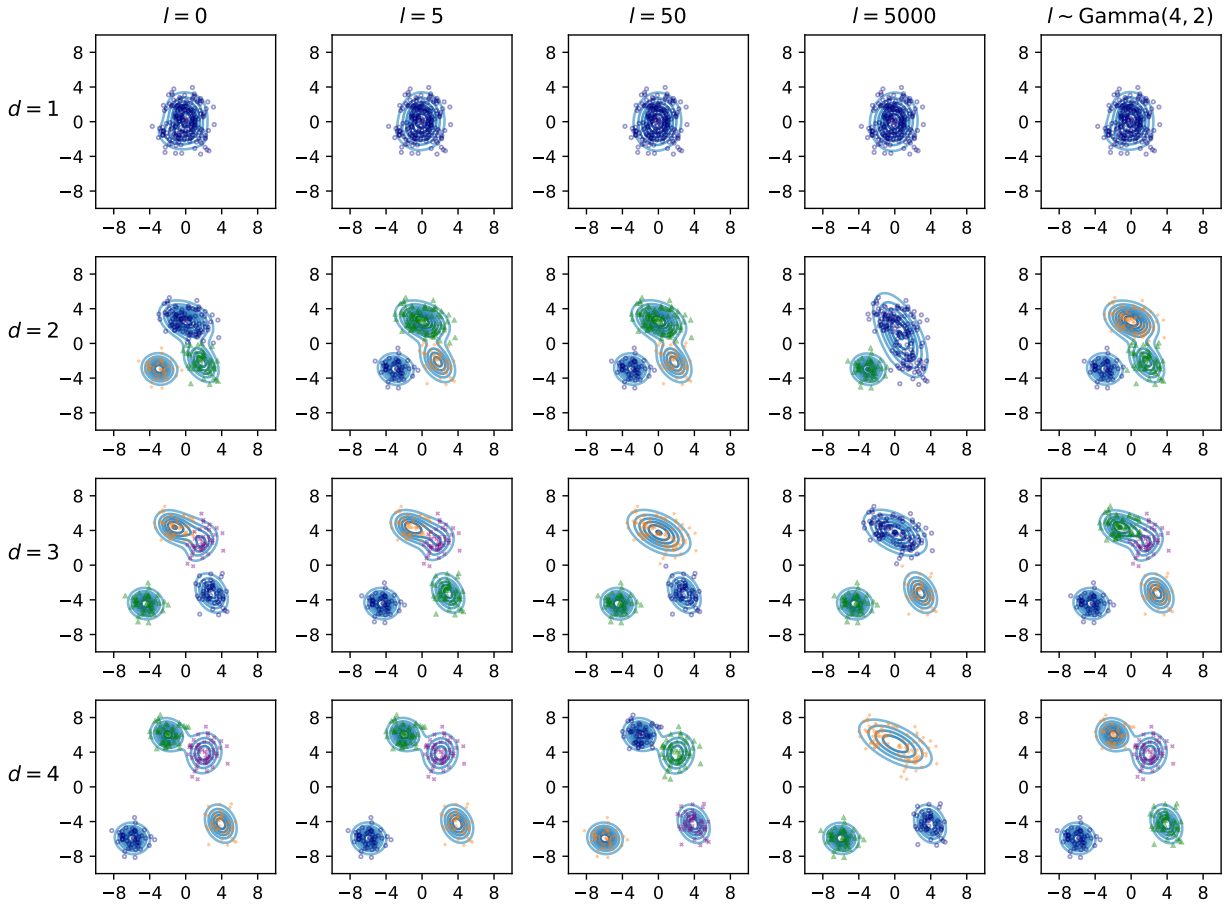

Figure 4.7. Contour and cluster assignments of the synthetic study in section 4.1.2 with squared-exponential MRMM.

mean and variance of the number of clusters dropping. Interestingly, moderate values of repulsion do not significantly harm the model fit. However, a strong repulsion strength does result in a drop in predictive power, especially for the hardcore MRMM.

4.2 Real Data Analysis

4.2.1 Chicago 2019 homicide data

We next consider a dataset of homicide recordings, collected in Chicago, Illinois in the year 2019 [1]. The data consists 501 entries, and we randomly split these into 416 (85%) training data points and 85 (15%) testing data points. Figure 4.14 shows the training data,

[1]↑obtained from https://data.cityofchicago.org/Public-Safety/Crimes-2019/w98m-zvie

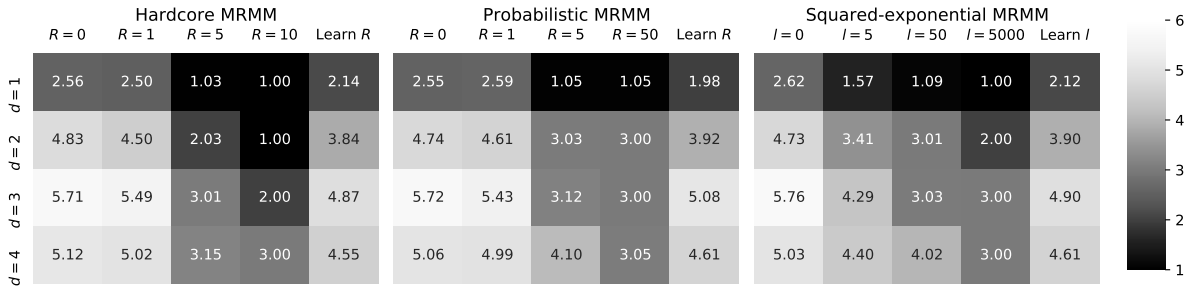

Figure 4.8. Synthetic study in section 4.1.2: Posterior mean of the number of clusters $\mathbb{E}[C \mid \boldsymbol{X}]$.

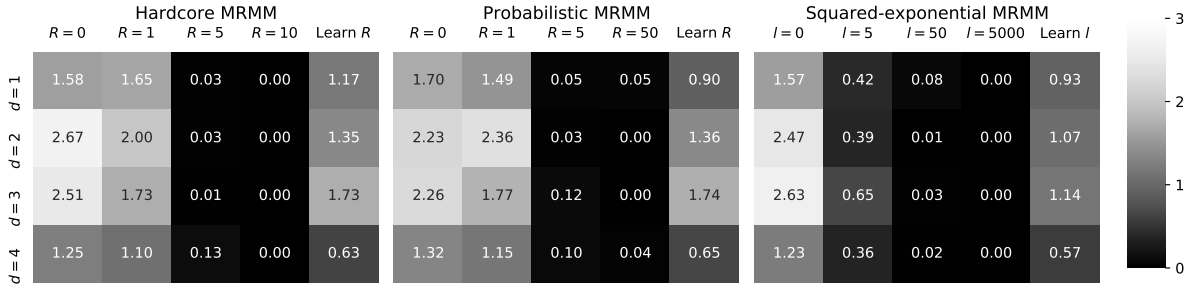

Figure 4.9. Synthetic study in section 4.1.2: Posterior variance of the number of clusters $\mathrm{Var}(C \mid \boldsymbol{X})$.

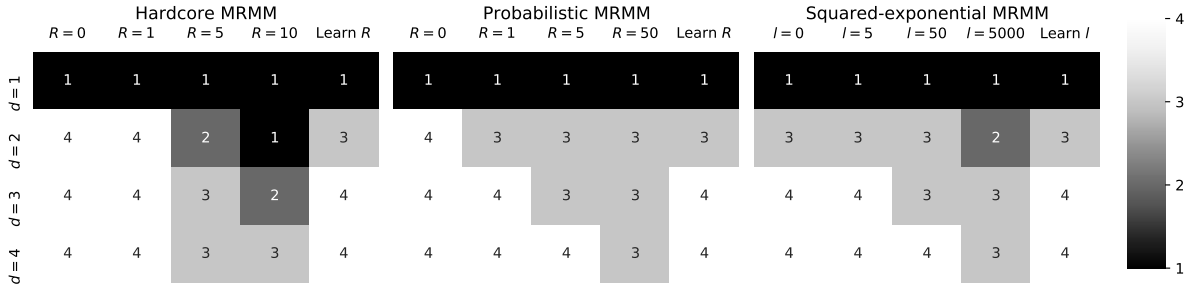

Figure 4.10. Synthetic study in section 4.1.2: The number of clusters estimated from minimizing the posterior expectation of Binder's loss function under equal misclassification costs, $\hat{C}_{\mathrm{B}}$.

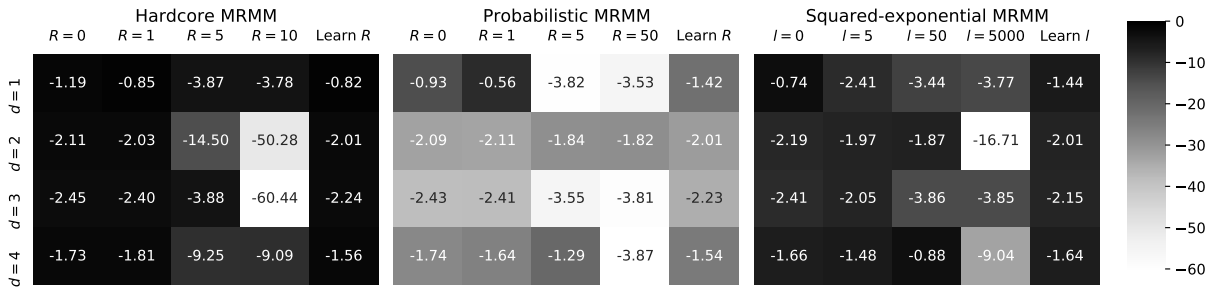

Figure 4.11. Synthetic study in section 4.1.2: The difference between posterior testing likelihood and the testing likelihood under the ground truth model M_0, i.e. $\ln p(\boldsymbol{X}_{\text{test}} \mid \boldsymbol{X}) - \ln p(\boldsymbol{X}_{\text{test}} \mid M_0)$.

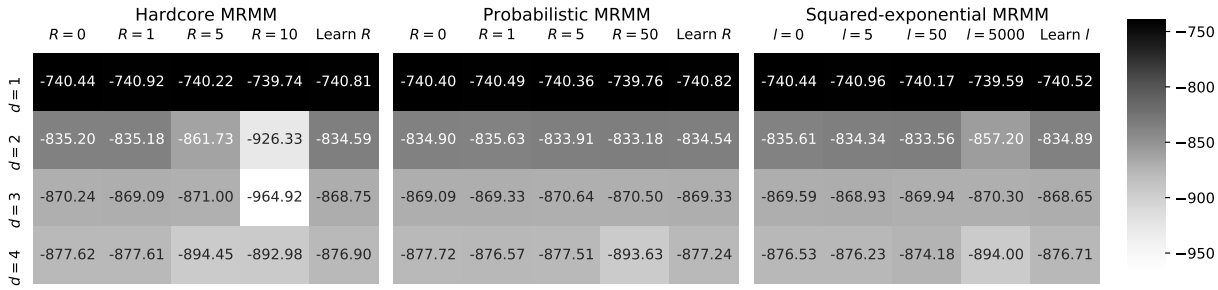

Figure 4.12. Synthetic study in section 4.1.2: The estimated log pseudo-marginal likelihood (LPML).

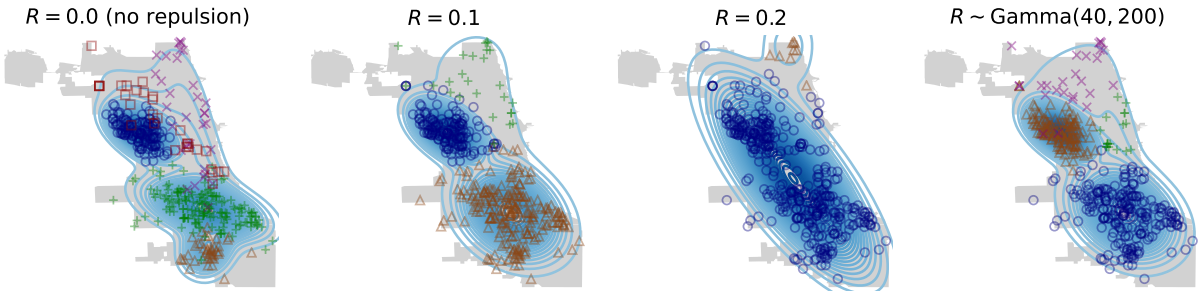

Figure 4.13. Contours and cluster assignments of Chicago crime data with hardcore MRMM in section 4.2.1.

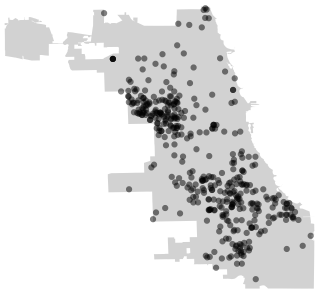

Figure 4.14. Chicago 2019 homicide data.

consisting of the latitude and longitude of each homicide, superimposed on a map of Chicago. These range from $(-87.8066, -87.5293)$ to $(41.6572, 42.0208)$, and we modeled these with MRMM, specifically, a two-dimensional Gaussian mixture model with hardcore repulsion between cluster locations. We set $p_\Theta(\theta)$ to a Gaussian density, with mean $(-87.6727, 41.8180)$ (centered in Chicago), and with variance set to $7 \times 10^{-3} I_2$ (to cover the entire city). We placed an inverse-Wishart prior with 2 degrees of freedom and scale matrix $3.5 \times 10^{-3} I_2$ on the covariance of each Gaussian mixture component. In settings where we wished to learn the thinning radius R, we placed a Gamma$(40, 200)$ prior on R, corresponding to a prior mean of 0.2 and standard deviation of 0.001. For all simulations, we ran 5,000 iterations of our MCMC sampler, and discarded the first 2,500 samples as burn-in.

Figure 4.13 and table 4.2 show the results from the hardcore MRMM with different thinning radii. Across all posterior samples, there were 3 dominant clusters, with the remaining clusters accounting for a small portion of observations. The leftmost panel in figure 4.13 shows the median clustering without any repulsion: here the observations to the south of Chicago are assigned to three clusters. Increasing the repulsion radius to .1 simplifies these three clusters into a single large cluster, even though the observations here deviate slightly from the Gaussian assumption. This is a clear illustration of MRMM being robust to model misspecification. Table 4.2 shows that this simpler model does not come at the cost of a serious loss in predictive power. Increasing the thinning radius to .2 on the other hand causes a steep drop in predictive performance, with a majority of the data points now being assigned to a single cluster (with a few observations to the north-east assigned to their own cluster). Inferring the thinning radius results in a posterior mean and variance $\mathbb{E}[R \mid \boldsymbol{X}] = 0.08$,

Table 4.2. Posterior summaries of hardcore MRMM on Chicago crime dataset in section 4.2.1.

Repulsion strength	$\mathbb{E}[C \mid \boldsymbol{X}]$	$\mathrm{Var}(C \mid \boldsymbol{X})$	$\hat{C}_{\mathrm{B}}$	$\ln p(\boldsymbol{X}_{\mathrm{test}} \mid \boldsymbol{X})$	LPML
$R = 0.0$ (no repulsion)	5.20	0.4028	5	252.54	1349.08
$R = 0.1$	3.51	0.2859	3	248.72	1312.30
$R = 0.2$	2.00	0.0000	2	232.95	1223.68
$R \sim \mathrm{Gamma}(40, 200)$	3.68	0.2416	4	248.51	1318.73

Table 4.3. Posterior summaries of probabilistic MRMM on Chicago crime dataset in section 4.2.1. Inferring the thinning radius yields the posterior mean and variance $\mathbb{E}[R \mid \boldsymbol{X}] = 0.15$, $\mathrm{Var}(R \mid \boldsymbol{X}) = 0.0001$.

Repulsion strength	$\mathbb{E}[C \mid \boldsymbol{X}]$	$\mathrm{Var}(C \mid \boldsymbol{X})$	$\hat{C}_{\mathrm{B}}$	$\ln p(\boldsymbol{X}_{\mathrm{test}} \mid \boldsymbol{X})$	LPML
$R = 0.0$ (no repulsion)	5.26	0.3914	6	252.20	1351.28
$R = 0.1$	4.39	0.2703	5	251.21	1341.60
$R = 0.2$	3.00	0.0000	3	247.43	1321.11
$R \sim \mathrm{Gamma}(40, 200)$	3.00	0.0040	3	246.73	1324.53

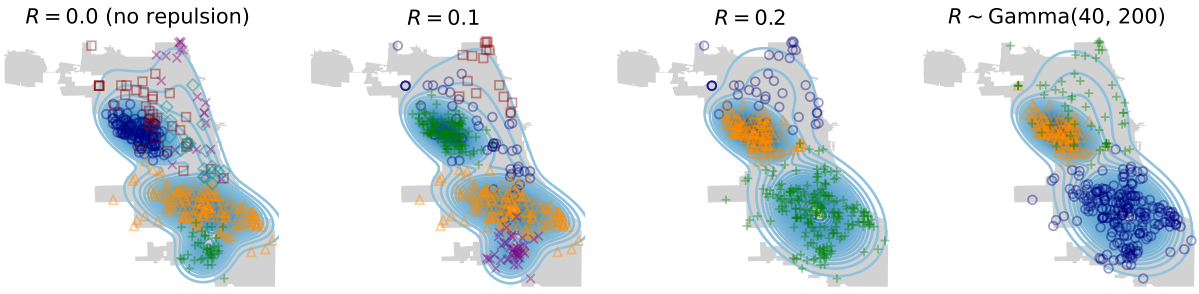

Figure 4.15. Contour plot and clustering of Chicago crime data from probabilistic MRMM in section 4.2.1.

$\mathrm{Var}(R \mid \boldsymbol{X}) = 0.0001$, and achieves a good trade-off between parsimony and goodness-of-fit. Here again, south Chicago is covered by a single cluster instead of multiple clusters as in the no-repulsion case.

Similar results were obtained using probabilistic thinning; see figure 4.15 and table 4.3. One takeaway of this and subsequent experiments is that the more complicated probabilistic and softcore thinning mechanisms discussed in Rao, Adams, and Dunson [30] are not necessary in mixture modeling applications. This is largely due to the fact that the number of mixture components is orders of magnitude smaller than the number of observations or

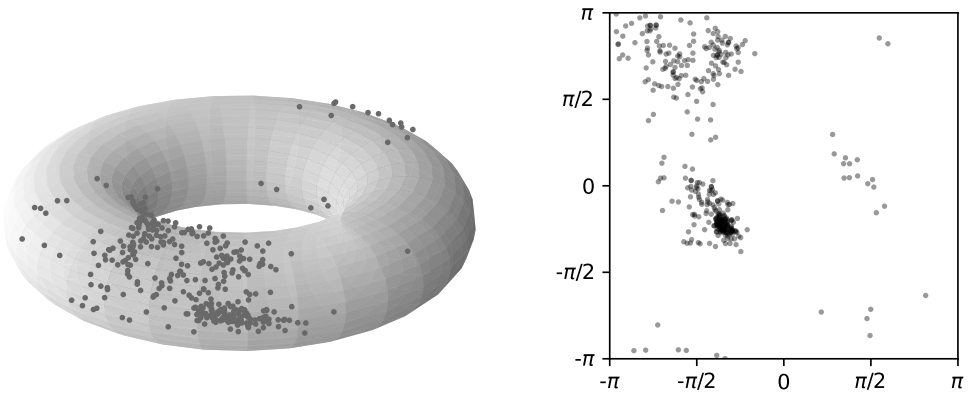

Figure 4.16. The Malate dehydrogenase protein data in section 4.2.2, plotted **(Left)** on a torus. **(Right)** as a Ramachandran plot, where the torus is flattened to 2-d.

events in a point processes. Consequently, the simple hardcore thinning mechanism will typically suffice.

4.2.2 Protein structure data

Our next experiment deals with the Malate dehydrogenase protein dataset, publicly available as `7mdh` in the protein data bank [72]. This consists of 500 pairs of torsion angles, each pair $x = (\phi, \psi) \in [-\pi, \pi) \times [-\pi, \pi)$ forming a point on a torus. Figure 4.16 plots this data, with the right panel showing a planar representation of the data known as the Ramachandran plot [73]. While the latter shows the underlying clustering structure more clearly, it ignores the fact that the edges wrap back to each other. Consequently, modeling this data with common distributions on two-dimensional Euclidean spaces (e.g. mixture of normals or Betas) is not appropriate. Instead, we model this data as a mixture of uncorrelated bivariate von Mises distributions [74].

The univariate von Mises distribution is widely used to model one-dimensional angular variables. For $\phi \in [-\pi, \pi)$, its density is $p(\phi \,|\, \mu, \kappa) = \frac{1}{2\pi I_0(\kappa)} \exp\{\kappa \cos(\phi - \mu)\}$, where μ is the center of the distribution (mean and mode), $\kappa > 0$ measures concentration around this, and the normalization constant $I_m(\cdot)$ is the modified Bessel function of the first kind of order m. This distribution is analogous to the univariate Gaussian distribution in the Euclidean

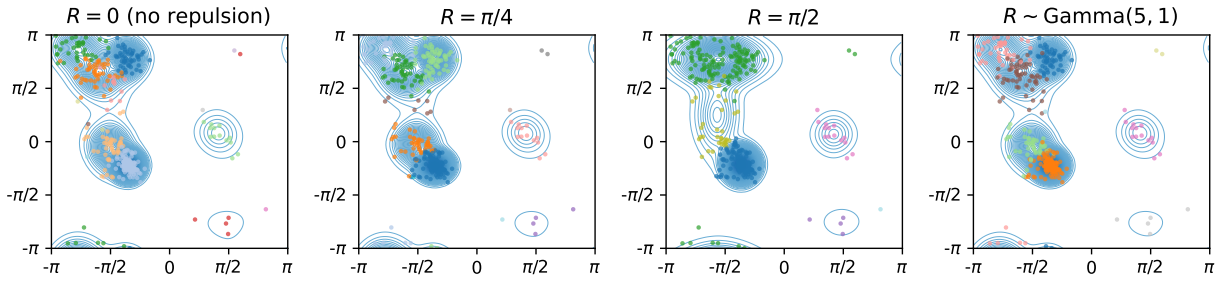

Figure 4.17. Contours and cluster assignments of the protein data from hardcore MRMM in section 4.2.2.

Table 4.4. Posterior summaries of hardcore MRMM on the Malate protein dataset in section 4.2.2.

Repulsion strength	$\mathbb{E}[C \mid \boldsymbol{X}]$	$\mathrm{Var}(C \mid \boldsymbol{X})$	$\hat{C}_{\mathrm{B}}$	$\ln p(\boldsymbol{X}_{\mathrm{test}} \mid \boldsymbol{X})$	LPML
$R = 0$ (no repulsion)	12.29	4.1242	14	-177.43	-644.23
$R = \pi/4$	10.22	1.6658	12	-177.52	-646.58
$R = \pi/2$	5.55	0.3999	6	-199.78	-703.62
$R \sim \mathrm{Gamma}(5, 1)$	11.13	2.5746	9	-177.76	-647.00

space, though it captures the periodicity of the angular variables. It converges to the uniform distribution on $[-\pi, \pi)$ when $\kappa \to 0$. Writing each observation as $x = (\phi, \psi)$, we model these using a Matérn repsulsive mixture model, where under each mixture component, the angles ϕ and ψ are independent von Mises variables. Write the parameters of each mixture component as $\theta = (\mu_1, \mu_2)$ and $\kappa = (\kappa_1, \kappa_2)$, then observations from that component have density $p_{\mathcal{X}}(x = (\phi, \psi); \theta, \kappa) \propto \exp\{\kappa_1 \cos(\phi - \mu_1) + \kappa_2 \cos(\psi - \mu_2)\}$. We set $p_{\Theta}(\theta)$ to the bivariate uniform distribution on $[-\pi.\pi] \times [-\pi, \pi]$, and placed a Gamma$(10, 1)$ prior on the concentration parameter κ. To induce Matérn thinning, we computed distances on the torus as $d_2((\phi, \psi), (\phi, \psi)) = \sqrt{d_1(\phi, \phi)^2 + d_1(\psi, \psi)^2}$, where $d_1(\phi, \phi) = \min\{|\phi - \phi|, \pi - |\phi - \phi|\}$. This distance was used in a standard harcore or probabilistic thinning kernel. We note that we can easily extend our model to more sophisticated geodesic distances, or model each component as a bivariate von Mises distribution with correlations (see Mardia [74] and Mardia, Taylor, and Subramaniam [75]).

We ran 5,000 MCMC iterations on the hardcore MRMM model and discarded the first half as burn-in. Figure 4.17 and table 4.4 show the results with different levels of repulsion.

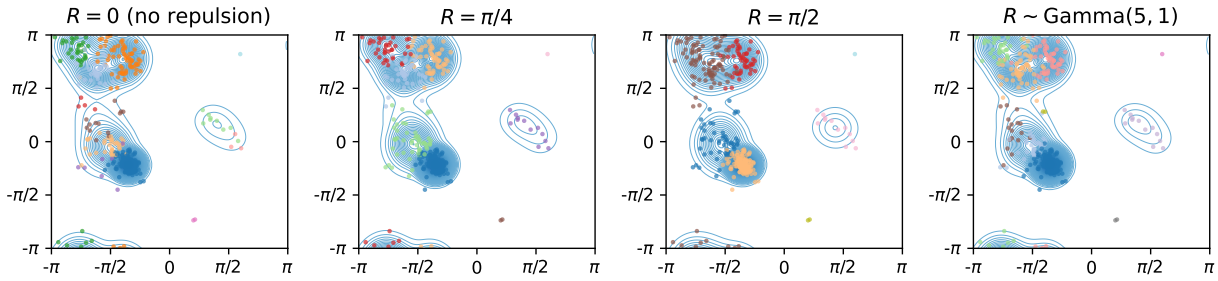

Figure 4.18. Contour plot and clustering of the protein data from probabilistic MRMM in section 4.2.2.

Table 4.5. Posterior summaries of probabilistic MRMM on the protein dataset in section 4.2.2. Inferring the thinning radius yields the posterior mean and variance $\mathbb{E}[R \mid \boldsymbol{X}] = 0.18\pi$, $\mathrm{Var}(R \mid \boldsymbol{X}) = 0.0017\pi^2$.

Repulsion strength	$\mathbb{E}[C \mid \boldsymbol{X}]$	$\mathrm{Var}(C \mid \boldsymbol{X})$	$\hat{C}_{\mathrm{B}}$	$\ln p(\boldsymbol{X}_{\mathrm{test}} \mid \boldsymbol{X})$	LPML
$R = 0$ (no repulsion)	12.15	3.5369	13	-142.15	-610.94
$R = \pi/4$	10.14	1.5298	9	-142.36	-618.13
$R = \pi/2$	7.37	0.4056	7	-145.14	-625.13
$R \sim \mathrm{Gamma}(5, 1)$	10.85	1.9969	11	-143.64	-622.55

Observe from figure 4.16 that the data consists three large clusters of observations, with a couple of smaller clusters. Our model without repulsion returns about 12 clusters on average under the posterior distribution, with the leftmost panel of figure 4.17 showing the median clustering. As with the Euclidean setting, increasing repulsion strength results in fewer clusters, simpler posterior distributions (indicated by smaller posterior variance) and more interpretable results. A strong repulsion ($R = \pi/2$) produced around 5 clusters, agreeing with the findings in Mardia, Taylor, and Subramaniam [75], though resulting in a drop in model fit and predictive power. Placing a Gamma$(5, 1)$ prior (mean 5, variance 5) on the thinning radius infers weaker repulsion (a posterior mean and variance for R equal to 0.19π and $0.017\pi^2$), and thus more clusters (11 on average). These results are partly because of our choice of cluster likelihoods, where the the two angles are independent under each cluster. The cluster near the origin on the other hand exhibits strong correlation between the angles, and our MRMM model has to split this into two (figure 4.17, right). As stated earlier, extending our model to allow correlated clusters is conceptually straightforward using the the

bivariate von Mises distribution. This however introduces normalization constants for each cluster that can be quite challenging to compute, requiring techniques from Rao, Lin, and Dunson [76] and Lin, Rao, and Dunson [77]. To avoid unnecessary complications, we have not followed this path. We emphasize though that modeling repulsion on non-Euclidean spaces using existing models is a less straightforward proposition. We also apply the probabilistic MRMM and have similar results on this dataset (see figure 4.18 and table 4.5).

4.2.3 Old Faithful dataset

The Old Faithful geyser eruption dataset [78] is a well-known dataset, recording eruption lengths of the Old Faithful geyser in the Yellowstone National Park. In Xie and Xu [60], the authors evaluated their model on this dataset, and in this section, we use it to compare our model with theirs. Following Xie and Xu [60], we paired each observed eruption duration time with the time length of the next eruption, resulting in 271 bivariate observations. We split this into training and test sets, with size 219 and 52 respectively.

Consistent with the setup of Xie and Xu [60], we used a Gaussian distribution for $p_\Theta(\theta)$, centered at $(0, 0)$, and with covariance $10I_2$. For the covariance matrix of each mixture component, Xie and Xu [60] assumed independence between the two dimensions and placed truncated inverse Gamma$(1, 1)$ priors on the diagonal elements. For MRMM, we used the more natural inverse-Wishart prior with 2 degrees of freedom and scale matrix I_2 on the covariance matrices. We set the repulsive parameter of Xie and Xu [60] to its default setting of their code (also the setting in their experiments) below. For different settings of the thinning radius of hardcore MRMM, we ran 5,000 MCMC iterations. Because of issues with MCMC mixing, we had to run the model of Xie and Xu [60] for 10,000 iterations. For both models, we discarded the first half of the samples as burn-in, and report posterior summaries of both approaches in table 4.6 and Figure 4.19.

First, observe that this dataset consists of four clearly separated clusters, and for all models, the posterior mean of the number of clusters was around this value. MRMM returns slightly higher estimates of this quantity compared to Xie and Xu [60], but with a much smaller sample variance, suggesting that the posterior is simpler and more concentrated. So

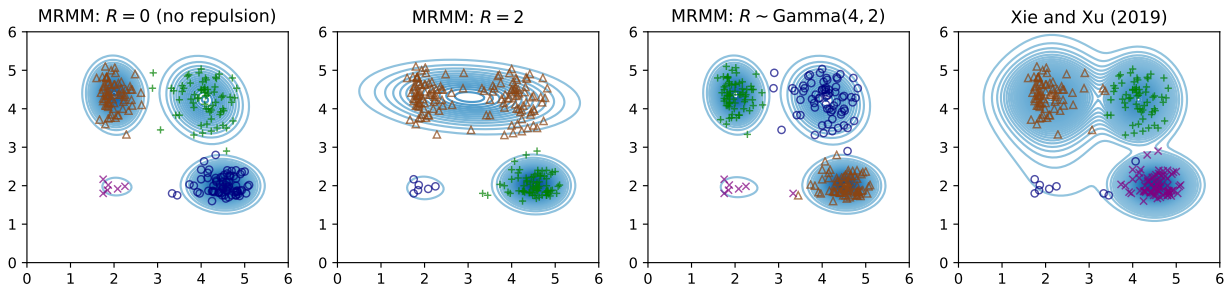

Figure 4.19. Contours and cluster assignments of Old Faithful dataset with hardcore MRMM in section 4.2.3.

Table 4.6. Posterior summaries of hardcore MRMM on the Old Faithful geyser eruption data in section 4.2.3.

Model	$\mathbb{E}[C \mid \boldsymbol{X}]$	$\text{Var}(C \mid \boldsymbol{X})$	$\hat{C}_{\text{B}}$	$\ln p(\boldsymbol{X}_{\text{test}} \mid \boldsymbol{X})$	LPML	Runtime (s)	ESS/s
Xie *et al.*	3.71	0.2116	4	-104.32	-464.22	225.6	0.01
MRMM:							
$R = 0$	4.02	0.0177	4	-95.80	-421.17	266.5	0.67
$R = 2$	3.00	0.0000	3	-114.84	-489.83	251.1	5.54
$R \sim \text{Gamma}(4, 2)$	4.01	0.0119	4	-95.77	-420.54	279.4	0.07

long as the thinning radius is not forced to too large a value, MRMM also returns much better fits, both in terms of predictive likelihood and LPML. Observe that both the model of Xie and Xu [60] and MRMM with $R = 2$ merge the two top clusters into a large cluster, whereas other settings of MRMM keep them clearly separated. In these settings, instead of compromising model fit, MRMM tends to simplify the posterior by concentrating around this solution, and avoiding additional extraneous clusters. This is also the case with a $\text{Gamma}(4, 2)$ prior on R, here the thinning radius has posterior mean $\mathbb{E}[R \mid \boldsymbol{X}] = 1.40$ and variance $\text{Var}(R \mid \boldsymbol{X}) = 0.1864$, with an average of 4 clusters.

We also reported the CPU run time to produce 5000 samples in table 4.6, with both requiring roughly the same time per iteration (note though that Xie and Xu [60] implemented their approach in Matlab and MRMM was written in Python). As we noted earlier, mixing in their case was poorer, and we had to run their algorithm for twice the number of iterations as ours to get stable results. This can also be seen in the reported ESS/s numbers, where

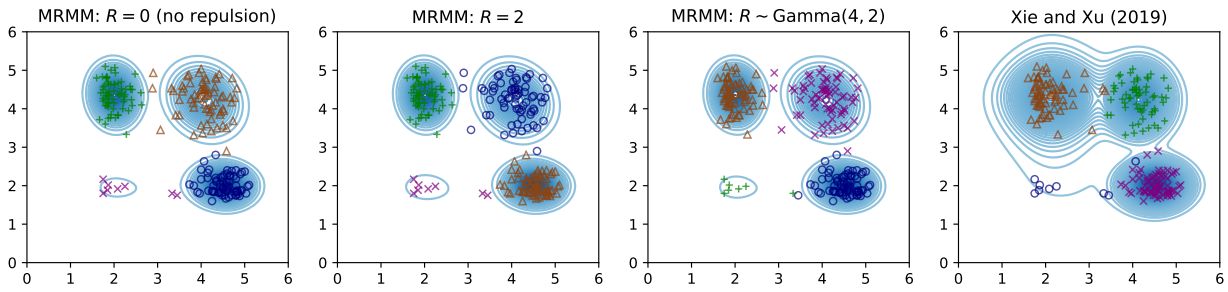

Figure 4.20. Contour plot and clustering of the Old Faithful geyser eruption data from probabilistic MRMM in section 4.2.3.

Table 4.7. Posterior summaries of probabilistic MRMM on the Old Faithful geyser eruption data in section 4.2.3. Inferring the thinning radius yields the posterior mean and variance $\mathbb{E}[R \mid \boldsymbol{X}] = 1.39$, $\text{Var}(R \mid \boldsymbol{X}) = 0.1540$.

Model	$\mathbb{E}[C \mid \boldsymbol{X}]$	$\text{Var}(C \mid \boldsymbol{X})$	$\hat{C}_{\text{B}}$	$\ln p(\boldsymbol{X}_{\text{test}} \mid \boldsymbol{X})$	LPML	Runtime(s)	ESS/s
Xie *et al.*	3.71	0.2116	4	-104.32	-464.22	225.6	0.01
MRMM:							
$R = 0$ (no repulsion)	4.02	0.0157	4	-95.83	-420.53	257.8	14.31
$R = 2$	4.00	0.0000	4	-95.93	-419.85	297.6	0.38
$R \sim \text{Gamma}(4, 2)$	4.01	0.0138	4	-95.96	-420.94	287.0	2.66

our sampler shows larger (often much larger) values. Running probabilistic MRMM yields similar results, as shown in figure 4.20 and table 4.7.

4.2.4 Galaxy dataset

The Galaxy dataset [79] is publicly available as part of the `DPpackage` in `R`, and contains 82 measured velocities of different galaxies from six well-separated conic sections of space. In Bianchini, Guglielmi, Quintana, *et al.* [58], the authors evaluated their model on this well-known dataset, using LPML as their goodness-of-fit criteria. We do the same in this section. Following the same preprocessing steps described in Bianchini, Guglielmi, Quintana, *et al.* [58], we centered the data and rescaled it by a factor of 10^{-3}, which resulted in a dataset ranged from -11.65 to 13.45. Like Bianchini, Guglielmi, Quintana, *et al.* [58], we set $p_\Theta(\theta)$ to place a mean 0 and standard deviation 10 Gaussian prior on the cluster locations, and an inverse-Gamma$(3, 3)$ prior on the variance of each mixture component.

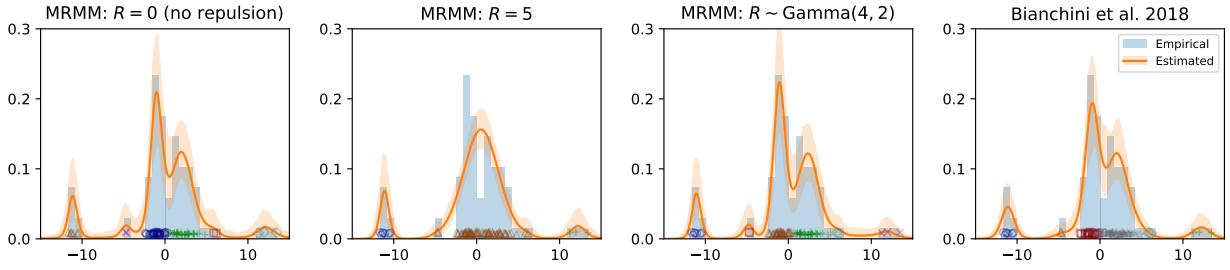

Figure 4.21. Contour plot and cluster assignments of the Galaxy data for hardcore MRMM in section 4.2.4.

Table 4.8. Posterior summaries for the Galaxy dataset inferred with hardcore MRMM in section 4.2.4.

Model	$\mathbb{E}[C \mid \boldsymbol{X}]$	$\mathrm{Var}(C \mid \boldsymbol{X})$	$\hat{C}_{\mathrm{B}}$	LPML	Runtime (s)	ESS/s
Bianchini *et al.*	6.00	1.2180	7	-207.94	600.4	0.02
MRMM:						
$R = 0$ (no repulsion)	7.69	4.0819	6	-210.13	772.9	0.83
$R = 5$	3.37	0.3046	3	-212.05	448.2	4.50
$R \sim$ Gamma$(4, 2)$	5.51	0.9339	6	-208.83	501.2	0.03

We ran both samplers for 10,000 iterations, and discarded the first 5,000 iterations as burn-in. Figure 4.21 and table 4.8 present the results of MRMM with hardcore thinning, along those of with Bianchini, Guglielmi, Quintana, *et al.* [58].

The left-most panel in figure 4.21 dispays the mean posterior density for the model without any repulsion. For this model, the posterior mean number of clusters is around 8, with a relatively large variance of 4. The two rightmost panels show the corresponding densities for MRMM (with a Gamma prior on the thinning radius), and the model of Bianchini, Guglielmi, Quintana, *et al.* [58]. Both models have about 6 clusters, though the posterior density or the predictive performance is not significantly different from the model without repulsion. By forcing the thinning radius to 5, the clusters around the origin merge into a single cluster. Whether this is an appropriate amount of repulsion must be determined by the practitioner, though we note that even here, the drop in performance, while noticeable, is not very large. With a Gamma$(4, 2)$ prior on R, we get a posterior mean $\mathbb{E}[R \mid \boldsymbol{X}] = 1.54$ and variance $\mathrm{Var}(R \mid \boldsymbol{X}) = 0.5305$, with the posterior mean of the number of clusters about

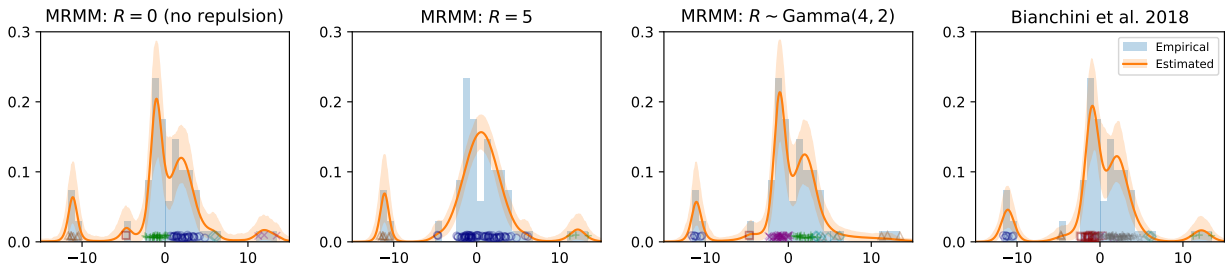

Figure 4.22. Contour plot and clustering of the Galaxy data from probabilistic MRMM in section 4.2.4.

Table 4.9. Posterior summaries of probabilistic MRMM on the Old Faithful geyser eruption data in section 4.2.4. Inferring the thinning radius yields the posterior mean and variance $\mathbb{E}[R \mid \boldsymbol{X}] = 1.87$, $\text{Var}(R \mid \boldsymbol{X}) = 0.3228$.

Model	$\mathbb{E}[C \mid \boldsymbol{X}]$	$\text{Var}(C \mid \boldsymbol{X})$	$\hat{C}_{\text{B}}$	LPML	Runtime (s)	ESS/s
Bianchini *et al.*	6.00	1.2180	7	-207.94	600.4	0.02
MRMM:						
$R = 0$ (no repulsion)	7.53	4.2370	6	-209.66	734.4	46.9
$R = 5$	3.47	0.3772	3	-212.36	410.4	172.4
$R \sim \text{Gamma}(4, 2)$	6.23	1.8120	6	-209.43	498.2	13.0

5.5. With the caveat that Bianchini, Guglielmi, Quintana, *et al.* [58] was written in `R`, and our model in Python, we report the CPU run times and ESS/s of both methods in table 4.8. These are comparable. With probabilistic MRMM, we obtained similar results, and they are reported in figure 4.22 and table 4.9.

5. CLUSTERING POPULATIONS WITH HIERARCHICAL STRUCTURES

In the previous two chapters, we discussed the Matérn repulsive mixture model, where structural restrictions of cluster separation were introduced to encourage interpretability and parsimony of the results. In the following two chapters, we consider another setting of clustering under structural restrictions, in this case incorporating prior knowledge about an underlying tree-structure to encourage statistical sharing among clusters.

Modern datasets are characterized by rich underlying structure, resulting from the mechanistic, spatio-temporal processes that led to their generation. Trees are a widely used example, representing a hierarchical organization of observations into partially overlapping sets at multiple granularities. A classic example, and one that is the focus of this work, are phylogenetic trees, showing relationships between various entities evolving from a common ancestor. The entities in a phylogenetic tree are typically biological species, though in this work we take a broader view, and also consider evolving languages and other social norms. Accounting for the underlying tree structure is important to understand relationships between and variations among the different entities in the tree, and allows practitioners to share statistical strength between different sets of observations.

A number of existing approaches model evolution on the tree at the individual level, with each leaf corresponding to a single measurement. In phylogenetic applications where each node corresponds to a species, this would correspond to each species having an associated phenotype. This phenotype then evolves along the tree, either gradually according to some diffusion processes (e.g. Brownian motion [80], [81]), or abruptly through a series of jumps, the latter modelled by the pure jump part in a Levy processes (often a compound Poisson process) [82]–[84].

In the setting of population genetics, the assumption of a single phenotype for each species implies an assumption of complete heritability. In reality, the measured phenotype can vary among individuals of the same species due to environmental factors [85], [86]. Now, to understand the influence of genetic as well as environmental factors on the expression of phenotypes, one needs models that operate at the population-level [87]. Such models allow

multiple measurements at each leaf, corresponding for instance to a number of non-identical individuals from each species. Population-level models also arise in other social contexts where individuals of the population at each leaf can exhibit varying values.

A simple statistical approach to population-level modeling treat observations from each population (i.e. at each leaf) as independent and identical draws from the distribution associated with that node, with the distributions at each leaf linked by the the tree structure. The latter is typically achieved by allowing the phenotype distribution to evolve along the tree. Modeling general dependent probability distributions presents statistical and computational challenges, and instead, it is common to model the distribution at each node as an element of some parametric family of probability distributions, e.g. the Gaussian distribution. The parameters of this distribution then evolve along the tree as before, again either gradually or through a jump process.

Parametric modeling approaches, while simple to work with, come with strong assumptions on the distributions of observations at each node, assumptions that are typically not satisfied in practice. A more flexible approach models these distributions with nonparametric priors that have much larger support over the space of probability distributions, and that allow modelers to approximate arbitrary distributions. An early work in this direction is that of Ansari and Didelot [37], who considered categorical measurements, and modeled these by associating a Dirichlet distribution with each node. While the Dirichlet distribution is strictly speaking a parametric density summarized by a finite number of parameters, under mild conditions, it support includes all distributions on some categorical space. This opens the path to truly nonparametric priors like the Dirichlet process that can approximate arbitrary probability distributions. The work of Ansari and Didelot [37] models the evolution of the node distributions with a series of Poisson distributed jumps, each jump triggering a new distribution drawn independently from a Dirichlet distribution. This assumption that the distributions before and after each jump are completely independent with each other considerably weakens the connection between species within the phylogenetic tree, and can result in overfitting with nodes with few observations.

In this work, we take a fully nonparametric hierarchical Bayesian approach to model evolving distributions on a phylogenetic tree. Our work uses the Pitman-Yor process as its

nonparametric workhorse because of a convenient marginalization property that it possesses, a property that allows us to integrate out intermediate clusters. In this chapter, we describe the model, as well as the associated MCMC sampler, while the next chapter evaluates these on real and synthetic datasets.

5.1 Phylogenetic HPYP model

We consider data consisting of groups of measurements organized along a given tree. Two specific examples are phenotype distributions of different species, or behavioral patterns of different human populations. In the former, the tree represents the evolutionary history of the species under consideration, and the in the latter, it might represent migration patterns of different human subpopulations. The branch lengths quantify the statistical dissimilarity between parent and child distributions. Each node in the tree, whether an internal node or a leaf, represents a population, and has associated with it a collection of zero or more observations. Denote the observed dataset as $\mathcal{D} = \left\{x_{(\mathrm{i},\mathrm{j})},\ \mathrm{i} = 1, \ldots, N,\ \mathrm{j} = 1, \ldots, n_{\mathrm{i}}\right\}$, where N is the number of nodes and n_{i} is the number of observations at node i. In practice, observations are often present only at the leaf nodes (e.g. measured traits from today species at the leaf nodes), but our model allows observations at internal nodes (e.g. measurements from fossil data).

We model each set of observations as independent and identical draws from a corresponding probability distribution, with the hierarchical dependence among these distributions determined by the tree structure. We will use G_{i} to denote the distribution at node i. The distributions G_{i} and G_{j} at two distinct nodes i and j may or may not be identical. Specifically, in this work, we aim to detect significant changes of the evolution of the distributions, which we represent as "jumps" distributed over the tree. Starting from the root, and moving towards the leaves, the distribution remains constant until a jump is encountered, after which a new distribution is sampled, and the process recurses. We model the jumps as a realization of a Poisson process, so that there might be zero, one or multiple jumps on each branch of the tree. If there are no jumps between two nodes i and j, then the associated distributions G_{i} and G_{j} are identical. It follows then that the shorter the separation between two nodes

is, the more likely their associated distributions are identical. In the event of a jump, Ansari and Didelot [37] models the new distribution G_{new} as independent of the old distribution G_{old} (specifically they model it as a draw from a fixed-parameter Dirichlet distribution). A more realistic assumption is that the new distribution is generated from some distribution centered at the old one. This hierarchical modeling approach has two advantages. First, under the approach of Ansari and Didelot [37], conditioned on there being at least one jump between two nodes, the distribution of the child node is independent of the actual number of jumps. By contrast, by centering the new distribution on the old one, the similarity decreases as the number of jumps increases (specifically, the new distribution continues to be centered at the old one, with variance increasing with the number of jumps). Secondly, a hierarchical modeling approach is useful in data-scarce settings, where individual nodes might have only a few observations, and it is important to pool information across multiple nodes. Under the approach of Ansari and Didelot [37], since the new distribution is independent of the old, it can only be informed by the observations directly associated with it. This can result in overfitting if only a few observations are associated with a complex nonparametric distribution. By contrast, organizing all distributions together in a hierarchical fashion models distributions separated by fewer jumps as more similar, so that observations associated with other distributions have a non-zero influence that diminishes with jump-separation. By shrinking the distributions towards each other, we can avoid overfitting, and allow better predictions.

Since the distribution only changes when a jump occurs, the jump structure specifies a clustering of nodes, with nodes having the same distribution belonging to the same cluster. As a consequence, tree nodes in the same cluster can be collapsed into a single node, with all associated observations assigned to it, a process we call pruning. Figure 5.1(a) provides an abstract example of such structure with nine populations, the root population G_0, five leaf populations $G_1, \ldots, G_5$ and three internal populations G_6, G_7 and G_8. It also shows the clustering (color blocks) introduced by the jumps (red crosses) at two branches in the tree. Here, the distribution at root G_0 is the same as G_7 and G_3. There are two jumps at the left branch of the root, but only one jump at the right branch of G_7, which means that the left cluster (G_1, G_2 and G_6) has a weaker dependence on the root distribution compared to the right cluster (G_4, G_5 and G_8). Figure 5.1(b) demonstrates the simplified tree introduced

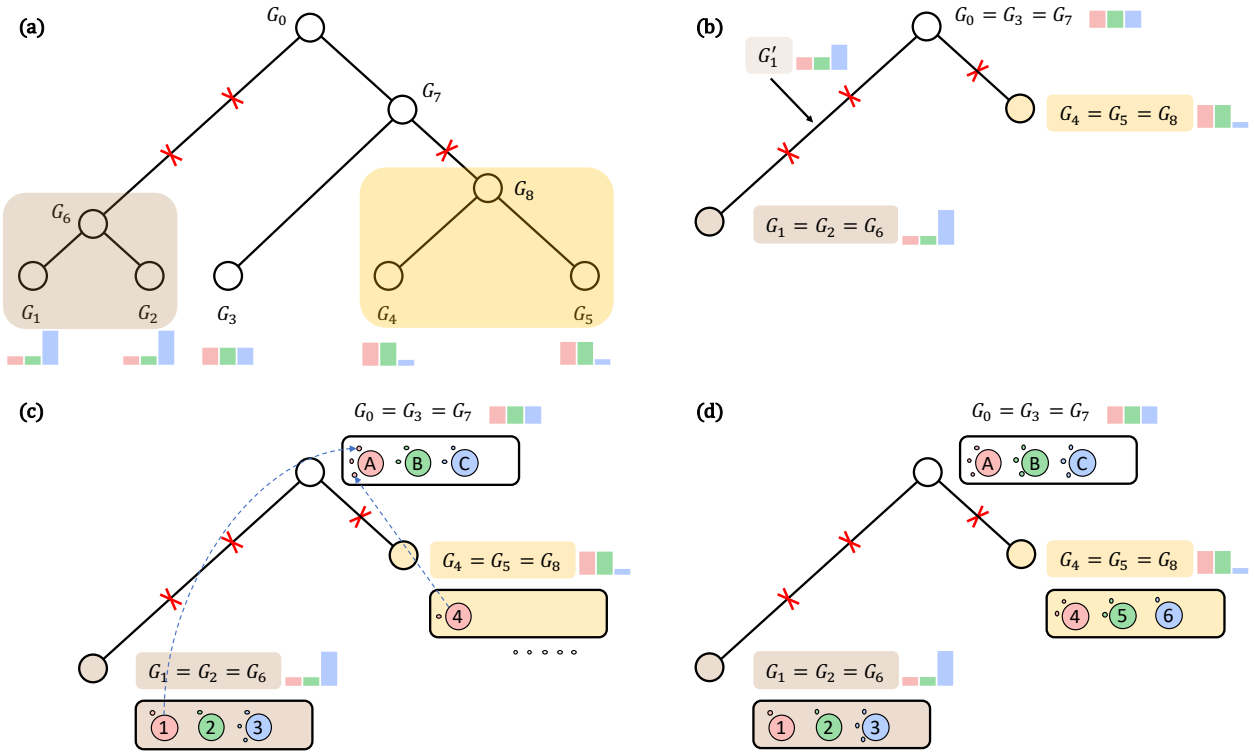

Figure 5.1. An example of tree node clustering introduced by jumps (red crosses) and the corresponding Chinese restaurant franchise (CRF) process in action. **(a)** Color coded clustering of distributions (nodes). The bar graphs shows the distributions at corresponding leaf nodes. **(b)** The simplified tree induced by the jumps in (a), with each cluster represented by one node. Specially, G_1' represents the intermediate distribution between the two jumps on the left branch. The underlying distributions are shown by bar graphs. **(c)** A step in CRF when all samples (observations) belongs to the brown cluster G_1 and the white cluster G_0 as well as the first sample from the yellow cluster G_4 are all seated. Five more samples from the yellow cluster are still waiting to be seated (assigned to tables). The tables are color-coded according to the categories in the bar plots. This is a valid representation only when G_1' can be marginalized out. **(b)** The final seating chart of all samples (observations).

by these jumps and the distributions associating to it. Note that there is a intermediate distribution G_1' between the root and the left cluster G_1, though as we will see below, our modeling does not require us to instantiate the intermediate distributions. In this figure, like in the rest of this work, we focus on the situation where the distributions are discrete, with the bar graphs denoting discrete distributions at leaf nodes, and from which observations are drawn independently.

In what follows, we formalize the intuition above into a mathematical model.

Hierarchical Pitman-Yor process (HPYP) for the simplified tree

A key component of our model is the Pitman-Yor process Pitman and Yor [48], a non-parametric prior that we use to model the distributions G_{i} (see section 2.4 for a brief review).

Recall that a Pitman-Yor process is specified by a concentration parameter α, a discount parameter $d \in (0, 1)$ and a base probability measure H on some space. A realization from it is a discrete probability measure G on the space, $G \,|\, \alpha, d, H \sim \text{Pitman-Yor}\,(\alpha, d, H)$. The base distribution H serves as the center of the prior, with $\mathbb{E}[G] = H$. The concentration and discount parameter satisfy $\alpha \geq -d$ and control the shape and spread of this prior around H. The Chinese restaurant process (CRP), describing the distribution of observations drawn i.i.d. from G (with G marginalized out) is defined as follows. The first observation x_1 is generated directly from the base H. Conditioning on the first j observations, the $(\mathrm{j}+1)$-st observation is generated from

$$x_{\mathrm{j}+1} \,|\, \{x_1, \ldots, x_{\mathrm{j}}\},\ \alpha,\ d,\ H \sim \sum_{k=1}^{K} \frac{n_k - d}{\alpha + \mathrm{j}} \delta_{y_k} + \frac{\alpha + K \cdot d}{\alpha + \mathrm{j}} H, \tag{5.1}$$

where $K = K(\mathrm{j})$ denotes the total number of clusters (tables) formed by the first j observations, and n_k and y_k are the cluster size and the first observation of cluster k, respectively. This is a mixture of joining an existing cluster and start a new cluster (simulating from the base H) at node i.

We model the root distribution G_0 as a sample from a Pitman-Yor process. Since we focus mostly on discrete spaces, we set the base distribution to the uniform distribution;

on Euclidean spaces, one might set it to the Gaussian distribution. We will also use the Pitman-Yor process to model the change in distribution after each jump. Specifically, if the distribution before a jump is G_{old}, then the new distribution after the jump is sampled from a Pitman-Yor process whose base measure is G_{old}, thereby ensuring it is centered at the old distribution.

$$G_{new} \,|\, \alpha, d, G_{old} \sim \text{Pitman-Yor}\,(\alpha, d, G_{old})\,.$$

This forms a hierarchical Pitman-Yor process (HPYP) on the simplified tree.

We choose the general Pitman-Yor process rather than the more well-known Dirichlet process because of a convenient marginalization property it possesses when the concentration parameter α is zero (see section 2.4 for details). This marginalization property can result in significant savings when there are multiple jumps in a branch. For instance, see the left branch in figure 5.1(b). With the marginalization property, $G_1 \,|\, G_0$ still forms a Pitman-Yor process, $G_1 \,|\, G_0 \sim \text{Pitman-Yor}\,(0, d^2, G_0)$, and hence, there is no need to instantiate the CRP corresponding to the intermediate G_1'. In general, on a branch with b jumps, the child distribution G_{child} is a realization of a Pitman-Yor process centered at the parent distribution G_{parent}, with discount parameter being d^b,

$$G_{child} \,|\, G_{parent} \sim \text{Pitman-Yor}\left(0, d^b, G_{parent}\right). \tag{5.2}$$

Thanks to this property, instead of having to instantiate all the associated measures, we can marginalize out those without any associated observations, only keeping track of the number of jumps in each branch. In Ansari and Didelot [37], the authors do not face this issue since the new distribution is independent of the old one, so that it does not matter whether there is one or more than one jumps on a branch. We have already stated the limitations of this approach.

Chinese restaurant franchise (CRF) to generate observations

To generate observations $\mathcal{D} = \left\{x_{(\mathrm{i},\mathrm{j})},\ \mathrm{i} = 1, \ldots, N,\ \mathrm{j} = 1, \ldots, n_{\mathrm{i}}\right\}$ from HPYP, we use the Chinese restaurant franchise (CRF). This is essentially a coupling of the CRPs associated

with the Pitman-Yor process on every tree node. Consider the node i and its parent i′ in the simplified tree. According to CRP, generating new samples at node i results in a mixture of joining an existing table (taking the same value of a previous sample at node i) and opening a new table generated from the base measure (equation (5.1)). As $G_{\mathrm{i}'}$ is the base measure of node i, opening a new table will request a new sample from the parent $G_{\mathrm{i}'}$. This can again be simulated with the CRP associated with the parent node i′, which might again need a new sample from the parent node of i′.

In general, generating a new observation at a node can result in a sequence of new samples being generated along ancestors of the node. Following the notation of observations, let $x_{(\mathrm{i,j})}$ denote the j-th sample at node i in the CRF generative process of the data. This is either an observation itself or a sample that is generated when a child node of it forms a new table (requires a new sample from the current node). Denote the table assignment of $x_{(\mathrm{i,j})}$ by an integer $t_{(\mathrm{i,j})}$, which represents the cluster joined by $x_{(\mathrm{i,j})}$ at node i. Because of the construction of CRF, generating one sample may lead (through the creation of new tables) to multiple samples being generated at ancestor nodes. To describe this behavior, we define the the table configuration of sample (i, j), $c_{(\mathrm{i,j})}$, which is a vector of varying length, to represent the series of table assignments driven by the sample $x_{(\mathrm{i,j})}$ in the generative process. The table configuration vector is of the form $c_{(\mathrm{i,j})} = (t_{(\mathrm{i,j})}, t_{(\mathrm{i}',\mathrm{j}')}, t_{(\mathrm{i}'',\mathrm{j}'')}, ...)$, where i′, i″, ... are the ancestor nodes of node i, and $t_{(\mathrm{i}',\mathrm{j}')}$, $t_{(\mathrm{i}'',\mathrm{j}'')}$, ... represent the table assignments of the samples generated when new clusters are formed. To be specific, the first element, $t_{(\mathrm{i,j})}$, denotes the cluster joined by $x_{(\mathrm{i,j})}$ at node i. If $t_{(\mathrm{i,j})}$ represents an existing cluster, the table configuration $c_{(\mathrm{i,j})}$ will be of length 1, as there is no new clusters to be generated at the parent node. When $t_{(\mathrm{i,j})}$ represents a new cluster, a new sample will be simulated from the parent node i′, $x_{(\mathrm{i}',\mathrm{j}')}$, and its table assignment $t_{(\mathrm{i}',\mathrm{j}')}$ forms the second element of $c_{(\mathrm{i,j})}$. The dimensionality of $c_{(\mathrm{i,j})}$ continues to grow if $x_{(\mathrm{i}',\mathrm{j}')}$ creates new clusters which leads to a new sample at the parent node of node i′. This process stops when a sample joins an existing cluster. The length of $c_{(\mathrm{i,j})}$ varies according to the number of new clusters generated in the process and the maximum length is the depth of node i.

The table configuration fully describes the generative process of one single observation. Taking this procedure to all observations in the simplified tree forms the full CRF process in generating the data.

The only parameter to tune in this model is the discount parameter d. Our experiment in section 6.1.1 shows that as long as it does not take values close to one, our model will perform well.

Inhomogeneous Poisson process for the jumps

As stated before, we model the jumps or changepoints as a realization of a rate-λ Poisson process, The Poisson process can be homogeneous (with λ a constant) or inhomogeneous, with $\lambda(p)$ varying with position p on the tree. In either event, the probability of a jump in an infinitesimal interval $(p, p+\Delta p)$ on the tree is $\lambda(p)\Delta p$. A special choice of the inhomogeneous rate that is very natural in many settings involves the use of evolutionary time. Here, every point p on the tree has an associated time t, with t increasing along each branch, and with the root having $t = 0$. Now, the Poisson intensity is indexed by t, and might reflect global events (e.g. climate shifts) that modulate the rate of jumps. Observe that for a balanced tree, the number of branches increases exponentially with evolutionary time, and for a homogeneous Poisson process, so does the average number of jumps. Thus, a time-varying intensity function $\lambda(t)$ is also useful to control the number of jumps: by allowing it to decrease appropriately with time, one can ensure that the probability of a jump is constant over time. Note that our model only cares about the number of jumps in each branch, and not their exact positions along the branch, As a consequence, we can also keep the probability of a jump constant by modeling the jumps as a homogeneous Poisson process, but first rescaling the branch lengths. This is the approach we take in our experiments: it is computationally slightly simpler than simulating from an inhomogeneous Poisson process, and also allows prior distributions on the Poisson rate to be specified more easily. When learning the jump rate, we place an exponential distribution prior over it. The synthetic study in section 6.1.2 further investigates the performance of our model with different prior settings of the jump rate, and we find that it is generally robust to model misspecification.

Let $\boldsymbol{b} = (b_1, \ldots, b_{|\boldsymbol{b}|})$ denote the vector of number of jumps on each branch of the tree. The full generative process for observing the data $\mathcal{D} = \left\{x_{(\mathrm{i},\mathrm{j})}\right\}$ at the nodes $\mathrm{i} = 1, \ldots, N$ is as follows,

1. Rescale the tree branches to have a constant jump rate over time (as described above).
2. Simulate the Poisson intensity λ from the prior.
3. Instantiate $\boldsymbol{b}$ from a rate-λ Poisson process on the rescaled tree.
4. Prune the tree according to the jumps to obtain the simplified tree, where each node represent a cluster of populations in the original tree.
5. Construct HPYP according to the jumps on the simplified tree.
6. Generate observations sequentially according to the Chinese restaurant franchise (CRF) associated with the HPYP of the simplified tree.

5.2 Posterior inference for phylogenetic HPYP model

Given the data $\mathcal{D} = \left\{x_{(\mathrm{i},\mathrm{j})},\ \mathrm{i} = 1, \ldots, N,\ \mathrm{j} = 1, \ldots, n_{\mathrm{i}}\right\}$, we aim to determine whether there are jumps in the tree, and to locate the branches with jumps if they exist. Given the jumps, the posterior distribution over the node distributions follows from the CRF. Thus, the key quantity of interest is the posterior distribution $p(\boldsymbol{b}, \lambda \,|\, \mathcal{D})$. Before describing the full details of the Gibbs sampler we proposed for posterior inference, we will first introduce a little notation below.

Observations generated with HPYP are exchangeable, i.e. observing the data in any order will not change the model. Without loss of generality, write the sequence of observations being considered as $x_{(\mathrm{i}_1,\mathrm{j}_1)}, x_{(\mathrm{i}_2,\mathrm{j}_2)}, \ldots, x_{(\mathrm{i}_n,\mathrm{j}_n)}$, where $n = \sum_{\mathrm{i}} n_{\mathrm{i}}$ is the total number of observations. We use symbols '$-$' and '$+$' on the superscript of the index to represent the indices of the preceding and succeeding observations, i.e. $(\mathrm{i}_k, \mathrm{j}_k)^- = (\mathrm{i}_{k-1}, \mathrm{j}_{k-1})$ and $(\mathrm{i}_k, \mathrm{j}_k)^+ = (\mathrm{i}_{k+1}, \mathrm{j}_{k+1})$. Let $\mathcal{D}_{(\mathrm{i},\mathrm{j})}$ and $\mathcal{C}_{(\mathrm{i},\mathrm{j})}$ be the collection of data observed until observation (i, j) (included) and their corresponding table configurations. Similarly, we use $\mathcal{D}_{(\mathrm{i},\mathrm{j})^-}$ and

$\mathcal{C}_{(\mathrm{i},\mathrm{j})^-}$ for the collection of data and associated table configurations prior to observation (i,j), and $\mathcal{D}_{(\mathrm{i},\mathrm{j})^+}$ and $\mathcal{C}_{(\mathrm{i},\mathrm{j})^+}$ for their counterparts until the succeeding observation $(\mathrm{i},\mathrm{j})^+$ (included).

The posterior inference of jump locations $\boldsymbol{b}$ is done using a Gibbs sampler (algorithm 5). In what follows, we will first give details of the updating rules and then describe how an estimation of the jumps $\boldsymbol{b}$ can be produced given the posterior samples.

1) Updating the jumps

Jumps in the tree are initialized according to the prior distribution. At each step of our Gibbs iteration, we will use a Metropolis-Hastings algorithm to update the jumps, i.e. given current number of jumps $\boldsymbol{b}$, a new proposal $\boldsymbol{b}^*$ is generated from a proposal distribution $q(\boldsymbol{b}^* \mid \boldsymbol{b})$ and the acceptance rate of the proposal is $\min\{1, A(\boldsymbol{b}, \boldsymbol{b}^*)\}$, where

$$\begin{aligned} A(\boldsymbol{b}, \boldsymbol{b}^*) &= \frac{p(\boldsymbol{b}^* \mid \lambda, \mathcal{D})\, q(\boldsymbol{b} \mid \boldsymbol{b}^*)}{p(\boldsymbol{b} \mid \lambda, \mathcal{D})\, q(\boldsymbol{b}^* \mid \boldsymbol{b})} = \frac{p(\boldsymbol{b}^* \mid \lambda, \mathcal{D})\, p(\mathcal{D} \mid \lambda)\, q(\boldsymbol{b} \mid \boldsymbol{b}^*)}{p(\boldsymbol{b} \mid \lambda, \mathcal{D})\, p(\mathcal{D} \mid \lambda)\, q(\boldsymbol{b}^* \mid \boldsymbol{b})} \\ &= \frac{p(\mathcal{D} \mid \boldsymbol{b}^*)}{p(\mathcal{D} \mid \boldsymbol{b})} \cdot \frac{p(\boldsymbol{b}^* \mid \lambda)\, q(\boldsymbol{b} \mid \boldsymbol{b}^*)}{p(\boldsymbol{b} \mid \lambda)\, q(\boldsymbol{b}^* \mid \boldsymbol{b})} \end{aligned} \tag{5.3}$$

Evaluating the acceptance rate requires accessing three quantities, the prior distribution of jumps $p(\boldsymbol{b} \mid \lambda)$, the proposal distribution $q(\boldsymbol{b}^* \mid \boldsymbol{b})$ and the data likelihood $p(\mathcal{D} \mid \boldsymbol{b})$.

The prior distribution $p(\boldsymbol{b} \mid \lambda)$ is simply a product of independent Poisson densities, which is easy to evaluate.

A natural way to propose a new vector of number of jumps $\boldsymbol{b}$ is to first randomly select a branch and then propose the number of jumps from the prior. However, due to the fact that it is very hard for a jump to move upwards or downwards in the tree under this proposal scheme, it does not mix well in practice. To address this issue, we further implemented a swap step, which swaps the number of jumps between a randomly selected branch and its parent. Taking the two kinds of proposals in turns helps the mixing of our algorithm dramatically.

Now all that is left is the likelihood term. The likelihood of the data given the jumps, $p(\mathcal{D} \mid \boldsymbol{b})$, is not tractable, however, we could circumvent it through a pseudo-marginal ap-

proach [88]. To be specific, when the likelihood $p(\mathcal{D} \mid \boldsymbol{b})$ has an unbiased estimator $\hat{p}(\mathcal{D} \mid \boldsymbol{b})$, accepting the proposal with

$$A(\boldsymbol{b}, \boldsymbol{b}^*) = \frac{\hat{p}(\mathcal{D} \mid \boldsymbol{b}^*)}{\hat{p}(\mathcal{D} \mid \boldsymbol{b})} \cdot \frac{p(\boldsymbol{b}^* \mid \lambda)\, q(\boldsymbol{b} \mid \boldsymbol{b}^*)}{p(\boldsymbol{b} \mid \lambda)\, q(\boldsymbol{b}^* \mid \boldsymbol{b})} \tag{5.4}$$

still yields a valid Metropolis-Hastings update. To construct the unbiased estimator, we rewrite the likelihood as

$$\begin{aligned} & p(\mathcal{D} \mid \boldsymbol{b}) = \sum_{\mathcal{C}} p(\mathcal{D}, \mathcal{C} \mid \boldsymbol{b}) \\ = \;& p\left(x_{(1,1)} \,\middle|\, \boldsymbol{b}\right) \sum_{\mathcal{C}} \prod_{(\mathrm{i},\mathrm{j})} p\left(x_{(\mathrm{i},\mathrm{j})^+} \,\middle|\, \mathcal{D}_{(\mathrm{i},\mathrm{j})}, \mathcal{C}_{(\mathrm{i},\mathrm{j})}, \boldsymbol{b}\right) p\left(c_{(\mathrm{i},\mathrm{j})} \,\middle|\, \mathcal{D}_{(\mathrm{i},\mathrm{j})}, \mathcal{C}_{(\mathrm{i},\mathrm{j})^-}, \boldsymbol{b}\right) \\ = \;& H(x_{(\mathrm{i},\mathrm{j})}) \sum_{\mathcal{C}} \prod_{(\mathrm{i},\mathrm{j})} p\left(x_{(\mathrm{i},\mathrm{j})^+} \,\middle|\, \mathcal{D}_{(\mathrm{i},\mathrm{j})}, \mathcal{C}_{(\mathrm{i},\mathrm{j})}, \boldsymbol{b}\right) p\left(c_{(\mathrm{i},\mathrm{j})} \,\middle|\, \mathcal{D}_{(\mathrm{i},\mathrm{j})}, \mathcal{C}_{(\mathrm{i},\mathrm{j})^-}, \boldsymbol{b}\right) \end{aligned} \tag{5.5}$$

where $\mathcal{C}_{(1,1)^-} := \varnothing$ and $p\left(x_{(\mathrm{i},\mathrm{j})^+} \,\middle|\, \mathcal{D}_{(\mathrm{i},\mathrm{j})}, \mathcal{C}_{(\mathrm{i},\mathrm{j})}, \boldsymbol{b}\right) := 1$ if (i,j) is the index of the last observation. The form of equation (5.5) suggests that particle filtering [38] is the desired tool to produce an unbiased estimator efficiently. The particles are realizations of the latent collection of table configurations driven by the data, and are constructed sequentially by considering one observation at a time. Following previous notations, we denote the particles associated with data up until observation (i,j) by $\mathcal{C}^s_{(\mathrm{i},\mathrm{j})}$, $s = 1, \ldots, S$, where S is the total number of particles.

The particle filtering procedure starts with considering the first observation $(\mathrm{i}_1, \mathrm{j}_1)$. As there is no cluster existing in the tree, new clusters with the same label have to be created at the current node i_1 and all its ancestors. Hence, all particles are initialized as $\mathcal{C}^s_{(\mathrm{i}_1,\mathrm{j}_1)} = \left\{c_{(\mathrm{i}_1,\mathrm{j}_1)}\right\}$, $s = 1, \ldots, S$. At step (i,j), particle s is expanded from the previous step by including $c^s_{(\mathrm{i},\mathrm{j})}$, a simulated table configuration of $x_{(\mathrm{i},\mathrm{j})}$ from the conditional distribution $p\left(c_{(\mathrm{i},\mathrm{j})} \,\middle|\, x_{(\mathrm{i},\mathrm{j})}, \mathcal{D}_{(\mathrm{i},\mathrm{j})^-}, \mathcal{C}_{(\mathrm{i},\mathrm{j})^-} = \mathcal{C}^s_{(\mathrm{i},\mathrm{j})^-}, \boldsymbol{b}\right)$. This simulation is the same as a posterior update step described by Teh [89], and is easy to implement. After all particles are updated, an

importance sampling step is required to ensure unbiasedness. According to equation (5.5), the importance weight of particle s is

$$\begin{aligned} w^s_{(\mathrm{i,j})} &= p\left(x_{(\mathrm{i,j})^+} \,\middle|\, \mathcal{C}^s_{(\mathrm{i,j})}, \mathcal{D}_{(\mathrm{i,j})}, \boldsymbol{b}\right) \\ &= \sum_{c_{(\mathrm{i,j})^+}} p\left(x_{(\mathrm{i,j})^+} \,\middle|\, c_{(\mathrm{i,j})^+}, \mathcal{C}^s_{(\mathrm{i,j})}, \mathcal{D}_{(\mathrm{i,j})}\right) p\left(c_{(\mathrm{i,j})^+} \,\middle|\, \mathcal{C}^s_{(\mathrm{i,j})}, \mathcal{D}_{(\mathrm{i,j})}\right) \end{aligned} \tag{5.6}$$

Although a full bottom to top travel of the tree is required to compute the weights, $w^s_{(\mathrm{i,j})}$ is not hard to evaluate, as $p\left(x_{(\mathrm{i,j})^+} \,\middle|\, c_{(\mathrm{i,j})^+}, \mathcal{C}_{(\mathrm{i,j})}, \mathcal{D}_{(\mathrm{i,j})}\right) = 0$ when the table configuration of the succeeding observation $c_{(\mathrm{i,j})^+}$ does not agree with its label. After evaluating the importance weights, the particles are sampled with replacement according to the importance weights to form the collection of particles at step $(\mathrm{i,j})$. The weights at this step also gives an unbiased estimator of the partial likelihood $p\left(x_{(\mathrm{i,j})} \,\middle|\, \mathcal{D}_{(\mathrm{i,j})^-}, \boldsymbol{b}\right)$,

$$\hat{p}\left(x_{(\mathrm{i,j})} \,\middle|\, \mathcal{D}_{(\mathrm{i,j})^-}, \boldsymbol{b}\right) = \frac{1}{S}\sum_{s=1}^{S} w^s_{(\mathrm{i,j})}.$$

When all the observations have been considered, an unbiased estimator of the likelihood is simply the product of all the sequential estimators.

$$\hat{p}\left(\mathcal{D} \,|\, \boldsymbol{b}\right) = \prod_{(\mathrm{i,j})} \hat{p}\left(x_{(\mathrm{i,j})} \,\middle|\, \mathcal{D}_{(\mathrm{i,j})^-}, \boldsymbol{b}\right)$$

Algorithm 4 fully states the particle filtering algorithm.

2) Updating the jump rate

A conditional updating rule for the Poisson rate is required when it is unknown in the problem. Given the jumps $\boldsymbol{b}$, the posterior of λ is independent of the data and follows a Gamma distribution.

$$\lambda \,|\, \boldsymbol{b}, \mathcal{D} \sim \mathrm{Gamma}(1 + \sum_{\mathrm{i}=1}^{|\boldsymbol{b}|} b_{\mathrm{i}},\ \rho + |\boldsymbol{b}|) \tag{5.7}$$

5.2.1 Posterior estimation of the jumps

With posterior samples of the jump locations, we propose to use the Bayes factor to determine whether jumps actually occurred in the tree. The Bayes factor performs as an index of evidence when comparing alternative statistical models [90]. Let M_0 be the null model with no jumps and M_1 be the model with at least one jump. Then the Bayes Factor

$$K = \frac{p\left(\mathcal{D} \mid M_1\right)}{p\left(\mathcal{D} \mid M_0\right)} = \frac{p\left(M_1 \mid \mathcal{D}\right)}{p\left(M_0 \mid \mathcal{D}\right)} \Big/ \frac{p(M_1)}{p(M_0)} \,.$$

A larger Bayes factor suggests strong evidence towards M_1. As suggested by Jeffreys [91], $\log_{10} K < 1$ implies substantial to no evidence towards M_1, while $\log_{10} K \in [1, 2)$ indicates a strong evidence and $\log_{10} K \geq 2$ represents a decisive evidence supporting M_1.

All that is left is to construct an estimation of the jump locations. Examining one branch at a time to determine whether jumps occurred is not a proper approach, as jumps on different branches are heavily correlated with each other. Therefore, we propose to produce an estimation of the vector of jumps $\boldsymbol{b}$ directly. To do this, we reframe the problem into a clustering problem instead. Given jumps, each node in the simplified tree represents a cluster of tree nodes (see, for instance, figure 5.1(b)). A pair of nodes in the tree belong to the same cluster if there is no jump on the path connecting them. Considering the clustering introduced by posterior samples of jumps, we obtain an estimation by minimizing the posterior expectation of Binder's loss function under equal misclassification costs [58], [71]. This is a central estimate of the posterior clustering structure (a 'median' posterior clustering). The vector of jumps $\boldsymbol{b}$ suggested by the estimated clustering serves as our estimation of the jumps.

Algorithm 4: Particle Filtering

Function `ParticleFilter(`$\boldsymbol{b}$, S, $\mathcal{D}$, d, $\mathcal{T}$`):`

Input : Number of particles S, dataset $\mathcal{D} = \left\{x_{(\mathrm{i,j})}\right\}$, Pitman-Yor process discount parameter d, the phylogeny tree $\mathcal{T}$ and number of jumps on each branch $\boldsymbol{b}$.

Output: The estimated likelihood $\hat{p}\left(\mathcal{D} \mid \boldsymbol{b}\right)$.

for $s \leftarrow 1$ **to** S **do**
- $\mathcal{T}^s \leftarrow$ `TrimTree(`$\mathcal{T}, \boldsymbol{b}$`)` `// the simplifed tree`
- Equip an empty restaurant at each node of $\mathcal{T}^s$
- Initialize $\mathcal{T}^s$ with the first observation $x_{(1,1)}$
- $\mathcal{C}^s_{(1,1)} \leftarrow$ table configurations of $\mathcal{T}^s$

for $x_{(\mathrm{i,j})} \in \mathcal{D}$, $(\mathrm{i,j}) > (1,1)$ **do**
- **for** $s \leftarrow 1$ **to** S **do**
  - Sample $\tilde{c}^s_{(\mathrm{i,j})}$, seating arrangement for observation $x_{(\mathrm{i,j})}$
  - Calculate the weight of the sample, $\tilde{w}^s_{(\mathrm{i,j})}$, from equation (5.6)
  - $\tilde{\mathcal{C}}^s_{(\mathrm{i,j})} \leftarrow \tilde{c}^s_{(\mathrm{i,j})} \cup \mathcal{C}^s_{(\mathrm{i,j})^-}$
- $\left\{\mathcal{C}^s_{(\mathrm{i,j})}\right\} \leftarrow S$ samples from $\left\{\tilde{\mathcal{C}}^s_{(\mathrm{i,j})}\right\}$ with weights $\left\{\tilde{w}^s_{(\mathrm{i,j})}\right\}$
- $w^s_{(\mathrm{i,j})} \leftarrow$ the original resampling weight of $\mathcal{C}^s_{(\mathrm{i,j})}$, $s = 1, \ldots, S$

return $\hat{p}\left(\mathcal{D} \mid \boldsymbol{b}\right) = \prod_{(\mathrm{i,j})} \left[\frac{1}{S} \sum_{s=1}^{S} w^s_{(\mathrm{i,j})}\right]$

Algorithm 5: Particle Markov Chain Monte Carlo

Function `ParticleMCMC(`S, $\mathcal{D}$, d, $\mathcal{T}$, M`):`

Input : Number of particles S, dataset $\mathcal{D} = \left\{x_{(\mathrm{i,j})}\right\}$, Pitman-Yor process discount parameter d and the phylogeny tree $\mathcal{T}$, number of iterations M.

Output: A chain of posterior samples of the jump rate λ and jump locations $\boldsymbol{b}$.

Initialize $\lambda \leftarrow$ sample from $\exp\left(L^{-1}\right)$

Given λ, initialize $\boldsymbol{b} \leftarrow$ sample from Poisson $\left(\lambda L_{\mathrm{i}}\right)$, $\mathrm{i} = 1, \ldots, B$

Estimate the likelihood $\mathcal{L} \leftarrow$ `ParticleFilter(`$\boldsymbol{b}, S, \mathcal{D}, d, \mathcal{T}$`)`

for $l \leftarrow 1$ **to** M **do** `// Metropolis-Hastings within Gibbs`
- Given $\boldsymbol{b}$, $\lambda \leftarrow$ sample from equation (5.7)
- Given λ, propose $\boldsymbol{b}^*$ from $q\left(\boldsymbol{b}^* \mid \boldsymbol{b}\right)$
- Estimate the likelihood $\mathcal{L}^* \leftarrow$ `ParticleFilter(`$\boldsymbol{b}^*, S, \mathcal{D}, d, \mathcal{T}$`)`
- Compute $A(\boldsymbol{b}, \boldsymbol{b}^*) = \frac{\mathcal{L}^*}{\mathcal{L}} \cdot \frac{p(\boldsymbol{b}^* \mid \lambda) q(\boldsymbol{b} \mid \boldsymbol{b}^*)}{p(\boldsymbol{b} \mid \lambda) q(\boldsymbol{b}^* \mid \boldsymbol{b})}$
- With probability $\min\{1, A(\boldsymbol{b}, \boldsymbol{b}^*)\}$, $\boldsymbol{b} \leftarrow \boldsymbol{b}^*$ and $\mathcal{L} \leftarrow \mathcal{L}^*$.

return *posterior samples of* λ *and* $\boldsymbol{b}$

6. EMPIRICAL RESULTS OF PHYLOGENETIC HPYP MODEL

In this chapter, we carry out several several synthetic and real data studies to evaluate our model and algorithm in chapter 5. Our main comparison is the `treeBreaker` approach proposed by Ansari and Didelot [37]. Our algorithm is implemented in Python3 and a `C++` implementation of `treeBreaker` is publicly available in Github[1]. Most of the experiments in this chapter focus on the situation of a binary dataset with exactly one observation at each leaf node, as this is the only case `treeBreaker` can handle. However, our implementation can be used in more complicated situations, such as when the data is not binary, when observations exist in internal nodes or when multiple observations are obtained at a node. We demonstrate the efficacy of our approach on one of the above situations in the last real data analysis where we detect changes of distribution in a non-binary post-marital residence data.

For all experiments, the tree is first rescaled as described in section 5.1 to allow us to keep the intensity of jumps constant over time. To be specific, every point on the tree is associated with a time: the distance from the root. If there are k points in the tree that share the same time tag (i.e. k branches), the jump rate at each of them needs to be scaled down by a factor of k in order to have a constant jump intensity over time. Adjusting the intensity is equivalent to scaling branch lengths, especially as we are only interested in the number of jumps at a branch, not the exact locations of the jumps. The collection of starting and ending times of branches in the tree partition the time interval from the root to the furthest leaf node into segments. At any time within a segment, there are the same amount of branches of the tree associated with it. Rescaling a segment associated with k fragments of branches is equivalent to shrunk all associated branch fragments in the tree by a factor of k. Having a Poisson process with constant intensity on the rescaled tree is equivalent to having an inhomogeneous Poisson process on the original tree such that the intensity over time is constant.

[1] ↑https://github.com/ansariazim/treeBreaker

Unless otherwise specified, we place an exponential prior on the Poisson rate (jump rate) λ of our model such that the prior mean rate results in having on average one jump in the rescaled tree. For most of the experiments, we set the discount parameter of our model to be 0.5, as we find that the choice of the discount parameter does not have a significant impact on the result (see section 6.1.1). In general, we do not recommend using a too large discount parameter, as it models very small changes in the distribution. It could introduce redundant jumps and lead to identifiability issues. We set the base measure H to be a multinomial distribution with equal probability to take all possible values. For all experiments, we run 50,000 MCMC iteration of our methods with the first half discarded as burn-in. We include the total CPU runtime and effective sample size (ESS) of jump rate of our algorithm in the result of real-data studies to assess the MCMC mixing. ESS is computed with the `effectiveSize` function from the `R` package `coda` [70] and is an estimation of the number of uncorrelated samples corresponding to the MCMC samples. Dividing it by the total CPU runtime of the algorithm yields ESS per second (ESS/s) for the time efficiency of producing independent samples.

6.1 Synthetic studies

We designed three synthetic studies to evaluate performance when a ground truth is known. An essential variable for this assessment is the amount of change in the distribution introduced by a jump, which we will refer to as "jump size". The jump size is quantified by the total variation (TV) and empirical total variation (EmpTV) between the two clusters of nodes before and after a jump. To be specific, for a jump (the ground truth) existing in the tree, the size of it is described by the total variation between the distributions before and after it. With data simulated from the two clusters, we can also obtain the empirical total variation between the two sets of observations. The empirical measure is important, as it reflects the variability in the simulating the data.

The first synthetic study compares different choices of the discount parameter, and examines the identifiability of the target jump with various jump sizes. The second synthetic

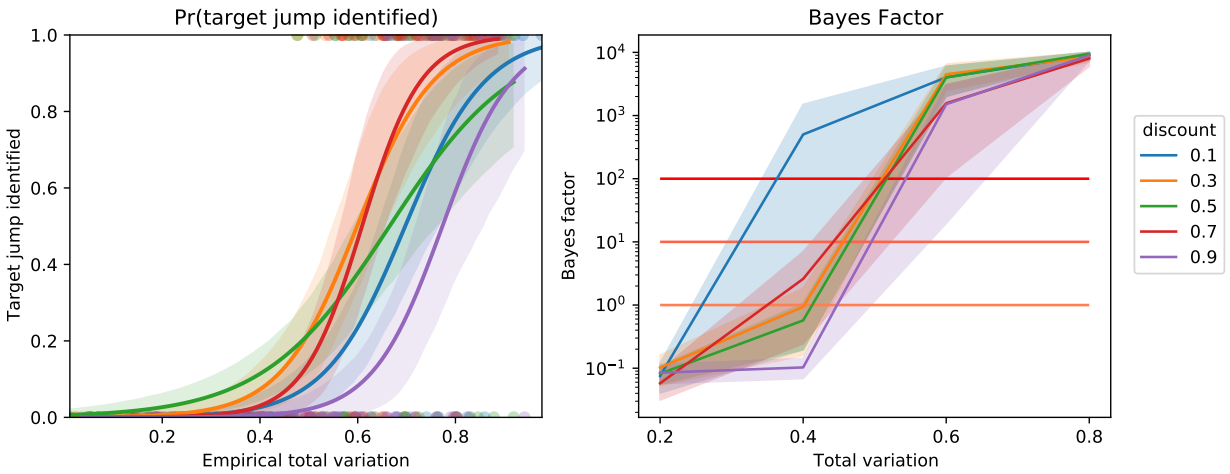

Figure 6.1. Synthetic study results of section 6.1.1. **(left)** Estimated probability of identifying the target jump (and the 95% confidence band) versus the empirical total variation. The probability is estimated by fitting a logistic regression to he indicator of identifying the target jump. **(right)** Bayes factor (mean and the 95% percentile band) versus the total variation. The Bayes factor is truncated at 10^4 to make the plot. The horizontal lines marks the conventional decision boundaries of Bayes factor described in section 5.2.1.

study focus on the robustness of our model to misspecification of the jump rate λ and the third synthetic study aims to compare our approach with `treeBreaker`.

6.1.1 Identifiability of jumps

This study is a sanity check of our model, and also investigates the effect of the discount parameter. For this experiment, we randomly generated a binary tree with 100 leaf nodes using the function `rtree` in `R` package `ape` [92]. For each replication, we limited ourselves to branches that have 10% to 50% leaves in the subtree below, and uniformly assigned a jump to one of these branches. Since we are considering the case with only one observation at each leaf node, the size requirement of the subtree guarantees that sufficient observations are affected by the jump. Figure 6.3 (left) shows a tree used in this study and a branch randomly selected to have a jump. All that is left is to generate data is to assign two Bernoulli distributions, one before before and one after the jump. Let p denote the total variation distance between these two, and we use this to set the two distributions to be

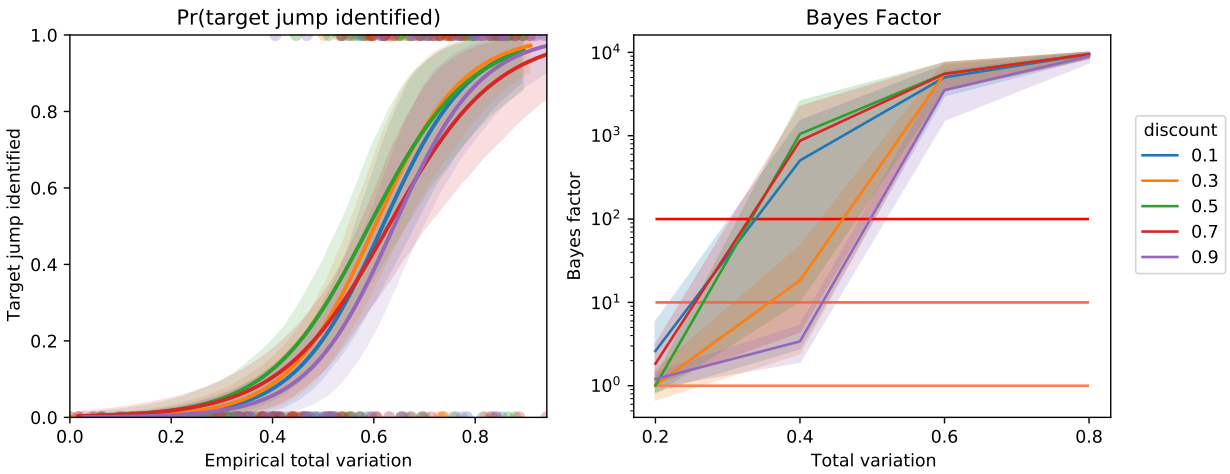

Figure 6.2. Synthetic study results of section 6.1.1 when the jump rate is known. **(left)** Estimated probability of identifying the target jump (and the 95% confidence band) versus the empirical total variation. The probability is estimated by fitting a logistic regression to he indicator of identifying the target jump. **(right)** Bayes factor (mean and the 95% percentile band) versus the total variation. The Bayes factor is truncated at 10^4 to make the plot. The horizontal lines marks the conventional decision boundaries of Bayes factor described in section 5.2.1.

symmetric about 0.5. To be specific, one of the distribution has $(0.5 + p)$ probability to be 1, whereas the other one takes 1 with $(0.5 - p)$ probability. The total variation p takes values of 0.2, 0.4, 0.6 and 0.8 in this study. We also vary the discount parameter d, setting d to values in $\{0.1, 0.3, 0.5, 0.7, 0.9\}$. For each setting of the total variation and discount parameter, we ran 20 replications, assigning a jump to a branch as described earlier, and generating corresponding datasets from the designed distributions.

We evaluate performance using two quantities, the Bayes factor and an indicator of whether the branch with jump is correctly identified. The Bayes factor compares the model with at least one jump to the null model with no jump (see section 5.2.1 for details) and shows how confident we are of the existence of jumps. Following the conventional decision rule (see section 5.2 for details), a Bayes factor greater than 10^2 suggests decisive evidence towards the existence of jumps in the tree. After obtaining the posterior "median" estimation of the jumps as described in section 5.2.1, the indicator, which we refer to as "target jump identified", takes value one when our posterior estimation of jumps matches exactly with the

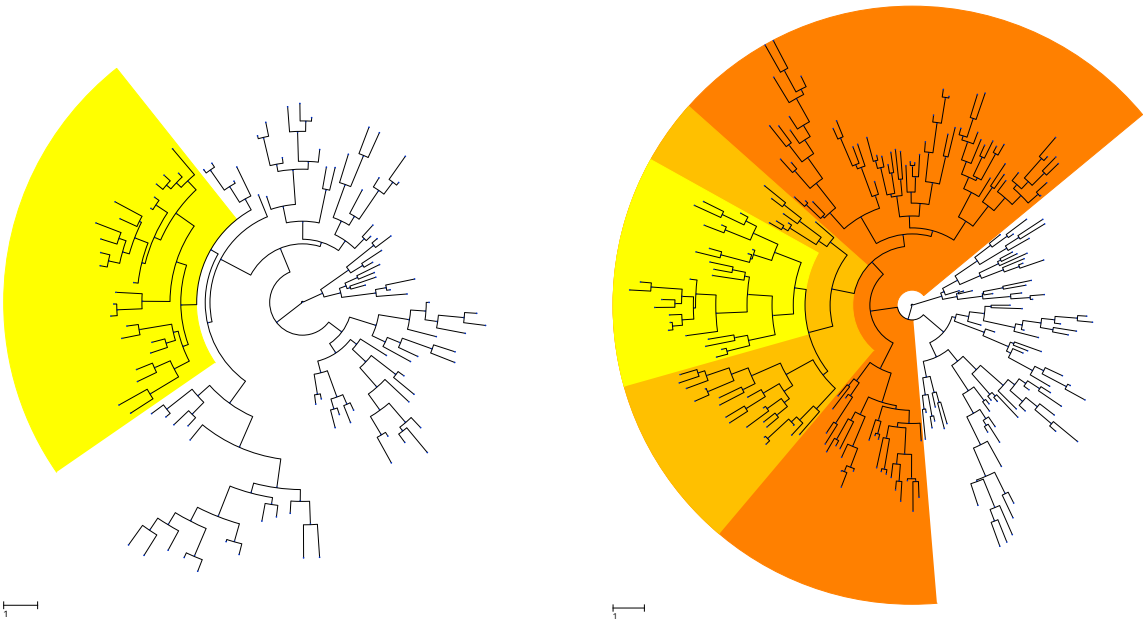

Figure 6.3. The simulated tree for **(left)** the first two synthetic studies in section 6.1.1 and 6.1.2, and **(right)** the third synthetic study in section 6.1.3. Color shading highlights the subtrees affected by branches with jumps.

ground truth, i.e. when the target branch and only the target branch is identified by our algorithm. Then, so see how accurately jumps are identified as a function of total variation, for all replications, we fit a logistic regression model on the indicator with the empirical total variation as the covariate variable.

Figure 6.1 summarizes the result. The left panel plots the estimated probability of identifying the target jump versus the empirical total variation, and the right panel plots the Bayes factor (truncated at 10^4) versus the total variation. When the total variation is small (0.2 and 0.4), there is little evidence in the data to support the existence of a jump, and thus, as we expected, both the Bayes factor and the probability of identifying the target is small. When the data strongly suggests a jump (TV = 0.8), it is captured by the Bayes factor, and the probability of detecting the jump is also close to 1.

Different choices of the discount parameter do not strongly impact the decision to determine the existence of jumps, but a very large discount parameter slightly harms the ability of correctly locating the jump in the tree. In the left panel, when the empirical total variation is large, the probability of identifying the target jump when the discount is 0.9 is consistently

below other choices of d. This is as expected, as with a high discount parameter, the model generates jumps with smaller sizes, and hence, besides the target branch, our algorithm tends to also detect false positives. Therefore, in practice, we recommend the user to choose a moderate discount parameter, and in the rest of experiments, we fix it to 0.5.

We further evaluate our algorithm when the jump rate is fixed to have on average one jump in the tree, $\lambda = 1/L$. The results are shown in figure 6.2. Similar to the case when the jump rate is learnt, our algorithm performs well when there is a large change in the distribution before and after the jump. Interestingly, the effect of the discount parameter vanishes in this case, as fixing the jump rate discourages introducing redundant jumps in the model.

6.1.2 Robustness to misspecification of jump rate

In this study, we seek to evaluate how sensitive our approach is to misspecifications in the jump rate. We used similar settings as in the previous study (section 6.1.1), except here, we fix the discount parameter to be 0.5. We investigated the performance of our model when the prior mean jump rate corresponds to on average 0.5, 1, 2, 5, or 10 jumps in the tree, whereas the ground truth corresponded to exactly one branch with jumps.

The results are shown in figure 6.4. The left panel shows that the choices of prior number of jumps does not have a significant impact on the probability of identifying the target branch, as the estimated probabilities with different prior number of jumps are consistent with each other. In other words, the prior of the jumps rate has little impact to inference of the jump locations. This result shows that when learning the jump rate, our model is very robust to potential misspecification of the prior on jump rate.

We also investigate performance when the jump rate is fixed to have also on average 0.5, 1, 2, 5, or 10 number of jumps in the tree, and the result is shown in figure 6.5. In this case, fixing the jump rate in the prior has a stronger impact on the result compared to the previous case. The right panel of figure 6.5 shows that with a high prior number of jumps (5 or 10), the Bayes factor stays very high even when the data provides weak support to the

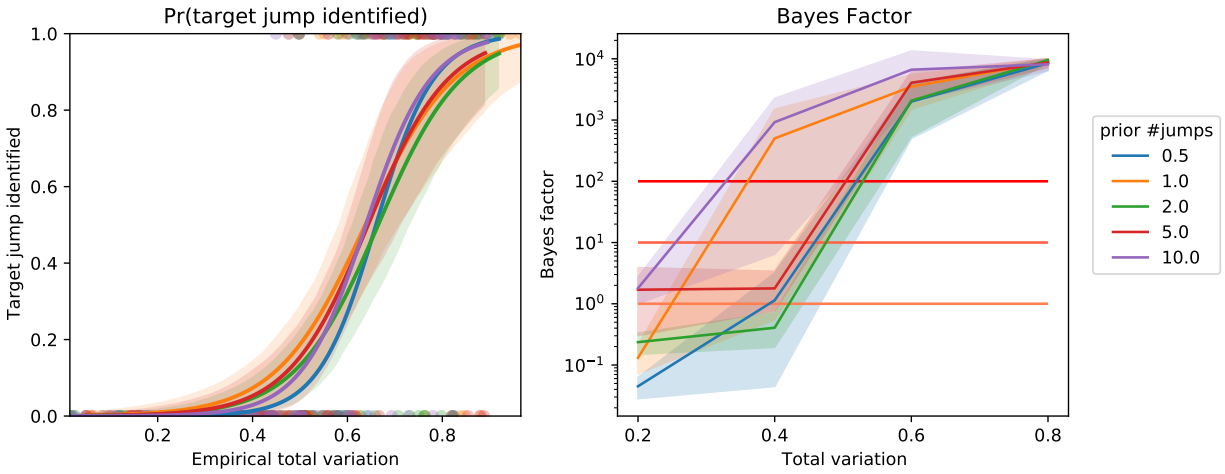

Figure 6.4. Synthetic study results of section 6.1.2. **(left)** Estimated probability of identifying the target jump (and the 95% confidence band) versus the empirical total variation. The probability is estimated by fitting a logistic regression to he indicator of identifying the target jump. **(right)** Bayes factor (mean and the 95 percentile band) versus the total variation. The Bayes factor is truncated at 10^4 to make the plot. The horizontal lines marks the conventional decision boundaries of Bayes factor described in section 5.2.1.

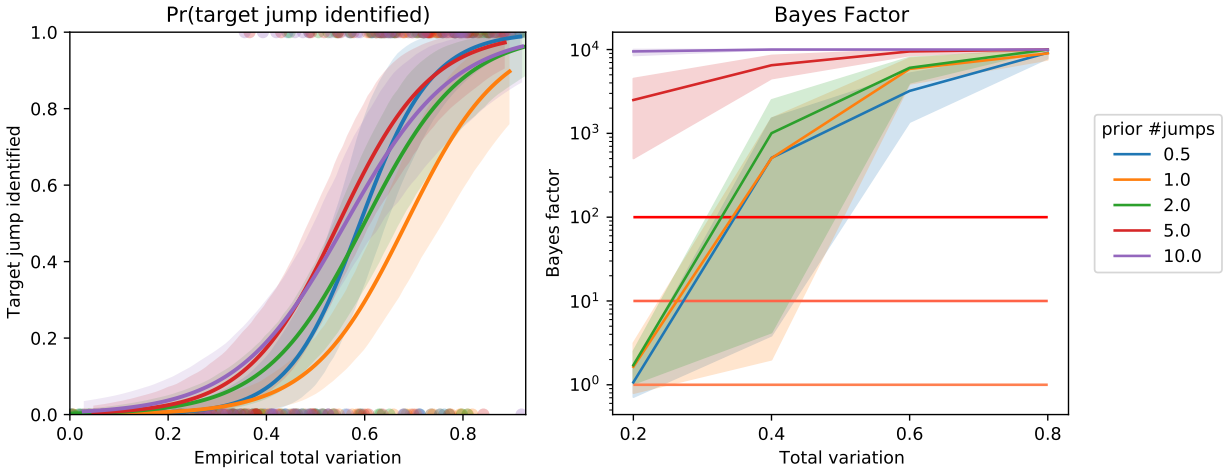

Figure 6.5. Synthetic study results of section 6.1.2 when the jump rate is known. **(left)** Estimated probability of identifying the target jump (and the 95% confidence band) versus the empirical total variation. The probability is estimated by fitting a logistic regression to he indicator of identifying the target jump. **(right)** Bayes factor (mean and the 95 percentile band) versus the total variation. The Bayes factor is truncated at 10^4 to make the plot. The horizontal lines marks the conventional decision boundaries of Bayes factor described in section 5.2.1.

existence of jumps. Comparing the two sets of results, we recommend that users infer the jump rate when there is no strong prior knowledge of the number of jumps in the tree.

6.1.3 Comparison with `treeBreaker` [37]

This study compares the performance of `treeBreaker` with our algorithm. Recall that the main difference between the two approaches is that `treeBreaker` assumes independence between the distributions before and after a jump while our approach captures the similarity between them by centering the new one on the old one. As a consequence, we expect that for trees with multiple moderate-sized jumps, our model can better leverage the dependency structure between the distributions, and hence outperform `treeBreaker`. The advantages of our approach are clearest with moderate sized jumps because when the jump size is too small, both approaches will not be able to detect any jumps, and when it is too large, one could be able to identify the jumps even without considering the dependency structure. Given this intuition, we design our study as follows. We simulated a binary tree with 200 leaves in the same way as the first two synthetic studies, and selected 3 branches to have jumps nested with each other. figure 6.3 (right) shows the simulated tree and the locations of those jumps. The branches are selected to have about 75%, 50% and 25% leaves in the subtrees, so that the four clusters (with different color shadings in figure 6.3 (right)) of nodes have roughly the same number of observations. The observation distributions of these four clusters are designed to reflect the nested structure of the jumps, with all the three jumps in the "same direction" and of the same size. To be specific, let p denote the total variation between the distributions before and after a jump. The Bernoulli distribution at the root takes 1 with probability $(0.5-1.5p)$, whereas the following ones have probability $(0.5-0.5p)$, $(0.5+0.5p)$, $(0.5+1.5p)$ respectively. With this design, the total variation takes values between 0 and $1/3$, and we simulate 100 datasets for p set to $\{0.05, 0.15, 0.25\}$ each.

To compare the performance with `treeBreaker`, we report the average receiver operating characteristic (ROC) curves and area under curve (AUC) for both approaches in figure 6.6. The ROC curve plots the true positive rate against the false positive rate, and AUC measures the area under a ROC curve. The diagonal line (dotted lines in figure 6.6) represents

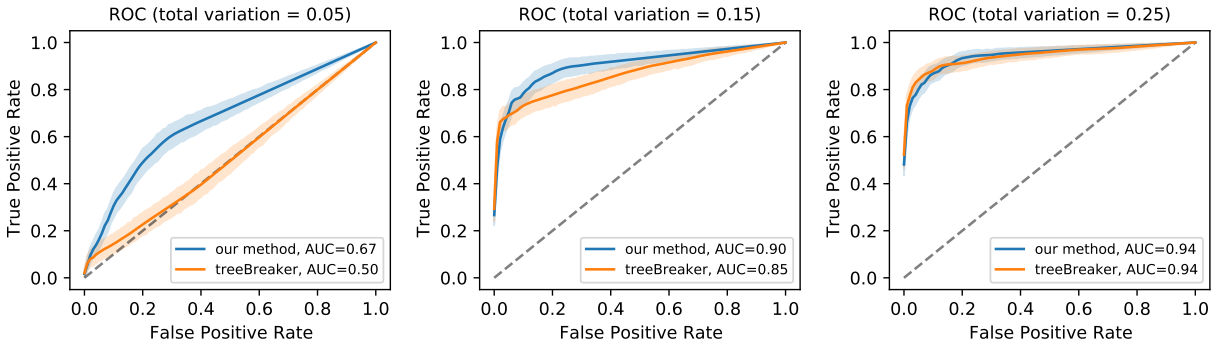

Figure 6.6. Average ROC curve (with 95 percentile band) and AUC results of section 6.1.3.

random guessing with AUC = 0.5. The rule of thumb is that the more a ROC curve is bending towards the top left corner, the better the performance is, and a high AUC also implies a sound performance of the algorithm. In this study, every run of our algorithm and `treeBreaker` infers a set of branches with jumps, consisting of the true positives and false positives. Aggregating all the results gives the average ROC curve and the corresponding AUC shown in figure 6.6. As we expected, when the jump size is relatively small, our approach produces a higher AUC than `treeBreaker` and performs better. When the total variation is 0.05, the `treeBreaker` has an average AUC of 0.5 with the ROC curve almost lie exactly on the diagonal line, which suggests that it cannot differentiate branches with jumps from the rest of branches in the tree. Our model performs much better under this situation, as we obtained an average AUC of 0.67. This case is the most difficult one, as the small total variation really cannot provide much evidence to support the existence of jumps. The reason why our approach behaves better in this case is that our model utilizes the shared statistical strength in the dependency structure between the populations, which gives us additional power in detecting jumps. When the jump size increases to (TV=0.15), the difference in AUC of the two models reduces to 0.05, and the ROC curves lie much closer to each other. Although our model still performs better in this case, the edge is not as clear as in the case with a smaller jump size. When the jump size is relatively large (TV = 0.25), our ROC and AUC agree with those of `treeBreaker`, which suggests that our algorithm has a comparable performance with `treeBreaker`.

6.2 Real Data Analysis

6.2.1 Detecting cytotoxic T-lymphocytes (CTLs) escape mutations in HIV

This section aims to demonstrate that our approach is able to produce similar results with `treeBreaker` on a real world problem. Human leukocyte antigen (HLA) type I genes are very important to the human immune system, encoding proteins on the surface of human cells which bring epitopes (segments of viral proteins) to the surface when a cell is infected by a virus [93]. Thanks to this functionality, cytotoxic T lymphocytes (CTLs), also known as T-cells, can identify the infected cells by recognizing the epitopes and destroy them. Therefore, HLA-driven mutations of the virus that result in weak binding of epitopes with HLA-encoded proteins can lead to virus escaping the immune response of the host.

In the work of Ansari and Didelot [37], `treeBreaker` is applied to the problem of detecting HLA-driven evolution of HIV to determine whether host HLA alleles are randomly distributed on the tips of the virus phylogenetic tree or whether there are clades where the distributions are distinct from each other. The dataset used is from a cohort with 261 subjects (leaf nodes in the tree) published by Carlson, Brumme, Rousseau, *et al.* [94]. The whole genome of the viruses are aligned and divided into 10 segments of 1000 nucleotides, and a phylogenetic tree are inferred from each of these alignments. This results in 10 different trees to describe the dependency structure of these 261 populations. The distribution of alleles is of interest. To be specific, whether an allele of HLA presents or not forms a binary distribution and observations are available on the leaf nodes of every estimated phylogenetic tree. A number of alleles are studied by Ansari and Didelot [37] to detect subgroups in the cohort with a distinct distribution of alleles, and a jump associated to the distribution of HLA allele B57 in the first phylogenetic tree is identified.

The dataset is available online[2]. Although it slightly differs from the one described in Ansari and Didelot [37], the difference is too small to affect the final result. We ran our algorithm on this dataset and found that the existence of jumps is strongly supported, as the Bayes factor is estimated to be $+\infty$. Figure 6.7 shows the branch with jumps we detected. This finding agrees with the result in Ansari and Didelot [37]; both methods identify the

[2]↑obtained from https://www.hiv.lanl.gov/content/immunology/hlatem/study5/index.html

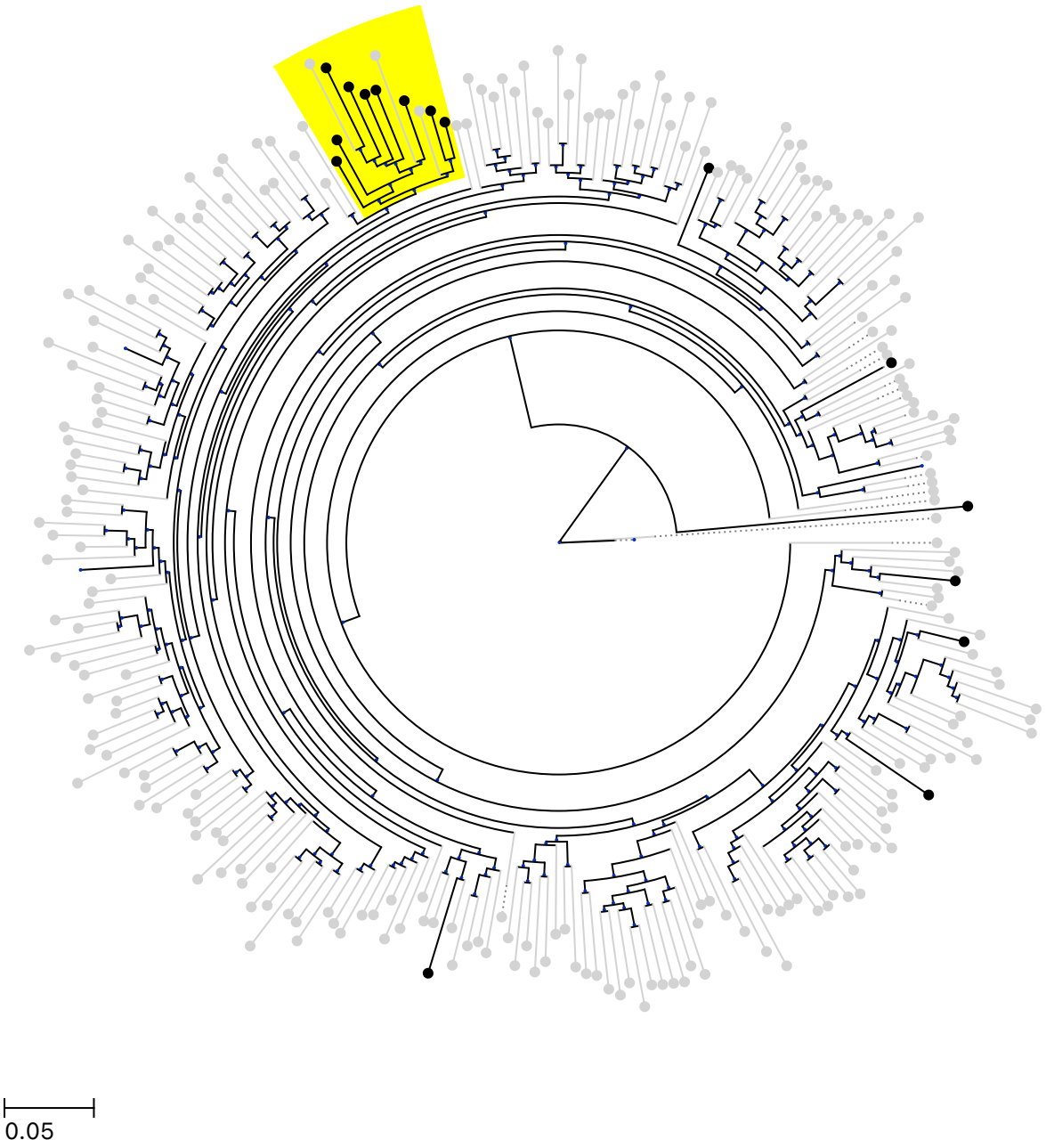

Figure 6.7. The data and result of the real data study to detect human leukocyte antigen (HLA)-driven evolution of HIV (section 6.2.1). Dots at leaf nodes represent whether allele B57 exists (black) or not (light grey) in the subject. The Bayes factor we obtained is $+\infty$ suggesting strong evidence towards having jumps in the tree. The color shaded area marks the jump we detected using our algorithm, which is consistent with the findings of Ansari and Didelot [37]. The total runtime of our algorithm is 2098.8s, and it produces ESS/s = 9667.

same clade where 9 out of the 12 hosts have the B57 allele (10 out of 12 in Ansari and Didelot [37] as their data slightly differs from the one we use), which is a much higher proportion compared to the rest of the tree, where only 7 hosts have allele B57.

6.2.2 Detecting changes in post-marital residence patterns

Unlike `treeBreaker`, our implementation also works on more complex scenarios, including the case where the data is not binary. In this study, we apply our approach to detect changepoints in the distribution of post-marital residence patterns within the Uto-Aztecan language family, a dataset where observations take one of four values. The measurements correspond to where newly-wed couple might live after marriage: with the family of the husband (patrilocality), of the wife (matrilocality), of either the husband or the wife (ambilocality) or a new residence separated from their families (neolocality). Moravec, Atkinson, Bowern, *et al.* [95] studied the post-marital residence patterns in five different language families. We note that their work focused on how the post-marital residence state transits on individuals, whereas our model focuses on the distributional change in the populations, and therefore, cannot directly compare the two approaches.

We requested the dataset directly from the authors and only explore the Uto-Aztecan language family, the smallest among the five. This language family forms a language tree with node representing language communities. There are 26 communities at leaf of the tree, and the primary social norm of post-marital residence is obtained for each of them. Running our algorithm on this dataset produces a Bayes factor of 908.76 which strongly suggests changes of distributions happening in the tree. In total, we locate two branches with jumps in the tree, and figure 6.8 shows the data and our results. Professor Murray Cox, a domain expert, helped us gain insight into the result. For the cluster at the root (above both jumps), new couples in the population prefer to reside with the husband (patrilocality). The first jump at the top portion of the tree creates a cluster at the subtree of Guarijio and Tarahumara, where new couples tend to seek new residence (neolocality). These two languages are spoken in a region with quite poor soil, so newly married couples have to move to a new location to find productive farming land. The second jump at the bottom portion of the tree instead

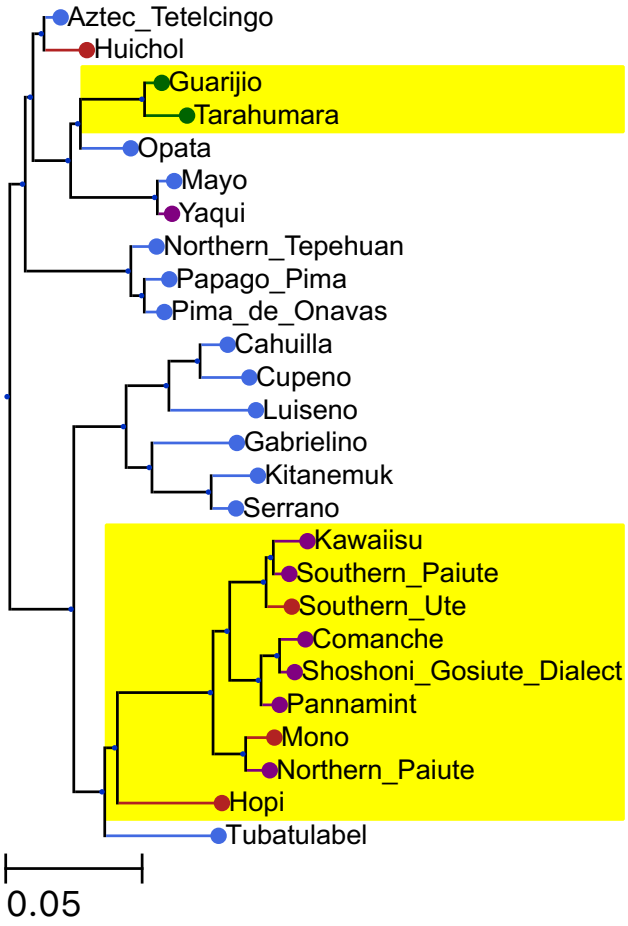

Figure 6.8. The data and result of the real data study to detect changes in post-marital residence patterns (section 6.2.2). Dots at the leaf nodes represent the data where the four categories are patrilocality (blue), matrilocality (red), ambilocality (purple) and neolocality (green). We obtain a Bayes factor of 908.76 and locate two branches with jumps as shown with color shading in the tree. The runtime of our algorithm is 208.8s, and it produces ESS/s = 10.

switches towards a distribution that strongly favors matrilocality and ambilocality. This is probably due to the change of practices result from transitioning into more desert-plains environment.

7. SUMMARY AND FUTURE WORK

7.1 Summary

This dissertation studies Bayesian nonparametric clustering under structural restrictions in two distinct problems. In the first part, we proposed a Matérn repulsive mixture model (MRMM), a novel approach to repulsive mixture modeling through the Matérn type-III repulsive point process. The structural restriction of this problem – repulsion between clusters – encourages interpretability and makes the inference robust to model misspecification. We derive a novel, simple and efficient MCMC sampling algorithm and evaluate performance on a number of synthetic and real datasets.

In the second part, we focus on the problem of clustering populations with a hierarchical dependency structure described by a tree. This structural restriction embodies prior domain knowledge and enables statistical sharing through the underlying structure of the problem. We perform clustering in this case by introducing "jumps" as locations in the tree where the distribution changes significantly. For this problem, we build a novel nonparametric Bayesian framework based on hierarchical Pitman-Yor processes and Poisson processes, and developed an efficient particle MCMC algorithm to handle the problem. The efficacy of our approach is demonstrated through various synthetic and real data analyses.

7.2 Future Work

For the first topic, there are a number of open avenues for future investigation. One involves introducing Matérn repulsive mechanisms into more general latent variable models such as latent feature models. Another class of models are time series models such as self-avoiding Markov models. Even restricting ourselves to mixture models, problems arise when working with high-dimensional parameter spaces. Possible approaches include projecting down to a lower-dimensional space before carrying out Matérn thinning. From a theoretical viewpoint, it is of interest to investigate asymptotic consistency of this class of repulsive mixture models, and also quantify rates of posterior convergence. This will follow much along the lines of Xie and Xu [60], though the specifics of the Matérn repulsion will require

some care. It is also of interest to quantify the rate of MCMC mixing, and conditions under which it exhibits desirable properties like geometric ergodicity.

For the second topic, there are also a few potential future directions. A natural extension to develop our model to work with continuous distributions. Carrying this out is relatively straightforward, and requires framing each node density G_{i} as a mixture model specified by a Pitman-Yor prior. Another direction is to extend the dependency structure from a tree to a network so that it finds a much wider applications in real-world problems. As for the theoretical direction, we are interested in the asymptotic behavior of the model, as well as assessing the mixing of the MCMC. Also, finding new applications of our approach, especially beyond the scope of evolution trees, can be very exciting.

REFERENCES

[1] J. H. Ward Jr, "Hierarchical grouping to optimize an objective function," *Journal of the American Statistical Association*, vol. 58, no. 301, pp. 236–244, 1963.

[2] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, Oakland, CA, USA, vol. 1, 1967, pp. 281–297.

[3] J. A. Hartigan, *Clustering algorithms*. John Wiley & Sons, Inc., 1975.

[4] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: A review," *ACM Computing Surveys (CSUR)*, vol. 31, no. 3, pp. 264–323, 1999.

[5] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet allocation," *Journal of Machine Learning Research*, vol. 3, no. Jan, pp. 993–1022, 2003.

[6] D. M. Blei and J. D. Lafferty, "Topic models," in *Text mining*, Chapman and Hall/CRC, 2009, pp. 101–124.

[7] M. Pagel and A. Meade, "A phylogenetic mixture model for detecting pattern-heterogeneity in gene sequence or character-state data," *Systematic Biology*, vol. 53, no. 4, pp. 571–581, 2004.

[8] G. J. McLachlan, R. W. Bean, and D. Peel, "A mixture model-based approach to the clustering of microarray expression data," *Bioinformatics*, vol. 18, no. 3, pp. 413–422, 2002.

[9] S. T. Jensen and J. S. Liu, "Bayesian clustering of transcription factor binding motifs," *Journal of the American Statistical Association*, vol. 103, no. 481, pp. 188–200, 2008.

[10] N. Friedman and S. Russell, "Image segmentation in video sequences: A probabilistic approach," in *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence*, ser. UAI'97, Providence, Rhode Island, 1997, pp. 175–181, ISBN: 1558604855.

[11] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern classification and scene analysis*. Wiley New York, 1973, vol. 3.

[12] E. W. Forgy, "Cluster analysis of multivariate data: Efficiency versus interpretability of classifications," *Biometrics*, vol. 21, pp. 768–769, 1965.

[13] J. H. Wolfe, "Object cluster analysis of social areas," Ph.D. dissertation, University of California, 1963.

[14] J. D. Banfield and A. E. Raftery, "Model-based Gaussian and non-Gaussian clustering," *Biometrics*, pp. 803–821, 1993.

[15] J. A. Blimes *et al.*, "A gentle tutorial of the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models," *International Computer Science Institute*, vol. 4, no. 510, p. 126, 1998.

[16] G. J. McLachlan and K. E. Basford, *Mixture Models: Inference and Applications to Clustering*. M. Dekker New York, 1988, vol. 38.

[17] A. Dasgupta and A. E. Raftery, "Detecting features in spatial point processes with clutter via model-based clustering," *Journal of the American Statistical Association*, vol. 93, no. 441, pp. 294–302, 1998.

[18] C. Fraley, "Algorithms for model-based gaussian hierarchical clustering," *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 270–281, 1998.

[19] P. D. McNicholas, "Model-based clustering," *Journal of Classification*, vol. 33, no. 3, pp. 331–373, 2016.

[20] K. Y. Yeung, C. Fraley, A. Murua, A. E. Raftery, and W. L. Ruzzo, "Model-based clustering and data transformations for gene expression data," *Bioinformatics*, vol. 17, no. 10, pp. 977–987, 2001.

[21] C. Fraley and A. E. Raftery, "Model-based clustering, discriminant analysis, and density estimation," *Journal of the American Statistical Association*, vol. 97, no. 458, pp. 611–631, 2002.

[22] V. Melnykov, R. Maitra, *et al.*, "Finite mixture models and model-based clustering," *Statistics Surveys*, vol. 4, pp. 80–116, 2010.

[23] J. Rousseau and K. Mengersen, "Asymptotic behaviour of the posterior distribution in overfitted mixture models," *Journal of the Royal Statistical Society - Series B*, vol. 73, no. 5, pp. 689–710, 2011.

[24] J. W. Miller and M. T. Harrison, "A simple example of dirichlet process mixture inconsistency for the number of components," in *Advances in Neural Information Processing Systems*, 2013, pp. 199–206.

[25] J. W. Miller and M. T. Harrison, "Inconsistency of pitman-yor process mixtures for the number of components," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 3333–3370, 2014.

[26] T. S. Ferguson, "A bayesian analysis of some nonparametric problems," *The Annals of Statistics*, pp. 209–230, 1973.

[27] C. E. Antoniak, "Mixtures of dirichlet processes with applications to bayesian nonparametric problems," *The Annals of Statistics*, pp. 1152–1174, 1974.

[28] J. Pitman and M. Yor, "The two-parameter poisson-dirichlet distribution derived from a stable subordinator," *The Annals of Probability*, pp. 855–900, 1997.

[29] J. Pitman *et al.*, "Combinatorial stochastic processes," Technical Report 621, Dept. Statistics, UC Berkeley, 2002., Tech. Rep., 2002.

[30] V. Rao, R. P. Adams, and D. D. Dunson, "Bayesian inference for Matérn repulsive processes," *Journal of the Royal Statistical Society - Series B*, vol. 79, no. 3, pp. 877–897, 2017.

[31] B. Matérn, "Spatial variation: Stochastic models and their application to some problems in forest surveys and other sampling investigations," *Meddelanden fran Statens Skogsforskningsinstitut*, vol. 49, p. 144, 1960.

[32] B. Matérn, *Spatial variation*. Springer Science & Business Media, 2013, vol. 36.

[33] J. Kingman, *Poisson Processes*, ser. Oxford Studies in Probability. Clarendon Press, 1992, ISBN: 9780191591242.

[34] H. Sun, B. Zhang, and V. Rao, "Bayesian repulsive mixture modeling with mat ern point processes," Department of Statistics, Purdue University, Tech. Rep., 2021.

[35] F. Nielsen, R. Nock, and S.-i. Amari, "On clustering histograms with k-means by using mixed α-divergences," *Entropy*, vol. 16, no. 6, pp. 3273–3301, 2014.

[36] K. Henderson, B. Gallagher, and T. Eliassi-Rad, "Ep-means: An efficient nonparametric clustering of empirical probability distributions," in *Proceedings of the 30th Annual ACM Symposium on Applied Computing*, 2015, pp. 893–900.

[37] M. A. Ansari and X. Didelot, "Bayesian inference of the evolution of a phenotype distribution on a phylogenetic tree," *Genetics*, vol. 204, no. 1, pp. 89–98, 2016.

[38] C. Andrieu, A. Doucet, and R. Holenstein, "Particle markov chain monte carlo methods," *Journal of the Royal Statistical Society - Series B*, vol. 72, no. 3, pp. 269–342, 2010.

[39] S. Ghosal, "The dirichlet process, related priors and posterior asymptotics," *Bayesian Nonparametrics*, vol. 28, p. 35, 2010.

[40] H. Bensmail, G. Celeux, A. E. Raftery, and C. P. Robert, "Inference in model-based cluster analysis," *Statistics and Computing*, vol. 7, no. 1, pp. 1–10, 1997.

[41] G. J. McLachlan, S. X. Lee, and S. I. Rathnayake, "Finite mixture models," *Annual Review of Statistics and its Application*, vol. 6, pp. 355–378, 2019.

[42] L. Devroye, *Non-uniform random variate generation*, 1986.

[43] J. Kingman, "Completely random measures," *Pacific Journal of Mathematics*, vol. 21, no. 1, pp. 59–78, 1967.

[44] M. Lomelı, S. Favaro, and Y. W. Teh, "A marginal sampler for σ-stable poisson–kingman mixture models," *Journal of Computational and Graphical Statistics*, vol. 26, no. 1, pp. 44–53, 2017.

[45] T. Broderick, M. I. Jordan, J. Pitman, *et al.*, "Beta processes, stick-breaking and power laws," *Bayesian Analysis*, vol. 7, no. 2, pp. 439–476, 2012.

[46] D. J. Aldous, "Exchangeability and related topics," in *École d'Été de Probabilités de Saint-Flour XIII—1983*, Springer, 1985, pp. 1–198.

[47] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei, *Sharing clusters among related groups: Hierarchical dirichlet processes*, 2005.

[48] J. Pitman and M. Yor, "The two-parameter poisson-dirichlet distribution derived from a stable subordinator," *The Annals of Probability*, pp. 855–900, 1997.

[49] J. Pitman, "Coalescents with multiple collisions," *The Annals of Probability*, pp. 1870–1902, 1999.

[50] M.-W. Ho, L. F. James, and J. W. Lau, "Coagulation fragmentation laws induced by general coagulations of two-parameter poisson-dirichlet processes," *arXiv preprint math/0601608*, 2006.

[51] F. Wood, J. Gasthaus, C. Archambeau, L. James, and Y. W. Teh, "The sequence memoizer," *Communications of the ACM*, vol. 54, no. 2, pp. 91–98, 2011.

[52] E. Gassiat, "Mixtures of nonparametric components and hidden Markov models," *Handbook of Mixture Analysis*, pp. 343–360, 2017.

[53] D. Stoyan, W. S. Kendall, and J. Mecke, *Stochastic Geometry and its Applications*. John Wiley & Sons, 1987.

[54] J. B. Hough, M. Krishnapur, Y. Peres, B. Virág, *et al.*, "Determinantal processes and independence," *Probability Surveys*, vol. 3, pp. 206–229, 2006.

[55] F. Lavancier, J. Møller, and E. Rubak, "Determinantal point process models and statistical inference," *Journal of the Royal Statistical Society - Series B*, pp. 853–877, 2015.

[56] F. Petralia, V. Rao, and D. B. Dunson, "Repulsive mixtures," in *Advances in Neural Information Processing Systems*, 2012, pp. 1889–1897.

[57] Y. Xu, P. Müller, and D. Telesca, "Bayesian inference for latent biologic structure with determinantal point processes (dpp)," *Biometrics*, vol. 72, no. 3, pp. 955–964, 2016.

[58] I. Bianchini, A. Guglielmi, F. A. Quintana, *et al.*, "Determinantal point process mixtures via spectral density approach," *Bayesian Analysis*, 2018.

[59] A. Scardicchio, C. E. Zachary, and S. Torquato, "Statistical properties of determinantal point processes in high-dimensional Euclidean spaces," *Physical Review E*, vol. 79, no. 4, p. 041 108, 2009.

[60] F. Xie and Y. Xu, "Bayesian repulsive Gaussian mixture model," *Journal of the American Statistical Association*, pp. 1–29, 2019.

[61] S. Dasgupta, "Learning mixtures of Gaussians," in *40th Annual Symposium on Foundations of Computer Science (Cat. No. 99CB37039)*, IEEE, 1999, pp. 634–644.

[62] J. J. Quinlan, G. L. Page, and F. A. Quintana, "Density regression using repulsive distributions," *Journal of Statistical Computation and Simulation*, vol. 88, no. 15, pp. 2931–2947, 2018.

[63] J. Fúquene, M. Steel, and D. Rossell, "On choosing mixture components via non-local priors," *Journal of the Royal Statistical Society - Series B*, vol. 81, no. 5, pp. 809–837, 2019.

[64] S. Frühwirth-Schnatter and G. Malsiner-Walli, "From here to infinity: sparse finite versus Dirichlet process mixtures in model-based clustering," *Advances in Data Analysis and Classification*, vol. 13, no. 1, pp. 33–64, 2019.

[65] E. F. Saraiva, A. K. Suzuki, L. A. Milan, *et al.*, "A Bayesian sparse finite mixture model for clustering data from a heterogeneous population," *Brazilian Journal of Probability and Statistics*, vol. 34, no. 2, pp. 323–344, 2020.

[66] G. Malsiner-Walli, S. Frühwirth-Schnatter, and B. Grün, "Model-based clustering based on sparse finite Gaussian mixtures," *Statistics and Computing*, vol. 26, no. 1-2, pp. 303–324, 2016.

[67] S. Richardson and P. J. Green, "On Bayesian analysis of mixtures with an unknown number of components (with discussion)," *Journal of the Royal Statistical Society - Series B*, vol. 59, no. 4, pp. 731–792, 1997.

[68] M. L. Huber and R. L. Wolpert, "Likelihood-based inference for Matérn type-III repulsive point processes," *Advances in Applied Probability*, vol. 41, no. 4, pp. 958–977, 2009.

[69] P. W. Lewis and G. S. Shedler, "Simulation of nonhomogeneous Poisson processes by thinning," *Naval Research Logistics Quarterly*, vol. 26, no. 3, pp. 403–413, 1979.

[70] M. Plummer, N. Best, K. Cowles, and K. Vines, "CODA: convergence diagnosis and output analysis for MCMC," *R News*, vol. 6, no. 1, pp. 7–11, 2006. [Online]. Available: https://journal.r-project.org/archive/.

[71] J. W. Lau and P. J. Green, "Bayesian model-based clustering procedures," *Journal of Computational and Graphical Statistics*, vol. 16, no. 3, pp. 526–558, 2007.

[72] H. M. Berman, T. Battistuz, T. N. Bhat, W. F. Bluhm, P. E. Bourne, K. Burkhardt, Z. Feng, G. L. Gilliland, L. Iype, S. Jain, *et al.*, "The protein data bank," *Acta Crystallographica Section D: Biological Crystallography*, vol. 58, no. 6, pp. 899–907, 2002.

[73] G. N. Ramachandran, C. Ramakrishnan, and V. Sasisekharan, "Stereochemistry of polypeptide chain configurations," *Journal of Molecular Biology*, vol. 7, pp. 95–99, 1963.

[74] K. V. Mardia, "Statistical of directional data (with discussion)," *Journal of the Royal Statistical Society*, vol. 37, no. 3, p. 390, 1975.

[75] K. V. Mardia, C. C. Taylor, and G. K. Subramaniam, "Protein bioinformatics and mixtures of bivariate von Mises distributions for angular data," *Biometrics*, vol. 63, no. 2, pp. 505–512, 2007.

[76] V. Rao, L. Lin, and D. B. Dunson, "Data augmentation for models based on rejection sampling," *Biometrika*, vol. 103, no. 2, pp. 319–335, 2016.

[77] L. Lin, V. Rao, and D. B. Dunson, "Bayesian nonparametric inference on the Stiefel manifold," *Statistica Sinica*, vol. 27, no. 2, pp. 535–553, 2017, ISSN: 10170405, 19968507.

[78] B. W. Silverman, *Density Estimation for Statistics and Data Analysis*. CRC press, 1986, vol. 26.

[79] K. Roeder, "Density estimation with confidence sets exemplified by superclusters and voids in the galaxies," *Journal of the American Statistical Association*, vol. 85, no. 411, pp. 617–624, 1990.

[80] R. P. Freckleton, P. H. Harvey, and M. Pagel, "Phylogenetic analysis and comparative data: A test and review of evidence," *The American Naturalist*, vol. 160, no. 6, pp. 712–726, 2002.

[81] D. Brawand, M. Soumillon, A. Necsulea, P. Julien, G. Csárdi, P. Harrigan, M. Weier, A. Liechti, A. Aximu-Petri, M. Kircher, *et al.*, "The evolution of gene expression levels in mammalian organs," *Nature*, vol. 478, no. 7369, p. 343, 2011.

[82] M. J. Landis, J. G. Schraiber, and M. Liang, "Phylogenetic analysis using lévy processes: Finding jumps in the evolution of continuous traits," *Systematic Biology*, vol. 62, no. 2, pp. 193–204, 2012.

[83] M. Landis and J. G. Schraiber, "Punctuated evolution shaped modern vertebrate diversity," *bioRxiv*, p. 151 175, 2017.

[84] P. Duchen, C. Leuenberger, S. M. Szilágyi, L. Harmon, J. Eastman, M. Schweizer, and D. Wegmann, "Inference of evolutionary jumps in large phylogenies using lévy processes," *Systematic Biology*, vol. 66, no. 6, pp. 950–963, 2017.

[85] H. H. Newman, F. N. Freeman, and K. J. Holzinger, "Twins: A study of heredity and environment.," 1937.

[86] J. Hirsch, "Behavior genetics and individuality understood," *Science*, vol. 142, no. 3598, pp. 1436–1442, 1963.

[87] P. M. Visscher, W. G. Hill, and N. R. Wray, "Heritability in the genomics era—concepts and misconceptions," *Nature reviews genetics*, vol. 9, no. 4, p. 255, 2008.

[88] C. Andrieu, G. O. Roberts, *et al.*, "The pseudo-marginal approach for efficient monte carlo computations," *The Annals of Statistics*, vol. 37, no. 2, pp. 697–725, 2009.

[89] Y. W. Teh, "A hierarchical bayesian language model based on pitman-yor processes," in *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, Association for Computational Linguistics, 2006, pp. 985–992.

[90] R. E. Kass and A. E. Raftery, "Bayes factors," *Journal of the American Statistical Association*, vol. 90, no. 430, pp. 773–795, 1995.

[91] H. Jeffreys, *The theory of probability*. Oxford University Press, 1961.

[92] E. Paradis and K. Schliep, "Ape 5.0: An environment for modern phylogenetics and evolutionary analyses in R," *Bioinformatics*, vol. 35, pp. 526–528, 2019.

[93] U. Shankarkumar, "The human leukocyte antigen (hla) system," *International Journal of Human Genetics*, vol. 4, no. 2, pp. 91–103, 2004.

[94] J. M. Carlson, Z. L. Brumme, C. M. Rousseau, C. J. Brumme, P. Matthews, C. Kadie, J. I. Mullins, B. D. Walker, P. R. Harrigan, P. J. Goulder, *et al.*, "Phylogenetic dependency networks: Inferring patterns of ctl escape and codon covariation in hiv-1 gag," *PLoS Computational Biology*, vol. 4, no. 11, e1000225, 2008.

[95] J. C. Moravec, Q. Atkinson, C. Bowern, S. J. Greenhill, F. M. Jordan, R. M. Ross, R. Gray, S. Marsland, and M. P. Cox, "Post-marital residence patterns show lineage-specific evolution," *Evolution and Human Behavior*, vol. 39, no. 6, pp. 594–601, 2018.

VITA

Hanxi Sun was born in 1992 in Beijing, China. In 2014, she obtained a B.S. degree in Statistics from the School of Mathematical Sciences and a B.S. degree in computer science from the School of Electronics Engineering and Computer Science at Peking University. Having received an M.A. degree in Statistics from Columbia University in December 2015, she joined Purdue University's department of statistics in August 2016 and earned a Ph.D. degree in Statistics in June 2021. Hanxi's research interests include machine learning, computational statistics, spatio-temporal processes, nonparametric Bayesian modeling, and deep generative models. After graduation, she would join the Hudson River Trading as an algorithm developer to pursue a career in quantitative research.