Statistical Machine Learning in Computational Genetics

by

Shufei Ge

M.Sc., Wuhan University, 2013
B.Sc., Central South University, 2011

Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of
Doctor of Philosophy

in the
Department of Statistics and Actuarial Science
Faculty of Applied Science

SIMON FRASER UNIVERSITY
Summer 2020

Approval

Name:	**Shufei Ge**
Degree:	**Doctor of Philosophy (Statistics)**
Title:	**Statistical Machine Learning in Computational Genetics**
Examining Committee:	**Chair:** Jinko Graham Professor
	Liangliang Wang Senior Supervisor Associate Professor
	Lloyd T. Elliott Supervisor Assistant Professor
	Jiguo Cao Internal Examiner Associate Professor
	Dehan Kong External Examiner Assistant Professor Department of Statistical Sciences University of Toronto
Date Defended:	**July 3, 2020**

Abstract

Statistical machine learning has played a key role in many areas, such as biology, health sciences, finance and genetics. Important tasks in computational genetics include disease prediction, capturing shapes within images, computation of genetic sharing between pairs of individuals, genome-wide association studies and image clustering. This thesis develops several learning methods to address these computational genetics problems. Firstly, motivated by the need for fast computation of genetic sharing among pairs of individuals, we propose the fastest algorithms for computing the kinship coefficient of a set of individuals with a known large pedigree. Moreover, we consider the possibility that the founders of the known pedigree may themselves be inbred and compute the appropriate inbreeding-adjusted kinship coefficients, which has not been addressed in literature. Secondly, motivated by an imaging genetics study of the Alzheimer's Disease Neuroimaging Initiative, we develop a Bayesian bivariate spatial group lasso model for multivariate regression analysis applicable to exam the influence of genetic variation on brain structure and accommodate the correlation structures typically seen in structural brain imaging data. We develop a mean-field variational Bayes algorithm and a Gibbs sampling algorithm to fit the model. We also incorporate Bayesian false discovery rate procedures to select SNPs. The new spatial model demonstrates superior performance over a standard model in our application. Thirdly, we propose the Random Tessellation Process (RTP) to model complex genetic data structures to predict disease status. The RTP is a multi-dimensional partitioning tree with non-axis aligned cuts. We develop a sequential Monte Carlo (SMC) algorithm for inference. Our process is self-consistent and can relax axis-aligned constraints, allowing complex inter-dimensional dependence to be captured. Fourthly, we propose the Random Tessellation with Splines (RTS) to acquire complex shapes within images. The RTS provides a framework for describing Bayesian nonparametric models based on partitioning two-dimensional Euclidean space with splines. We also develop an inference algorithm that is "embarrassingly parallel". Finally, we extend the mixtures of spatial spline regression with mixed-effects model under the Bayesian framework to accommodate streaming image data. We propose an SMC algorithm to analyze online fashion brain image.

Keywords: Statistical learning; Space partitioning; Monte Carlo methods; Kinship; Genome-wide association studies; Spatial Spline, Imaging genetics

Dedication

To my loving husband, my son and my parents.

Acknowledgements

First, I would like to pay my special regards to my senior supervisor Dr. Liangliang Wang and thank her for introducing me to the area of Bayesian Monte Carlo methods and imaging genetics, for encouraging and sending me to attend local and international conferences and workshops. Also for the continuous support and help on my Ph.D study, for her patience, motivation, and immense knowledge and for discussing research with me frequently.

I wish to express my sincere appreciation to my supervisor Dr. Lloyd Elliott for bringing me to the area of machine learning and genetics, for providing research discussions frequently, for being extremely patient with all my questions and for his persistent help and immense support on my Ph.D study.

I would also like to thank all the faculty and staff members in the Department of Statistics and Actuarial Science for offering such a warm and friendly working environment. Special thanks go to Charlene, Kelly, Sadika and Jay for their help and support that made my student life much easier.

My gratitude also extends to Drs. Brent Kirkpatrick, Farouk Nathoo, Jiguo Cao, Yee Whye Teh, Yin Song and Shijia Wang for research collaborations.

During my time at SFU, I had lots of fun and inspiring conversations with my friends and labmates: Alex, Boyi, Charlie, Chirith, Chenlu, Chuyuan, Dongmeng, Erin, Faezeh, Grace, John, Jacob, Luyao, Nate, Payman, Pulindu, Renny, Sidi, Sonny, Tianyu, Will, Yue, Yuping, Yanjun, Zubia, Zhiyang and anyone else I'll be sorry I've forgotten.

In the end, I am grateful to my parents, siblings, sisters-in-law, nieces, nephews and parents-in-law, for their boundless support and encouragement. I would also like to say thank you to my lovely son, for giving me motivation to accomplish my personal goals. And above all else, I would like to show my sincere appreciation to my husband, for his endless love, continuous support and cold humour.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Statistical machine learning is an application of computer algorithms accesses and learns from data (Friedman et al., 2009; Alpaydin, 2020). The goal of machine learning is often more about making predictions than understanding. Statistical machine learning has played a key role in many areas, such as health sciences, financial industry, biology and genetics (Beam and Kohane, 2018; James et al., 2013; Khandani et al., 2010; Baldi and Brunak, 2001; Tarca et al., 2007). A vast set of learning methods have been developed for learning from data. Broadly speaking, statistical learning methods can be classified as supervised or unsupervised learning. In supervised learning, we have an outcome measure, usually named responses or dependent variables, either qualitative (e.g. with/without disease) or quantitative (e.g. cortical thickness). We try to predict the outcome measure via a set of input measures, often called predictors, features or independent variables (Russell and Norvig, 2002; Mohri et al., 2018; Alpaydin, 2020). In unsupervised learning, we do not observe any outcome measure and generally the task is to cluster or group the data via a set of features (Barlow, 1989; Hofmann, 2001; Dy and Brodley, 2004).

The outcome measures vary among different applications, either qualitative or quantitative. We call the prediction task classification, if the response is qualitative, and call the prediction task regression, if the response is quantitative (Draper and Smith, 1998; Breiman et al., 1984; Kotsiantis et al., 2007). In recent decades, various learning methods have been developed for supervised learning. Linear models are classical methods in supervised learning, in which we assume a linear relationship between the responses and the predictors. However, this assumption often violates the nature of many real applications. Methods beyond linearity have been developed, including smoothing splines, kernel smoothing methods and additive models. In addition, more flexible nonparametric models, such as decision trees, neural networks, and support vector machine based approaches are developed to model the nonlinearity between responses and predictors. With recent advances in biological techniques, huge amounts of genetic data sets have been generated. More advanced learning methods are in demand to analyze and interpret these data sets, which would benefit the development of drugs, treatment and policy.

In this thesis, we aim to develop learning methods to address several important problems in computational genetics.

The computation of kinship coefficients is important for assessing the genetic sharing between pairs of individuals, which is fundamental in correcting for cryptic-relatedness in genome-wide association studies (Yu et al., 2005; Rakovski and Stram, 2009; Kang et al., 2010), understanding breeding relationships and human genealogy (Thompson, 1985), and also for describing and fostering genetic diversity in animal breeding. Genetic diversity is critical since it prevents the raising of recessive Mendelian disease at a population level (Woods et al., 2006). In Chapter 2, we develop the fastest known algorithms for computing kinship coefficient of a set of individuals with a known pedigree. Our method also scales to large pedigrees. In addition, our algorithms consider the possibility that the founders of the known pedigree may themselves be inbred, which had not previously been addressed in literature, and computes the appropriate inbreeding-adjusted kinship coefficients.

In Chapter 3, we develop a Bayesian bivariate spatial group LASSO model for multivariate regression analysis applicable to studies examining the influence of genetic mutations on brain structure. This spatial model is motivated by an imaging genetics study of the Alzheimer's Disease Neuroimaging Initiative, where the objective is to uncover the association between neuroimaging measures as quantitative traits and genetic variations (Vounou et al., 2010; Stein et al., 2010; Silver et al., 2011; Inkster et al., 2010; Hibar et al., 2011; Ge et al., 2012; Thompson et al., 2013; Stingo et al., 2013; Zhu et al., 2014; Hibar et al., 2015; Huang et al., 2015, 2017; Lu et al., 2017). A bivariate spatial process model is developed to accommodate the correlation structures typically seen in structural brain imaging data. We thereby model both the spatial correlation in the imaging phenotypes obtained from neighbouring regions on the same hemisphere of the brain, and also the correlation in the phenotypes obtained from different hemispheres (left/right) of the brain. We develop both a Gibbs sampling and a mean-field variational Bayes (Ormerod and Wand, 2010) algorithm to fit the model. We also incorporate Bayesian false discovery rate procedures (Morris et al., 2008) to select SNPs.

Thirdly, motivated by the need of cutting multi-dimensional Euclidean space with non-axis aligned cuts, we propose the Random Tessellation Process (RTP) to model complex genetic data structures to predict disease status in Chapter 4. The RTP is a multi-dimensional partitioning tree process allowing non-axis aligned cuts, which generalizes axis-aligned methods such as decision trees and the Mondrian process (Kemp et al., 2006; Roy and Teh, 2008). The RTP provides a framework for describing Bayesian nonparametric models based on space partitioning. The RTP includes the Mondrian process and the binary space partitioning-tree process (Fan et al., 2018) as special cases. Our process is self-consistent and can relax axis-aligned constraints, allowing complex inter-dimensional dependence to be captured. We derive a sequential Monte Carlo algorithm (SMC) (Doucet et al., 2000) for inference, which takes advantage of the hierarchical structure of the generating process for the RTP prior. We also provide random forest versions of RTPs.

In Chapter 5, motivated by modeling shapes within images via Bayesian nonparametric space partitioning methods. We propose the Random Tessellation with Spines (RTS). The proposed nonlinear boundaries enable more complex data structures to be acquired. The RTS construction is based on the theory of stable iterated tessellations in stochastic geometry. We develop an SMC

algorithm for the RTS inference, which takes advantage of the hierarchical structure of the generative process of the RTS. We use simulation studies to demonstrate that our Random Tessellation with Splines may be an appropriate consistent process. We apply our RTS to an HIV-1-infected human macrophage data set, and compare with modern machine learning methods.

Finally, we extend the mixtures of spatial spline regression with mixed-effects model under the Bayesian framework to accommodate streaming image data in Chapter 6. The traditional Markov chain Monte Carlo approach is not scalable to streaming image data since it requires all observed information to update the posterior distribution of the parameters. To tackle this issue, we propose an SMC algorithm to analyze online fashion image data. The existence of model sufficient statistics improves the efficiency of the proposed online SMC algorithm. Instead of saving all batch data for inference, we only require storage of the model sufficient statistics and every data point is only used once, which is a configuration well suited for large-scale stream type data. In addition, the proposed algorithm provides an unbiased estimator of the marginal likelihood as a by-product of the approach, which can be used for model selection. Numerical experiments are used to demonstrate the effectiveness of our method.

In Chapter 7, we conclude this thesis work and present directions for future work.

Chapter 2

Efficient Computation of the Kinship Coefficients

2.1 Introduction

The kinship coefficient is a measure of genetic similarity between pairs of individuals. Computing kinship coefficients is fundamental to understanding breeding relationships and human genealogies (Thompson, 1985). Kinship coefficients have been used for correcting GWAS case-control studies for known relationships and for reducing spurious signals in case-control and quantitative trait association studies (Yu et al., 2005; Rakovski and Stram, 2009; Kang et al., 2010; Eu-Ahsunthornwattana et al., 2014). They have also been used for pedigree case-control disease association (Thornton and McPeek, 2007). When breeding pedigree animals such as dogs, cats, horses, cows, or endangered species, using kinship coefficients to prevent inbreeding would foster genetic diversity. This genetic diversity is important, because it prevents excessive homozygosity and the raise of recessive Mendelian disease in the population as a whole (Woods et al., 2006). There is some confusion as to how quickly kinship coefficients can be computed (Abney, 2009). The kinship coefficients are defined by counting paths of genetic transmission of alleles. The popular recursive algorithm described in Karigl (1981) was implemented in Zheng and Bourgain (2009).

There were two ways to obtain kinship coefficients. One is the empirical kinship coefficient estimated using genetic data without knowing the pedigree graph, and the other is based on defining the kinship coefficient as a function of a pedigree graph. Recently, Wang (2019) proposed to estimate genetic similarity using phylogenies. The topic of estimation from data is outside the scope of this chapter. This chapter focuses on computing and approximating the kinship coefficient as a function of a pedigree graph.

While there exists a nice set of recursive equations, the algorithmic properties have needed more treatment in the literature. This chapter aims to fill this gap by introducing three fastest known kinship algorithms: two exact algorithms and one approximate algorithm. Our numerical studies show that our proposed algorithms are several orders of magnitude faster than the Zheng and Bourgain (2009) and are faster than the Abney (2009) when the number of individuals per generation is large.

2.2 Background

The pedigree graph, $P = (V, E)$, is the canonical descriptor of known relationships. This is a directed acyclic graph with individuals as vertices, V, and each vertex having a gender, either male or female. The graph has edges, E, directed from parent to child indicating direct relationship. The graph is acyclic, meaning that no individual can be their own ancestor. The graph has in-degree at most two with one parent of each gender, meaning that each individual has at most two parents, one of each gender. Let $I \subseteq V$ be the *individuals of interest*, or the individuals whose relationships we would like to quantify. Let there be n individuals in the pedigree, meaning that $n = |V|$.

A pedigree graph is a complete description of relationships between individuals, because it includes every parent-child edge. Many interesting quantifications of pedigree relationships are NP-complete to compute. Most crucially, the pedigree likelihood (Piccolboni and Gusfield, 2003; Kirkpatrick, 2011), which is the basis of many other pedigree calculations, is NP-complete. This means that the pedigree and its quantities are inefficient. We now turn our attention to more efficient representations.

There are a variety of ways to summarize the relationships in a pedigree in condensed forms. One can look at a pair-wise pedigree relationships and describe those using the language of family relationships: parent, child, nephew, cousin, etc. One can look at pair-wise genetic relationships and ask whether two alleles are inherited from an identical allele in an ancestor, i.e. whether the alleles are identical-by-descent (IBD). One can look at the k-wise relationships describing the probability of $2k$ alleles being IBD which is *generalized kinship coefficient* (Abney, 2009). One can also summarize a pair-wise relationship by the probability of two random alleles being IBD; this is the kinship coefficient, sometimes called the coefficient of relationship. This last summary of relationships, called the *kinship coefficient*, is the subject of this chapter. This chapter discusses the kinship coefficient, as it is defined, as a function of a pedigree graph.

Formally, for two individuals in a pedigree, define *identity-by-descent (IBD)* as the event that both individuals inherited an allele from the same ancestor. For two individuals i and j, there are four alleles, two for individual i, a_1 and a_2 and two for individual j, b_1 and b_2. In complete generality, there are 15 ways for these 4 alleles to be IBD (Jacquard, 1972). We draw these 15 possibilities as graphs on 4 alleles, as in Figure 2.1. There is an edge between every pair of alleles that is IBD. For example, allele a_1 and a_2 can be IBD while none of the other alleles are IBD, see row 3 of Figure 2.1. As another example, alleles a_1, b_1, and b_2 could be IBD while allele a_2 is not related to the others, see row 7 of Figure 2.1. Mathematically, the *kinship coefficient* of two individuals i and j in a pedigree is the probability of IBD between two randomly drawn alleles, one from each person. Let the matrix ϕ contain all the pair-wise kinship coefficients in a pedigree. Entries ϕ_{ij}, for $i \neq j$, are kinship coefficients, and entries ϕ_{ii} are related to inbreeding coefficients. Label the alleles of individuals i and j with distinct labels: (a_1, a_2) and (b_1, b_2), respectively. By the definition of the

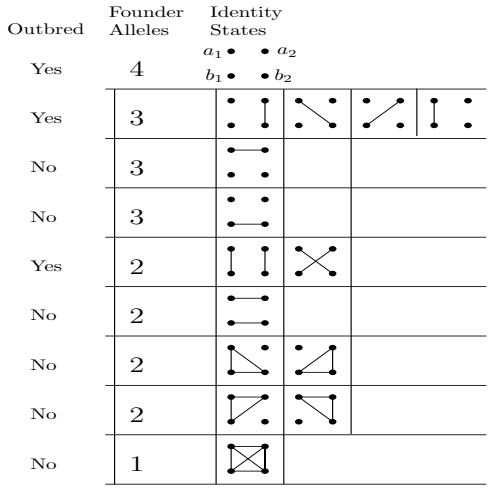


Figure 2.1: **Identity States.** Each identity state is a graph with four alleles of two individuals drawn as the nodes and edges appearing between every pair of IBD alleles. The 15 identity states are grouped so that each row corresponds to one of the 9 condensed identity states. The number of founder alleles for each identity state is listed, along with whether the identity state is out-bred.

kinship coefficient,

$$\begin{aligned}\phi_{ij} = (1/4)Pr[a_1 \equiv b_1] + (1/4)Pr[a_1 \equiv b_2] \\ + (1/4)Pr[a_2 \equiv b_1] + (1/4)Pr[a_2 \equiv b_2],\end{aligned}$$

where this '≡' operator means IBD. Unlike the IBD probabilities which can apply to specific alleles in data, the kinship coefficients are an expectation over the structure of the pedigree and are independent of the data. The kinship coefficients are often defined as an expectation over all possible inheritance paths (or all possible joint assignments to the segregation indicators of a pedigree), this chapter will develop the equivalence with an expectation over the identity states and their probabilities.

Suppose that we consider only out-bred IBD possibilities (defined as having $\phi_{ii} = 1/2$ for all i). The identity states encode all the possibilities for IBD, including inbreeding possibilities, and Figure 2.1 has three rows indicating they are out-bred (rows 1, 2, 5). These correspond to the a pair of individuals, i and j sharing zero, one, or two alleles IBD, respectively. In this case, we can denote the probability of each of these event as f_0^{ij}, f_1^{ij}, and f_2^{ij} respectively. In this particular case, the kinship coefficients are simple to compute

$$\phi_{ij} = (2f_2^{ij} + f_1^{ij})/4.$$

Assume that there might be inbreeding in the pedigree. Now, we wish to compute all the kinship coefficients in one computation. The recursive algorithm for computing this was developed in detail in Thompson (1985) and Karigl (1981). This section will only give the recursive equations which are

a top-down recursion on the pedigree. For all founders f, the matrix is initialized as

$$\begin{aligned} \phi_{ff} &= 1/2, \text{ and} \\ \phi_{fj} &= 0, \text{ for any } j \text{ that is not a descendant of} f. \end{aligned}$$

Let the mother and father of i be denoted m and p. Then use the kinship coefficient between j and i's parents to compute

$$\phi_{ij} = (\phi_{mj} + \phi_{pj})/2,$$

where i is not an ancestor of j and $i \neq j$.

In a similar manner, we compute the kinship coefficient for i from its parents,

$$\phi_{ii} = (1 + \phi_{mp})/2,$$

where this function of ϕ_{mp} accounts for the probability of picking the same allele when uniformly choosing two alleles from the same person.

Some researchers write the kinship matrix as given by the output of the above recursion. Others transform the matrix, so that it contains the inbreeding coefficients for each individual instead of the kinship coefficients. This transformation, applied only to the diagonal, is

$$\begin{aligned} \Phi_{ij} &= \phi_{ij} \text{ for all } i \neq j, \text{and} \\ \Phi_{ii} &= 2\phi_{ii} - 1. \end{aligned}$$

We find it convenient to represent the inbreeding coefficient on the diagonal, and this is the convention used in this chapter.

2.3 Methods

Before introducing the details of the algorithms, we need to develop some mathematical methods that ease the task of algorithm development. First, the kinship recursion can be initialized in several ways. If the founders are known to be inbred, it is appropriate to initialize the recursion with that information, for more accurate computation of kinship coefficients. Second, the kinship coefficients can be represented in terms of identity state coefficients. This relationship is useful for developing a sampling algorithm that samples identity states.

2.3.1 Generalized Initialization of the Kinship Recursion

We relate the kinship to inbreeding coefficients estimated for unrelated individuals. The founders of a pedigree have an ancestry that relates them, even if that ancestry is not recorded in the pedigree graph.

Let Ψ be the kinship coefficients for the founders, i.e. a matrix of $F \times F$ where F is the number of founders. We initialize the algorithm using the kinship coefficients of the founders as follows:

$$\begin{aligned} \phi_{ff} &= (1 + \Psi_{ff})/2, \\ \phi_{fg} &= \Psi_{fg} \text{ for founders } f \neq g, \text{ and} \\ \phi_{fj} &= 0, \text{ for non-founder, not a founder child,} \\ &\quad j \text{ that is not a descendant of } f, \end{aligned} \tag{1}$$

where Ψ_{ff} is the inbreeding coefficient for founder f. Recall that the diagonal elements of Ψ are the inbreeding coefficients. The recursive equations also need to be slightly modified, by the addition of the following case for founder children c not descended from f:

$$\begin{aligned} \phi_{fc} &= \phi_{cf} = (\phi_{m(c),f} + \phi_{p(c),f})/2, \\ &\quad \text{for founder } m(c) \text{ or } p(c) \text{ and founder } f. \end{aligned}$$

It may be difficult to obtain the kinship coefficients on the founders if their genealogy is unknown. More feasibly, we can estimate the inbreeding coefficient from the homozygosity in Leutenegger et al. (2003). We modify the above recursion in Equation (1) to initialize it with the average inbreeding among all the founders using the average inbreeding coefficient for founders h, $\phi_{fg} = 1/F \sum_h \Psi_{hh}$ for founders f and g, $f \neq g$. The change to the recursive equations is still relevant.

If the kinship coefficients of the founders are not properly taken into account, then the kinship will be computed assuming that the founders are out-bred. Thus, the method will inaccurately represent the genetic relationships between the individuals whose ancestry contains inbreeding in the founders.

2.3.2 Relating Identity States and the Kinship Coefficient

To relate the identity coefficients into kinship coefficients, we take an expectation over the identity states. Since there is a deterministic mapping between the condensed identity states and the identity states, the expectation can be written in terms of the condensed identity states. The proof for this approach takes as a first step the expression of the kinship as an expectation over inheritance paths (Kirkpatrick, 2012), which is easily converted into an expectation over identity states.

For an identity state, let $t \in \{aa, ab, bb\}$ denote an *edge type*, for example, ab indicates any edge between the alleles in two different individuals a and b, i.e. and edge between one of the nodes $\{a_1, a_2\}$ and one of the nodes $\{b_1, b_2\}$. Another example, aa, indicates an edge between nodes a_1 and a_2 within the same individual. Recalling that the identity state is a graph on nodes $\{a_1, a_2, b_1, b_2\}$, we let $e(s, t)$ be a function of identity state s and edge type t which gives the number of edges of type t in identity state s. For example, in Figure 2.1, row 8, column 1, $e(s, aa) = 1, e(s, ab) = 2, e(s, bb) = 0$.

The kinship coefficient $\Phi_{a,b}$ between individuals $a \neq b$ is related to the identity states and their coefficients via the following expectation over the set of 15 possible identity states (a set that we

denote by $\mathcal{S}$):

$$\Phi_{a,b} = \sum_{s \in \mathcal{S}} \frac{e(s, ab)}{4} \mathbb{P}[S = s]. \tag{2}$$

In this equation, $e(s, ab)/4$ can be interpreted as the fraction of the 4 possible edges between one node in $\{a_1, a_2\}$ and one node in $\{b_1, b_2\}$ that are present in the identity state s.

The inbreeding coefficient $\Phi_{a,a}$ is computed slightly differently via:

$$\Phi_{a,a} = \sum_{s \in \mathcal{S}} e(s, aa) \mathbb{P}[S = s], \tag{3}$$

where $e(s, aa)$ indicates whether the single possible edge between nodes a_1 and a_2 exists.

The transformation to obtain the kinship coefficient from Equation (3) is $\phi_{a,a} = (1 + \Phi_{a,a})/2$. The convention in this chapter is to use $\Phi_{a,a}$.

2.4 Efficient Implementations

We will discuss three algorithms for obtaining kinship coefficients which are efficient in different scenarios. Recall that there are n individuals in our pedigree. The first scenario is exact computation in time polynomial to the size of the pedigree (i.e., running-time $O(n^2)$), and the second scenario is approximate computation in linear time (i.e., running-time $O(n)$).

First, the recursions given, above, can be implemented efficiently in an $O(n^2)$-time algorithm. The challenging portion of the recursive equations is the check for which individuals are ancestors of each other. This can be done in constant time, provided that the correct data structure tabulates the ancestry information. The details of this algorithm are given in Section 2.4.1.

Second, when the individuals of interest are a subset of the individuals in the pedigree, then we can use a divide-and-conquer approach for applying the first exact algorithm. In this approach we recursively cut the pedigree graph to obtain sub-pedigrees on which to apply the full exact algorithm. As long as the individuals of interest all reside in a single sub-pedigree, this approach is more efficient than the first exact algorithm. Details are given in Section 2.4.2.

Third, when a linear algorithm is desired and the kinship coefficients of a subset of individuals are needed, there is a sampling algorithm that samples identity states. Given sample size d and the kinship of $\sqrt{n}$ founders, the sampling algorithm works in $O(nd)$ time and the kinship of $\sqrt{n}$ of the individuals is computed. A smaller version of this sampling algorithm that estimates the kinship of one pair of individuals has previously been introduced (Sun et al., 2014). That algorithm was evaluated for sampling error, and it was discovered that several thousand identity states are needed for very large pedigrees. Details of this algorithm appear in Section 2.4.3. Our proposed algorithms are implemented as PedKin in C++ and are available under the GNU GPL v2.0 open source license. The PedKin source code is available at: `http://www.intrepidnetcomputing.com/research/code/`.

2.4.1 Efficient, Exact Algorithm

There is an $O(n^2)$-time implementation of the kinship calculation where n is the number of individuals in the pedigree, and $I = V$. Note that any implementation that touches every cell of the kinship matrix requires running-time at least $O(n^2)$. Any implementation of the kinship that represents the full kinship matrix requires $O(n^2)$ space.

We define A_i, the *ancestor set* for individual i in the pedigree, as the set containing i and all its ancestors. This object can easily be computed in $O(n^2)$ time. The ancestor set allows us to do the ancestry check in the kinship recursion in constant time.

The ancestor sets are computed in a top-down recursion on the pedigree graph. Initialize the recursion with $A_f = \{f\}$, where $f \in V$ is a founder in the pedigree, meaning the individual has no parents.

Now, the remaining individuals' ancestor sets are computed from their parents' using $A_i = \{i\} \cup A_{m(i)} \cup A_{f(i)}$, where $m(i)$ is the mother of i and $f(i)$ is the father of i.

The top-down order is obtained by creating a topological sort of the graph. This is done by creating a queue of individuals, initializing the queue with the founders, and popping an individual from the front of the queue while simultaneously adding their children to the end of the queue. This produces an order such that each individual is considered after all of their parents.

The recursion from Section 2.2 can now be implemented in $O(n^2)$ time, using the ancestor sets. The ancestor sets can be queried quickly, if they are stored in a look-up table. This makes the recursion a double loop over the individuals in the pedigree, when those individuals are considered in the top-down order. Algorithm 1 gives the details of the recursive loops.

2.4.2 Faster, Recursive-Cut Exact Algorithm

In the case where we require the kinship coefficients of a subset of the pedigree individuals, $I \subset V$, we can improve the running-time for the exact calculation. Recall from Section 2.3.1, that a correct kinship computation can be done after initializing the founders of a pedigree with their known kinship coefficients.

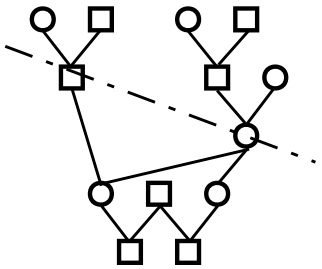


Figure 2.2: **Recursive Cut.** The two individuals through which the dashdotted line passes are in the cut set. The older subpedigree has those two individuals as leafs and has two components. The younger pedigree has those two individuals as founders.

Consider a large pedigree, which is a directed acyclic graph with the founders as sources and the leafs as sinks. Informally, we can segment the large graph by taking vertex cuts that run 'horizontally' and create a generation. The individuals in the cut become the leafs of the upper segment and the founders of the lower segment of the pedigree. This splits the pedigree in half at the cut, and allows us to apply the exact algorithm from Section 2.4.1 to each segment of the pedigree. Formally, a *vertex*

Algorithm 1 $O(n^2)$ Exact Kinship Algorithm

Let A be an $n \times n$ matrix initialized with all zeros.
for all the founders $i \in V$ **do**
    Let $A_{ii} = 1$
for each $i \in V$ with parents $m(i)$, $f(i)$ such that their A row is set **do**
    for each $j \in p(i)$ **do**
        for each $v \in V$ **do**
            $A_{iv} = A_{m(i)v}$ AND $A_{f(i)v}$
            if $v == i$ **then**
                $A_{ii} = 1$
Let Φ be an $n \times n$ matrix initialized with -1.
for every founder $f \in V$ **do**
    $\Phi_{f,f} = (1 + \Psi_{ff})/2$
for every pair of founders $f, g \in V$ with $f \neq g$ **do**
    $\Phi_{f,g} = \Psi_{fg}$
for every founder f and every non-founder $j \in V$, j not founder child **do**
    if $A_{jf} == 0$ **then**
        $\Phi_{fj} = 0$
for every $i \in V$ whose parents have been assigned kinship **do**
    for every $j \in V$ whose parents have been assigned kinship **do**
        if $i == j$ **then**
            $\Phi_{ii} = (1 + \Phi_{mp})/2$
        else
            if $A_{ji} == 0$ **then**
                $\Phi_{ij} = (\Phi_{mj} + \Phi_{pj})/2$
for every $i \in V$ **do**
    $\Phi_{ii} = (2 * \Phi_{ii}) - 1$

cut is defined as a partition of the vertices of the graph into two sets such that the *cut vertices*, when removed, disrupt every path from any source to any sink. A vertex cut produces two edge-disjoint subgraphs. In our application the sub-pedigrees will share the vertices of the cut set, but will be edge-disjoint. In one subpedigree, the cut vertices will be the leafs, and in the other subpedigree, the cut vertices will be the founders.

A *pedigree cut* for the purposes of this algorithm will be defined as a set of cut vertices which defines a generation of individuals. Any pedigree cut separates the pedigree into two sub-pedigrees, one containing the cut vertices as leafs which we will refer to as the upper, or older, sub-pedigree, and one containing the cut vertices as founders which we will refer to as the lower, or younger,

sub-pedigree. In Figure 2.2, the two individuals through which the dashdotted line passes are in the cut set. The upper sub-pedigree, by the placement of the pedigree cut, now has new leaf individuals, which are the cut vertices. Let the cut set be set $C^i \subseteq V$. Let the older subpedigree be for individuals V^i and the younger subpedigree be for individuals V^{i+1} In this case, we know that $C^i \subseteq V^i$ and $C^i \subseteq V^{i+1}$.

We can recursively bipartition the pedigree and sub-pedigrees many times to get a reasonable running time of the exact kinship algorithm on any single sub-pedigree. The finest recursive partitioning will produce generational sub-pedigrees, but with the flexibility of definitions to allow staggered generations. The number of generations puts a lower bound on the number of recursive cuts that are possible. Also, the individuals of interest all need to appear in a single sub-pedigree, if we are to obtain all their pair-wise kinship coefficients from this algorithm.

Now, we apply the exact kinship algorithm for the sub-pedigrees from the top down. The kinship coefficients for the leafs of each sub-pedigree are used to initialize the founder kinship coefficients for the next sub-pedigree. This is done iteratively down the pedigree, until the kinship coefficients of the individuals of interest are obtained.

When applied to a generational pedigree drawn using the diploid Wright-Fisher model, the pedigree has $|C^i| = 2N$ individuals per generation for all generations $0 \leq i \leq G$ and $V^i = C^i$. This recursive-cut kinship algorithm runs on this generational pedigree in $O(N^2G)$ time. More generally if we cut an arbitrary pedigree into m partitions with the maximum partition having $s = |V^i|$ individuals, then the running-time of the recursive-cut kinship algorithm is $O(s^2m)$. When n is the number of individuals in the pedigree, this is an improvement on the $O(n^2)$ running-time of the exact method, since $s \leq n/m$.

The main disadvantage of both these exact algorithms is that they are only polynomial in running time and perhaps may not be fast enough for applications on very large pedigrees. For very large pedigrees, we need a linear-time algorithm for scalability.

2.4.3 Efficient, Approximate Algorithm

There is a linear-time approximate algorithm. Recall that n is the number of individuals in the pedigree. The approximate algorithm runs in $O(nd)$-time algorithm for $I \subset V$ where $|I| = O(\sqrt{n})$, $F = O(\sqrt{n})$ for number of founders and d is the sample size. This algorithm quickly estimates the kinship coefficients by sampling identity states and using the expectations in Equations (2) and (3). Recall that in diploid individuals, each individual has two alleles at every site in the genome. Of these two alleles, one comes from the father, and one from the mother. From each parent, the allele is copied either from the grand-father or from the grand-mother. This binary choice of grand-paternal origin is usually stored in *segregation* indicators, $x_i = (x_i^m, x_i^f)$, for individual i where $x_i^p \in \{m, f\}$. An inheritance path consists of the segregation indicators of all the individuals in the pedigree.

Consider the graph of all the alleles of all the individuals at one site with edges connecting the alleles that are inherited from parent to child (i.e., the edges indicated by the segregation indicators). This is a graph of the inheritance path, and it has connected components. The connected component

Algorithm 2 $O(nd)$ Approximate Kinship Algorithm when $|V| = n$, $|I| = O(\sqrt{n})$, $F = O(\sqrt{n})$ and d is the number of samples.

Let Φ be the $n \times n$ matrix initialized with zeros.
for s in 1 to d **do**
    for $i \in V$ **do**
        Flip 2 $\{m, f\}$-labeled coins, and assign their values to (x_i^m, x_i^f).
    Initialize all $c_i = (0, 0)$; Let $counter = 1$.
    for $l \in V$ where l is a leaf **do**
        Let $(c_i^m, c_i^f) = (counter, counter + 1)$
        $counter = counter + 2$
    for each $i \in V$ provided all of i's children have been processed **do**
        for $p \in \{m, f\}$ and $j \in V$ being the appropriate gendered parent **do**
            if $c_j^{x_i^p} < c_i^p$ **then**
                $c_j^{x_i^p} = c_i^p$
    for every pair of founders r and q **do**
        flip a coin for edges $(a_1, b_1)(a_2, b_2)$ or $(a_1, b_2)(a_2, b_1)$, $a = 0$ or $a = 1$, respectively
        for $x \in \{m, f\}$, and let $y \in \{m, f\} \setminus x$ **do**
            Let $u \in [0, 1]$ be a uniform random variable
            if $u \leq \Psi_{rq}$ **then**
                if $a == 0$ **then**
                    $c_r^x = c_q^x$
                else
                    $c_r^x = c_q^y$
    for each $i \in V$ provided all i's parents have been processed, denote i_p as i's parent **do**
        $c_i^p = c_{i_p}^{x_i^p}$
    for each $i \in I$ **do**
        for each $j \in I$ **do**
            Create the identity state graph for $(c_i, c_j) = (c_i^m, c_i^f, c_j^m, c_j^f)$
            if $i == j$ **then**
                if $c_i^m == c_i^f$ **then**
                    $\Phi_{ii} = \Phi_{ii} + \frac{1}{d}$
            else
                $\Phi_{ij} = \Phi_{ij} + \frac{e}{4d}$, where e is the number edges between c_i and c_j in the identity state graph.

is the set of inherited copies of a particular ancestral allele, which is the root of the connected component. Let the *Connected Component (CC) membership* of each allele be given by a tuple of integers, $c_i = (c_i^m, c_i^f)$, for individual i.

Our method, Algorithm 2, will sample an inheritance path with one pass through the pedigree. When considering each individual, the algorithm flips a coin to set the segregation indicators for that individual. Later, these indicators will be used to determine which alleles are identical-by-descent for this inheritance path.

Two more passes through the pedigree will be used to set the CC membership with consistent values. The leaf alleles of the pedigree are populated with distinct integers representing their putative CC membership. The first pass proceeds from the bottom of the pedigree to the top, and processes each child before their parents. For each allele in each individual, their CC membership integer is copied to the allelic ancestor given by the segregation indicator. If two children give two different CC memberships to the parent, the tie is broken with the largest integer value. To account for founder inbreeding, we merge founder CC memberships randomly according to the probabilities given by the initialization kinship coefficients. The second pass proceeds from top to bottom, and the tie-broken CC membership integers are simply copied back down the path of allelic inheritance. After these two passes, every allele in a connected component of the inheritance path graph will have the same CC membership, and every pair of alleles from different CC's will have distinct CC membership unless the CC's were randomly merged by founder inbreeding.

Finally, for every pair of individuals of interest, we can use the CC membership to obtain the identity state. This identity state can be used to update the estimated kinship with the appropriate terms from Equations 2 and 3.

The details of these procedures are left to Algorithm 2. We will explore the convergence properties of this sampling algorithm using simulations in Section 2.4.4.

2.4.4 Simulation Results

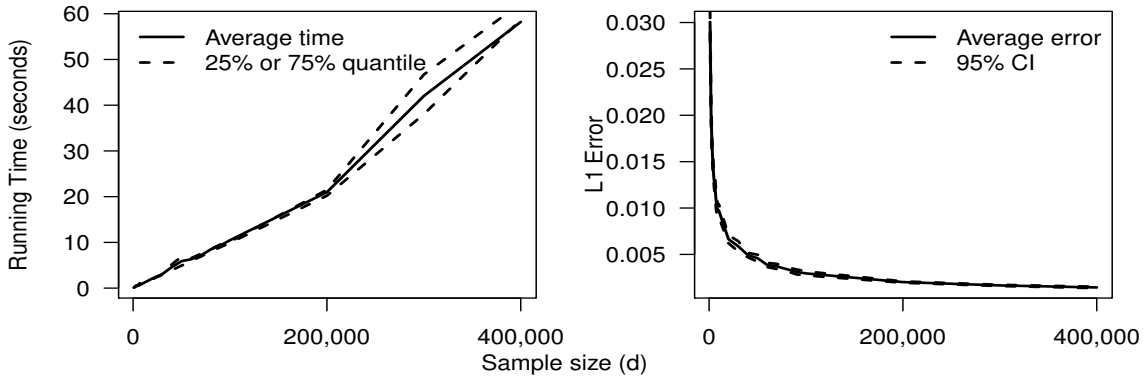


Figure 2.3: **Simulation results of the approximate algorithm.** The left panel represents the relationship between of running time (seconds) and number of iterations and the right panel represents the relationship between L_1 error and number of iterations.

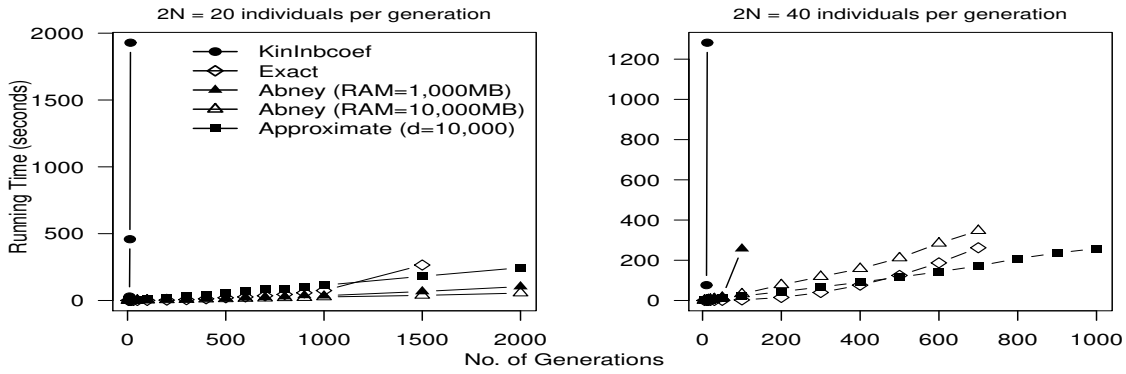


Figure 2.4: **Comparison of running time.** two panels describe the relationship between the running time (in seconds) and the number of generations G.

The diploid Wright-Fisher model is used to generate pedigree data in our simulation study. To assess the convergence of our approximate kinship algorithm, we simulate a scenario with $2N = 20$ individuals per generation, and $G = 10$ generations in total. The sample size d ranges from 1000 to $400,000$. We measure the error of estimates in the L_1 form. The left panel in Figure 2.3 shows that the running time increases almost linearly with number of iterations. The right panel in Figure 2.3 indicates that L_1 form error decreases rapidly before the sample size reaches $3,000$. The L_1 error is 1.6×10^{-2} when the sample size $d = 3,000$ and keeps decreasing with the the sample size d increases.

KinInbcoef (Zheng and Bourgain, 2009) is a C++ program that computes inbreeding and kinship coefficients for general pedigrees using the recursion algorithm mentioned in Section 2.2, IdCoefs (Abney, 2009) is a C program that computes the generalized kinship coefficients and condensed identity coefficients. Both programs will be used as the competitors in following simulations.

To compare the running time of our exact algorithm and approximate kinship algorithm with KinInbcoef and IdCoefs, we simulate two scenarios of pedigree data: one is 20 individuals per generation and another is 40 individuals per generation. We consider a sequence of numbers of generations for both scenarios. The simulations were carried out on the Grex SGI Altix XE 1300 cluster of Westgrid. The results show both our exact and approximate algorithms can handle large pedigrees efficiently.

We compare our exact algorithm with the two competitors. Figure 2.4 shows our exact method is several orders of magnitude faster than the KinInbcoef, and it is faster than IdCoefs when the number of individuals per generation is not too small. Shown on the right panel, IdCoefs requires a large RAM size to run efficiently. Note that its performance becomes much worse if we decrease the RAM size from 10,000 MB to 1,000 MB. Therefore, IdCoefs cannot scale to large pedigrees.

The approximate method is not only efficient but also accurate. As shown in Figure 2.4, the time cost of our approximate method is linear with number of generations. We use the sample size $d = 10,000$. It runs faster than the exact algorithm when the pedigree is larger, for example, when ($2N = 20$, $G > 1000$) and ($2N = 40$, $G > 400$). The corresponding accuracy of the estimate is to 3 decimal digits.

2.5 Discussion

The above kinship algorithms are applicable to large pedigrees, since they either have efficient polynomial or linear running-times. This chapter gives two exact algorithms and one approximation algorithm. The fastest known exact algorithm for kinship computations is the recursive-cut exact algorithm. All these algorithms are easily run using the source code that is published in tandem with this chapter. All of the algorithms in this chapter have running times that are parameterized by the number of individuals. Since the kinship coefficients are defined by inheritance paths which correspond to edges in the pedigree graph, it is possible that the number of edges in a pedigree graph provides the true lower-bound on the number of operations needed to compute the exact kinship coefficients. We leave it as an open problem whether there is an efficient algorithm, perhaps along the lines of the recursive-cut algorithm, that has running-time parameterized by the number of edges in the pedigree graph. Another open problem is whether there is a sparse algorithm for computing the kinship. This type of algorithm would only compute the non-zero entries in the kinship matrix. If designed properly, such an algorithm would need far less space, since it would not need to represent the entire kinship matrix. We believe that such an algorithm does exist and has yet to be discovered.

Chapter 3

A Bayesian Spatial Model for Imaging Genetics

3.1 Introduction

We consider multivariate multiple regression modeling within the context of imaging genetics where interest lies in uncovering the associations between genetic variations and neuroimaging measures as quantitative traits (QTs). This problem has received a great deal of attention recently and is challenging because it combines the analysis of neuroimaging data with genetic data (Vounou et al., 2010; Stein et al., 2010; Silver et al., 2011; Inkster et al., 2010; Hibar et al., 2011; Ge et al., 2012; Thompson et al., 2013; Stingo et al., 2013; Zhu et al., 2014; Hibar et al., 2015; Huang et al., 2015; Kong et al., 2015; Huang et al., 2017; Lu et al., 2017; Kong et al., 2020). Recent reviews of statistical issues in this area are discussed in Liu and Calhoun (2014) and Nathoo et al. (2019).

The neuroimaging measures can serve as endophenotypes for neurological disorders such as Alzheimer's disease (AD). AD has been considered widely as an application in imaging genetics with many studies focussing on the Alzheimer's Disease Neuroimaging Initiative (ADNI) database. As described in Szefer et al. (2017), the estimated heritability of late-onset AD is 60 - 80 percent (Gatz et al., 2006).The largest susceptibility allele is the $\epsilon 4$ allele of the Apolipoprotein E (APOE) gene (Corder et al., 1993), which may play a role in 20 to 25 percent of AD cases. The remaining heritability of AD may be explained by many additional genetic variants and these may have a small effect.

Data analysis within this setting can range from studies considering a specific candidate region of interest (ROI) within the brain and a specific candidate genetic marker in the simplest case, to massive brain-wide genome-wide analyses in the most challenging case. In our work, we consider the intermediary setting where interest lies in assessing the association between a moderate number of brain imaging phentoypes, eg. 111 ROIs in Vounou et al. (2010); 12 ROIs in Wang et al. (2012); 93 ROIs in Zhu et al. (2014); 56 ROIs in Greenlaw et al. (2017) and with the number of SNPs ranging from between a few hundred to a few thousand. Within this setting a multivariate model with regression matrix jointly characterizing the associations between all ROIs and genetic markers

is feasible; although, as detailed in the aforementioned references, we still face a challenging multivariate potentially high-dimensional regression problem.

Greenlaw et al. (2017) propose a Bayesian group sparse multi-task regression model where the primary focus is the use of a shrinkage prior based on a product of multivariate Laplace kernels developed following the ideas of Park and Casella (2008) and Kyung et al. (2010). The specific prior developed is motivated by the penalized multi-task regression estimator proposed by Wang et al. (2012). This development is an effort to move from point estimation to Bayesian credible intervals and fully Bayesian inference.

While these authors demonstrate the advantage of characterizing posterior uncertainty in their imaging genetics application to the ADNI study, their model makes a simplifying assumption for the covariance matrix of the imaging phenotypes, where the first level of the model assumes:

$$\boldsymbol{y}_\ell \,|\boldsymbol{W}, \sigma^2 \overset{ind}{\sim} MVN_c(\boldsymbol{W}^{\mathrm{T}}\boldsymbol{x}_\ell \,,\, \sigma^2 I_c), \quad \ell = 1, \ldots, n, \tag{1}$$

where $\boldsymbol{y}_\ell = (\boldsymbol{y}_{\ell 1}, \ldots, \boldsymbol{y}_{\ell c})^T$ denotes the vector of imaging phenotypes for subject ℓ and c is the dimension of the imaging phenotype, where $\ell = 1, \ldots, n$; $\boldsymbol{A}^T$ denotes transpose of matrix $\boldsymbol{A}$, $\boldsymbol{W}$ is the regression matrix; $\boldsymbol{x}_\ell = (\boldsymbol{x}_{\ell 1}, \ldots, \boldsymbol{x}_{\ell d})^T$, where $\boldsymbol{x}_\ell$ denotes the vector of genetic markers for subject ℓ and d is the number of such markers. The assumed covariance structure ignores spatial correlation as well as bilateral correlation across brain hemispheres. By the latter we mean correlation in similar structures on opposite hemispheres of the brain (e.g., a priori we expect the volume of the right hippocampus to be correlated with the volume of the left hippocampus).

We develop a new model that allows for this type of correlation by adopting a proper bivariate conditional autoregressive (BCAR) process (Gelfand and Vounatsou, 2003; Jin et al., 2005) for the errors in the regression model. While spatial models for functional magnetic resonance imaging (fMRI) and other neuroimaging modalities have been developed to a large extent (Penny et al., 2005; Bowman, 2005; Bowman et al., 2008; Derado et al., 2013; Teng et al., 2018, 2019), to our knowledge there has been very little development of explicitly spatial models for imaging genetics. One exception is the mixture model developed by Stingo et al. (2013) where an Ising prior, a binary Markov random field, is used for Bayesian variable selection. Our model is rather different in both its aims and structure as it is based on a continuous bivariate Markov random field that is specified at the first level of the model for the imaging phenotype directly.

In Figure 3.1 we show several summaries of the data from our motivating application demonstrating the need to account for correlation across brain hemispheres. For example, the sample correlation between the volume of the right cerebral cortex and the volume of the left cerebral cortex is 0.90, and across all 28 pairs of measurements, the median left/right correlation between corresponding phenotypes is approximately 0.8. The bivariate CAR structure allows us to account for this between-hemisphere correlation while also allowing us to account for within-hemisphere correlation using a graph structure based on a neighbourhood matrix.

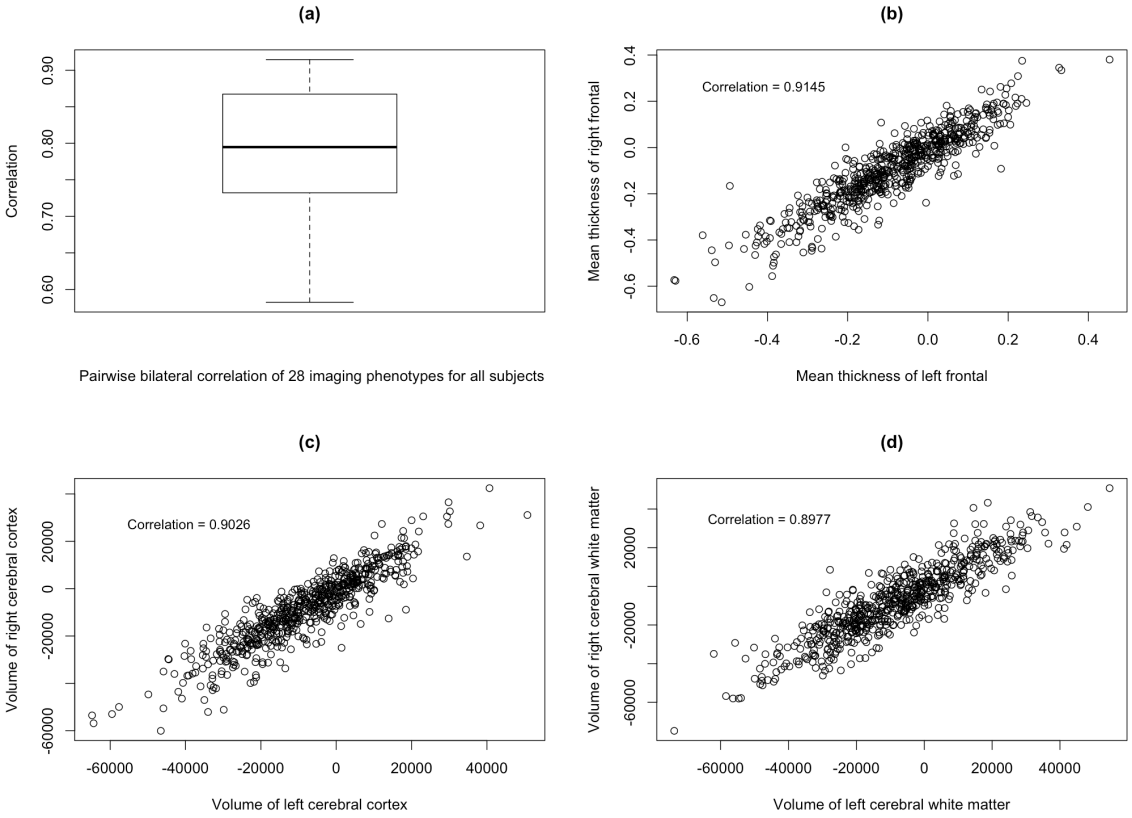


Figure 3.1: The correlation plot for left and right brain measures showing the bilateral correlation between 28 brain measurements across brain hemispheres for 632 subjects. Panel (a) presents the boxplot of all of the pairwise bilateral correlations for the 28 brain measurements across left and right hemispheres. Panel (b) presents a scatter plot showing mean thickness of left/right frontal with correlation 0.91. Panel (c) presents a scatter plot showing the volume of left/right cerebral cortex with correlation 0.90. Panel (d) presents a scatter plot between volume of left/right cerebral white matter with correlation 0.90.

Typically, models incorporating multivariate CAR specifications are used for modelling observations (in the case of a proper CAR model) or spatially-varying parameters when multiple observations or parameters appear at each spatial site. For our application the use of this process is non-standard in the sense that we do not model multiple observations at each site, but rather, we pair corresponding observations on opposite hemispheres of the brain and use the bivariate spatial process to model a combination of the bilateral correlation across the left and right brain hemispheres as well as potential correlation within each hemisphere. As a matter of fact for the MRI data considered in our application the bilateral correlation is a very strong signal in the observed data and so it is important to account for it.

For the bivariate spatial model we use a separable BCAR process as it is reasonable in our application to assume (as it might be in other neuroimaging studies) that the spatial covariance on the two hemispheres of the brain is similar. Non-separable multivariate spatial models (Gelfand and

Banerjee, 2010; MacNab, 2016) could be adopted for more flexibility allowing the spatial structure on the two hemispheres to be different; however, we do not expect that this additional flexibility would be useful in the current context. This spatial process is combined with a group LASSO prior for the regression coefficients, where each group corresponds to a single row of $\boldsymbol{W}$. Each row in this case represents the associations between a given SNP and all of the phenotypes. We employ a bivariate Gaussian scale mixture representation of a group LASSO prior in order to facilitate Bayesian computation.

To compute the posterior distribution we develop two algorithms, both of which are implemented in our R package *bgsmtr* for imaging genetics regression modelling. The package is available for download on the Comprehensive R Archive Network (CRAN). The first algorithm is a Gibbs sampling algorithm and the second is a faster mean-field variational Bayes (VB) approximation to the posterior distribution (Ormerod and Wand, 2010). Within the context of hierarchical models for spatial data, mean-field VB inference has been considered by Ren et al. (2011) who make comparisons with inference from MCMC within the context of spatial process models. In addition to the computation of the posterior distribution, the *bgsmtr* package now incorporates Bayesian FDR procedures (Morris et al., 2008) for SNP selection. This can be used alongside or as an alternative to SNP selection based on credible intervals.

The overall contribution of our work is four-fold. First, we develop an explicitly spatial model for imaging genetics based on the BCAR process. Second, we develop both an MCMC algorithm and a mean-field VB algorithm for approximating the posterior distribution. Third, we incorporate Bayesian FDR procedures for SNP selection within the new spatial model. Fourth, our new developments are implemented in the latest version of the *bgsmtr* R package that is available for download on CRAN.

The remainder of this chapter is structured as follows. In Section 2, we present our new spatial model for imaging genetics. Computation of the posterior distribution and SNP selection is discussed in Section 3. Section 4 presents a simulation study evaluating the performance of the spatial model relative to a non-spatial model and inference based on MCMC relative to that from VB. In Section 5 we apply our new model to our motivating application examining data from the ADNI-1 study, examining 56 structural brain imaging phenotypes, 486 SNPs from 33 genes, and 632 subjects. The chapter concludes with a discussion in Section 6.

3.2 Bayesian Spatial Regression Model

Let $\boldsymbol{y}_\ell = (\boldsymbol{y}_{\ell,1}, \ldots, \boldsymbol{y}_{\ell,c})^T$ and $\boldsymbol{x}_\ell = (\boldsymbol{x}_{\ell,1}, \ldots, \boldsymbol{x}_{\ell,d})^T$ denote the imaging measures at c ROIs and the genetic data respectively for subject ℓ, $\ell = 1, \ldots, n$, where $\boldsymbol{x}_{\ell,j} \in \{0, 1, 2\}$ represents the number of minor alleles of the j_{th} SNP for subject ℓ. The regression model takes the form $E(\boldsymbol{y}_{\boldsymbol{\ell}}) = \boldsymbol{W}^T \boldsymbol{x}_\ell,\ \ell = 1, \ldots, n$, where $\boldsymbol{W}$ has dimensions $d \times c$ and $W_{i,j}$ represents the association between the i_{th} SNP and the j_{th} imaging phenotype.

Our model is developed for settings where the imaging data are symmetric with the same measures collected on each hemisphere of the brain. This is true when the neuroimaging data are

considered at the voxel level and it is also the case for the study considered here where we analyze MRI data from the ADNI-1 database preprocessed using the FreeSurfer V4 software (Fischl, 2012). The FreeSurfer software is used to conduct automated parcellation to define volumetric and cortical thickness values from the 28 ROIs considered in Shen et al. (2010); Szefer et al. (2017); Greenlaw et al. (2017) on each hemisphere leading to $c = 56$ brain measures in total.

As described in Szefer et al. (2017), potential confounders in the analysis are population stratification and APOE genotype. Since true population structure is not observed, a set of principal coordinates from multidimensional scaling are used to derive proxy variables for population stratification in the data. We also adjust for APOE genotype, since it can account for the population stratification in the data, over and above the principal components or principal coordinates (Lucotte et al., 1997).

The response imaging measures at each brain ROI are first adjusted for the ten principal coordinates, as well as for dummy variables representing APOE genotype, using weighted ordinary least squares regression. The residuals from each regression are then used as the adjusted neuroimaging phenotypes (Szefer et al., 2017).

Let $\boldsymbol{y}_{\ell,i} = (y_{l,i}^{(L)}, y_{l,i}^{(R)})'$ be the brain summary measures obtained at the i_{th} ROI in the left hemisphere (L) and the right hemisphere (R). Then $\boldsymbol{y}_\ell = (\boldsymbol{y}'_{\ell,1}, \ldots, \boldsymbol{y}'_{\ell,c/2})'$ is the imaging data ordered so that left-right imaging phenotype pairs are adjacent in the response vector. There are thus $c/2$ ROIs on each hemisphere and we let $\boldsymbol{A}$ denote a $c/2 \times c/2$ symmetric neighborhood matrix which in the simplest case can have binary elements, where $A_{i,j} = 1$ indicates that ROI i and j are neighbors $i \neq j$, or more generally $A_{i,j} \geq 0$ and $A_{i,i} = 0$, $i = 1, \ldots, c/2$. A user input is thus a neighborhood matrix $\boldsymbol{A}$ or in the absence of user input our software implementation takes $A_{i,j}$ to be the average of the absolute value of the sample correlation between phenotype/ROI i and phenotype/ROI j, where the average is taken over left/right hemisphere. The regression model then takes the form

$$\boldsymbol{y}_\ell = \boldsymbol{W}^T \boldsymbol{x}_\ell + \boldsymbol{\epsilon}_\ell. \tag{2}$$

The model for the errors $\boldsymbol{\epsilon}_\ell$ is a mean-zero multivariate normal distribution of dimension c, which can be specified through a set of $c/2$ compatible bivariate conditional distributions for $\boldsymbol{\epsilon}_{l,i} = (\epsilon_{l,i}^{(L)}, \epsilon_{l,i}^{(R)})'$, specified as follows:

$$\boldsymbol{\epsilon}_{l,i} | \boldsymbol{\epsilon}_{l(-i)}, \rho, \boldsymbol{\Sigma} \sim \text{BVN}\left(\frac{\rho}{A_{i.}} \sum_{j=1}^{c/2} A_{i,j} \boldsymbol{\epsilon_{l,j}}, \frac{1}{A_{i.}} \boldsymbol{\Sigma}\right),$$

where $\boldsymbol{\epsilon_{l(-i)}}$ denotes the rest after removing $\boldsymbol{\epsilon}_{l,i}$ from $\boldsymbol{\epsilon}_l$, $A_{i.} = \sum_{j=1}^{\frac{c}{2}} A_{i,j}$, $\rho \in [0, 1)$ characterizes spatial dependence with $\rho = 0$ corresponding to independence across all ROI pairs and $\boldsymbol{\Sigma}$ is a 2×2 matrix where $\kappa = \Sigma_{12} / \sqrt{\Sigma_{11}\Sigma_{22}} \in (-1, 1)$ quantifies within pair dependence, and with $\kappa = 0$ corresponding to independence within ROI pairs. We reiterate that the model allows for correlation across brain hemispheres but it also has the case of independence as a special case.

As far as we are aware, this spatial model for neuroimaging data is one of the first to explicitly model dependence across brain hemispheres in addition to accounting for local dependence. Often,

this left/right bilateral dependence is ignored with neuroimaging data. In our data it is a very clear and strong signal as is evident in Figure 3.1. As the parameter Σ is free in the model it can be informed by the data. Therefore, we expect that the posterior will reflect some degree of between hemisphere correlation when it is present to a sufficient degree, and will remain at roughly a diagonal form when it is not, in diseases that have major differences across hemispheres, for example. The prior for Σ is chosen so that it is centered on a diagonal matrix.

Under this new specification the first level of the regression model takes the following form:

$$\boldsymbol{y}_\ell | \boldsymbol{W}, \boldsymbol{\Sigma} \overset{ind}{\sim} \text{MVN}_c(\boldsymbol{W}^T \boldsymbol{x}_\ell, (\boldsymbol{D_A} - \rho \boldsymbol{A})^{-1} \otimes \boldsymbol{\Sigma}),\ l = 1, \ldots, n, \tag{3}$$

where MVN_c denotes a c-dimensional multivariate normal distribution, $\otimes$ is the kronecker product, $\boldsymbol{D_A} = \text{diag}\{A_{i.}, i = 1, ..., c/2\}$ and as before $A_{i.} = \sum_{j=1}^{\frac{c}{2}} A_{i,j}$. For the regression coefficients, we let $\tilde{W}_{i,j^*} = (W_{i,j}, W_{i,j+1}),\ \ j = 2j^* - 1,\ \ j^* = 1, ..., \frac{c}{2}$, and we adopt a shrinkage prior based on a bivariate Gaussian scale mixture

$$\tilde{W}_{i,j^*} | \omega_i^2, \Sigma \overset{ind}{\sim} \text{BVN}(\boldsymbol{0}, \omega_i^2 \Sigma),$$

$$\omega_i^2 | \lambda^2 \overset{iid}{\sim} \text{Gamma}(\frac{c+1}{2}, \lambda^2/2),$$

$$\Sigma \sim \text{Inv-Wishart}(v, \boldsymbol{S}),$$

where ρ and λ^2 are tuning parameters controlling spatial dependence and regression sparsity respectively. Tuning of the model is discussed in Section 3.3 after our discussion of computational algorithms for fitting the model. The remaining hyperparameters v and $\boldsymbol{S}$ are set at $v = 2$ and $\boldsymbol{S} = \boldsymbol{I}$ to yield a prior that is somewhat vague, and they can be varied as part of a sensitivity analysis.

3.3 Computation and SNP Selection

3.3.1 Bayesian Computation

Bayesian inference for our proposed model is based on the posterior distribution $P(\boldsymbol{\Theta}|\boldsymbol{Y})$, where $\boldsymbol{\Theta} = \{\boldsymbol{W}, \boldsymbol{\Sigma}, \boldsymbol{\omega^2}\}$, $\boldsymbol{\omega}^2 = (\omega_1^2, \cdots, \omega_{\text{d}}^2)$ and $\boldsymbol{Y}$ denotes the imaging data for all n subjects. Posterior computation can be implemented using Gibbs sampling. The update steps for this algorithm are listed in Algorithm 1 and their derivations are given in the Supplementary Material (Appendix B).

As a faster alternative approach to computing the posterior distribution, we also develop a mean-field VB algorithm. As opposed to Monte Carlo sampling, VB inference is based on solving an optimization problem. The approximation $q(\boldsymbol{\theta})$ to the posterior distribution $P(\boldsymbol{\theta}|\boldsymbol{Y})$ is based on constructing and optimizing a lower bound on the marginal likelihood $P(\boldsymbol{Y})$.

Assuming that $q(\boldsymbol{\theta})$ has the same support as $P(\boldsymbol{\theta}|\boldsymbol{Y})$, the log-marginal likelihood can be written as $\log P(\boldsymbol{Y})$

$$
\begin{aligned}
&= \int q(\boldsymbol{\theta}) \log\{\frac{P(\boldsymbol{Y}, \boldsymbol{\theta})}{q(\boldsymbol{\theta})}\} d\boldsymbol{\theta} + \int q(\boldsymbol{\theta}) \log\{\frac{q(\boldsymbol{\theta})}{P(\boldsymbol{\theta}|\boldsymbol{Y})}\} d\boldsymbol{\theta} \\
&= E_q[\log\{\frac{P(\boldsymbol{\theta}, \boldsymbol{Y})}{q(\boldsymbol{\theta})}\}] + E_q[\log\{\frac{q(\boldsymbol{\theta})}{P(\boldsymbol{\theta}|\boldsymbol{Y})}\}] \\
&= \mathfrak{F}(q, \boldsymbol{Y}) + KL(q||p) \geq \mathfrak{F}(q, \boldsymbol{Y}).
\end{aligned}
$$

Algorithm 3 Gibbs Sampling Algorithm

1. Set tuning parameters λ^2 and ρ.
2. Initialize $\boldsymbol{W}, \boldsymbol{\Sigma}, \boldsymbol{\omega^2}$ and repeat steps (3) - (5) below to obtain the desired Monte Carlo sample size after burn-in.
3. For $i = 1, ..., d$, update $\boldsymbol{W}_{(i)}^T$ according Equation 9.
4. Update $\boldsymbol{\Sigma}$ using Equation 10.
5. For $i = 1, \ldots, d$ update ω_i^2 through Equation 11.

Here, $KL(q||p)$ denotes the Kullback-Leibler divergence from q to p and the final inequality is true since $KL(q||p) \geq 0$. The approximation to $P(\boldsymbol{\theta}|\boldsymbol{Y})$ by $q(\boldsymbol{\theta})$ is obtained by restricting $q(\boldsymbol{\theta})$ to a manageable class of distributions and maximizing the lower bound $\mathfrak{F}(q, \boldsymbol{Y})$ (which is equivalent to minimizing $KL(q||p)$) over that class. The functional $\mathfrak{F}(q, \boldsymbol{Y})$ is referred to as the evidence lower bound (ELBO). In the case of mean-field VB, the restriction of $q(\boldsymbol{\theta})$ is to a product form $q(\boldsymbol{\theta}) = \prod_{j=1}^{J} q_j(\boldsymbol{\theta}_j)$. In the specific context of our model the assumed mean-field approximation is as follows

$$
P(\boldsymbol{\Theta}|Y) \approx \Big[\prod_{i=1}^{d} q(\boldsymbol{W}_{(i)}) q(\omega_i^2) \Big] q(\boldsymbol{\Sigma}), \tag{4}
$$

where $\boldsymbol{W}_{(i)}$ is the i_{th} row $\boldsymbol{W}$.

We maximize the functional $\mathfrak{F}(q_1, \ldots, q_J, \boldsymbol{Y})$ over the q_j's using a coordinate ascent procedure. The update steps for this procedure take the form (Ormerod and Wand, 2010)

$$
q_i(\boldsymbol{\theta}_i) = \frac{\exp\{E_{\theta_{-i}}[\log P(\boldsymbol{\theta}_i|\boldsymbol{Y}, \boldsymbol{\theta}_{-i})]\}}{\int \exp\{E_{\theta_{-i}}[\log P(\boldsymbol{\theta}_i|\boldsymbol{Y}, \boldsymbol{\theta}_{-i})]\} d\boldsymbol{\theta}_i},
$$

where the expectation is taken with respect to $q_{-i}(\boldsymbol{\theta}_{-i}) = \prod_{l \neq i} q_l(\boldsymbol{\theta}_l)$. This leads to a set of update equations related to the EM algorithm (Beal et al., 2003) that are iterated until convergence to a local optimum. These update equations are presented in Algorithm 2 and their derivations are detailed in the Supplementary Material (Appendix B). On convergence, the approximation to the posterior

Algorithm 4 Mean-field Variational Bayes Algorithm

1. Set tuning parameters λ^2 and ρ.

2. Initialize $q_{(\boldsymbol{W})}, q_{(\boldsymbol{\Sigma})}, q_{(\boldsymbol{\omega^2})}$ and cycle through steps (3) - (5) below until the increase in the lower bound $\mathcal{L}(q)$ is negligible.

3. For $i = 1, ..., d$, update

$$\boldsymbol{\Sigma}_{q(\boldsymbol{W}_{(i)})} \leftarrow \left(\left[\mu_q(\eta_i) \otimes I_{\frac{c}{2}} \otimes (v_{q(\boldsymbol{\Sigma})} S^{-1}_{q(\boldsymbol{\Sigma})}) \right] + \sum_{\ell=1}^{\mathrm{n}} (\boldsymbol{x}_{\ell(i)} \otimes I_c)[(D_A - \rho A) \otimes (v_{q(\boldsymbol{\Sigma})} S^{-1}_{q(\boldsymbol{\Sigma})})](\boldsymbol{x}^T_{\ell(i)} \otimes I_c) \right)^{-1},$$

$$\boldsymbol{\mu}_{q(\boldsymbol{W}_{(i)})} \leftarrow \boldsymbol{\Sigma}_{q(\boldsymbol{W}_{(i)})} \Big(- \sum_{\ell=1}^{\mathrm{n}} (\boldsymbol{x}_{\ell(i)} \otimes I_c)[(D_A - \rho A) \otimes (v_{q(\boldsymbol{\Sigma})} S^{-1}_{q(\boldsymbol{\Sigma})})](\boldsymbol{x}^T_{\ell(-i)} \otimes I_c)(\boldsymbol{\mu}_{q(\boldsymbol{W}_{(-i)})})$$
$$+ \sum_{\ell=1}^{\mathrm{n}} (\boldsymbol{x}_{\ell(i)} \otimes I_c)[(D_A - \rho A) \otimes (v_{q(\boldsymbol{\Sigma})} S^{-1}_{q(\boldsymbol{\Sigma})})] \boldsymbol{y}_\ell \Big).$$

4. Update $S_{q(\boldsymbol{\Sigma})}$ as

$$S_{q(\boldsymbol{\Sigma})} \leftarrow \sum_{l=1}^{n} \sum_{i=1}^{\frac{c}{2}} \sum_{j=1}^{\frac{c}{2}} b_{i,j} \tilde{y}^*_{l,i} \tilde{y}^{*\,T}_{l,i} + \sum_{i=1}^{d} \sum_{j^*=1}^{\frac{c}{2}} E_q(\tilde{W}_{i,j^*} \tilde{W}^T_{i,j^*}) \mu_{q(\eta_i)} + S,$$

   where:

$$b_{i,j} = [D_A - \rho A]_{i,j}\,.$$

5. For $i = 1, ..., d$, update $\mu_{q(\eta_i)}$,

$$\mu_{q(\eta_i)} \leftarrow \sqrt{\frac{\lambda^2}{E_q(c^*_i)}},$$

   where:

$$E_q(c^*_i) = E_q \left(tr(\sum_{j^*=1}^{\frac{c}{2}} \tilde{W}_{i,j^*} \tilde{W}^T_{i,j^*} \Sigma^{-1}) \right).$$

   Update:

$$\mu_{q(\omega^2_i)} \leftarrow \frac{1}{\mu_{q(\eta_i)}} + \frac{1}{\lambda^2},$$
$$Var_{q(\omega^2_i)} \leftarrow \frac{1}{\mu_{q(\eta_i)} \lambda^2} + \frac{2}{(\lambda^2)^2}.$$

---

distribution is based on (4) as well as the solutions

$$q(\boldsymbol{W}_{(i)}) \equiv \text{MVN}(\boldsymbol{\mu}_{q(\boldsymbol{W}_{(i)})}, \boldsymbol{\Sigma}_{q(\boldsymbol{W}_{(i)})}),\ i = 1, \ldots, d,$$
$$q(\omega^2_i) \equiv \text{Reciprocal Inverse Gaussian}(\mu_{q(\eta_i)}, \lambda_{q(\eta_i)}),\ i = 1, \ldots, d,$$
$$q(\boldsymbol{\Sigma}) \equiv \text{Inverse-Wishart}(S_{q(\boldsymbol{\Sigma})}, v_{q(\boldsymbol{\Sigma})}),$$

where the statistics $\{\boldsymbol{\mu}_{q(\mathbf{W}_{(i)})}, \boldsymbol{\Sigma}_{q(\mathbf{W}_{(i)})},\ i = 1, \ldots, d\}$; $\{\mu_{q(\eta_i)}, \lambda_{q(\eta_i)}, i = 1, \ldots, d\}$; $S_{q(\boldsymbol{\Sigma})}$, $\nu_{q(\boldsymbol{\Sigma})}$, also referred to as variational parameters, are obtained as the output of the iterative Algorithm 2.

To initialize the variational Bayes algorithm, we use a ridge regression estimator obtained separately for each column of $\boldsymbol{W}$ obtained by fitting ridge regression with individual scalar-valued phenotypes as the response. The ridge estimators are then used to initialize the mean of the variational posterior distribution. The output of variational Bayes is then used to initialize the MCMC sampling algorithm.

3.3.2 Bayesian FDR

The Bayesian FDR procedure applied in our work for SNP selection follows the approach developed in Morris et al. (2008), but it has been adapted and implemented for the current spatial model. We assume that we have N samples $W_{i,j}^{(1)}, \ldots, W_{i,j}^{(N)}$ from the posterior distribution for each of the regression coefficients $W_{i,j}$, $i = 1, \cdots, d$, $j = 1, \cdots, c$. Let c^* be a known critical value that is chosen a priori to represent an effect size of interest. Given this value, we compute a posterior tail probability for the i-th SNP at region j as $p_{i,j} = Pr(|W_{i,j}| > c^*|\mathbf{Y}), i = 1, \ldots, d; j = 1, \ldots, c$, which can be approximated by $p_{i,j} \approx N^{-1} \sum_{i^*=1}^{N} I\left\{|W_{i,j}^{(i^*)}| > c^*\right\}$ and we replace any $p_{i,j} = 1$ with $1 - (2N)^{-1}$. Given these posterior tail probabilities and a desired global FDR-bound α, we denote by ϕ_α the corresponding threshold chosen so that a SNP-region pair (i, j) is selected if $p_{i,j} > \phi_\alpha$. The cut-off ϕ_α can be computed by sorting $\left\{p_{i,j}, i = 1, \cdots, d; j = 1, \ldots, c\right\}$ in descending order $\{p(i), i = 1, \cdots, d \times c\}$, then $\phi_\alpha = p(\lambda)$, with $\lambda = \max\left\{l^* : (l^*)^{-1} \sum_{l=1}^{l^*} (1 - p(l)) \leq \alpha\right\}$. The threshold ϕ_α is a cutpoint on the posterior probabilities that controls the expected Bayesian FDR below level α. The value of c^* can be chosen based on prior knowledge of what constitutes an effect size of interest, or in the absence of such knowledge, it can be chosen based on the data.

3.3.3 Model Selection and Tuning

To compare the spatial and non-spatial models and to choose values for the tuning parameters, one option is the use of the WAIC (Vehtari et al., 2017). This criterion can be computed from posterior simulation output and takes the form

$$WAIC = -2 \sum_{l=1}^{n} \log E_{\boldsymbol{W},\boldsymbol{\Sigma}}[p(\boldsymbol{y}_\ell \,|\, \boldsymbol{W}, \boldsymbol{\Sigma})|\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n] + 2 \sum_{l=1}^{n} VAR_{\boldsymbol{W},\boldsymbol{\Sigma}}[\log p(\boldsymbol{y}_\ell \,|\, \boldsymbol{W}, \boldsymbol{\Sigma})|\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n],$$

where $p(\boldsymbol{y}_\ell \,|\, \boldsymbol{W}, \boldsymbol{\Sigma})$ is the multivariate normal density function associated with the conditional autoregressive model (3), and the expectation and variance are taken with respect to the posterior distribution, with lower values being preferred.

An alternative more convenient approach to tuning the model is to use a simple modified moment estimator based on the ridge regression estimator that we use to initialize VB. The estimator takes

the form

$$\lambda^2 = \frac{dc(c+1)}{\max\{1, \nu - 3\}} \left(\sum_{i,j} \hat{\boldsymbol{W}}^2_{R_{i,j}} \right)^{-1},$$

and its form is derived in Appendix C.

For VB we use the moment estimate for λ^2 and fix the value of $\rho = 0.95$ corresponding to a reasonable degree of spatial dependence on the graph represented by the neighbourhood matrix $\boldsymbol{A}$. The MCMC algorithm is then initialized using the output of VB and the value of λ^2 can be set at the moment estimate while ρ can be chosen using the WAIC. The WAIC can also be used to compare the spatial model with the non-spatial model.

While the WAIC can be used for selection of ρ, we have found that selection of λ^2 using the WAIC can result in patterns where WAIC decreases in a monotone fashion as λ^2 increases. In these situations, we recommend that either the modified moment estimate is used or that the results be summarized over a range of values for λ^2. One example of the latter is to plot the regularization path which shows, for each region, the value of the posterior mean corresponding to each SNP as a function of varying λ^2. This is demonstrated in our application.

3.4 Simulation Study

We conduct a simulation study in which the dimension of the data is the same as that in our motivating application with 56 structural brain imaging phenotypes, 486 SNPs from 33 genes, and 632 subjects. The data are simulated from the spatial model with parameter values set at the estimates obtained from the real data (the estimates from the MCMC algorithm). We use this setting as it is the most realistic choice of parameter values. Thus, while we are simulating from the model we are simulating from it under a setting that aligns realistically with the data from our application.

We use 100 simulation replicates and run the MCMC algorithms for both the spatial model and the non-spatial model for 10000 iterations with 5000 iterations used as burn-in for each replicate, and we run the variational Bayes algorithm to convergence of the ELBO, where convergence is declared when the absolute relative change in the ELBO is smaller than 10^{-4} for two consecutive iterations. The tuning parameters are chosen using the WAIC over a coarse grid for the spatial model implemented via MCMC; set using the default approach in the R package for the original non-spatial model of Greenlaw et al. (based on five-fold cross-validation over a grid of possible values with out-of-sample prediction based on an approximate posterior mode, the estimator of Wang et al. (2012), that can be computed quickly) and using our moment based approach to obtain λ^2 for VB, while the spatial parameter is held fixed at the default value of $\rho = 0.95$ for VB (the true value set when simulating the data is $\rho = 0.8$).

Three measures are used for comparison in this study. For the best overall performance, the spatial model implemented using MCMC has an average mean-squared error (MSE) of 0.0055, followed by the non-spatial model with an average MSE of 0.012 and finally the spatial model implemented with VB with an average MSE of 0.043. Here, the average is taken over the 27,216 regression coefficients

in the model. Next, we examine the correlation between the posterior mean estimates of vec($\boldsymbol{W}$) and the true values averaged over simulation replicates. The spatial model implemented using MCMC has the best performance (0.69), followed by the non-spatial model implemented via MCMC (0.67) and then the spatial model implemented using VB (0.65). The average-squared bias (where the average is taken over the regression coefficients in the model) is lowest for the non-spatial model (3.29×10^{-4}), followed by the spatial model implemented with MCMC (4.70×10^{-3}) and finally the spatial model implemented using VB (9.10×10^{-3}) has the highest average-squared bias. Both the non-spatial and spatial models appear to exhibit adequate coverage probability overall when implemented with MCMC while the variational Bayes implementation of the spatial model exhibits coverage of 95% equal-tail credible intervals that is slightly lower at 0.91. The average posterior standard deviation obtained from the spatial model (MCMC) is slightly larger than that obtained from the non-spatial model perhaps indicating that the spatial model is adequately accounting for the dependence. As is typical with variational Bayes the posterior standard deviation is underestimated. This arises from the Kullback-Leibler objective function $KL(q||p)$ for variational Bayes which under-penalizes approximations that are under-dispersed.

To evaluate the empirical FDR of the Bayesian FDR approach when the expected Bayesian FDR is controlled with $\alpha = 0.05$, leading to the threshold for posterior probabilities $\phi_\alpha = 0.05$, we conduct an additional 100 simulation replicates where the error structure underlying the simulated data is again set to that obtained from the real data application. All but 100 SNPs have their 56 coefficients set to exactly zero while 50 rows of W have coefficients set to $W_{ij} = 1$, 25 rows have coefficients set to $W_{ij} = 2$ and the remaining 25 rows have coefficients set to $W_{ij} = 3$. Here we evaluate the empirical FDR for the spatial model and both of its implementations. The methodology of Greenlaw et al. (2017) uses credible intervals rather than Bayesian FDR for SNP selection. Both the VB and MCMC algorithms use the moment estimate for λ^2. The empirical proportion of false discoveries averaged over simulation replicates is reported in Supplementary Material (Appendix D) for both VB and MCMC as a function of c^*. We note that these c^*. values are based on each simulated response matrix $\mathbb{Y}$ having its columns centered and scaled. The empirical FDR appears to drop to 0 with increasing c^*. faster with the MCMC implementation which is consistent with the VB implementation being more liberal in selecting SNPs.

3.5 ADNI-1 Study of MRI and Genetics

We apply our spatial model as well as the group sparse multi-task regression model of Greenlaw et al. (2017) to MRI and genetic data collected from $n = 632$ subjects from the ADNI-1 database. The ADNI was launched in 2003 as a public-private partnership, led by Principal Investigator Michael W. Weiner, MD. The primary goal of ADNI has been to test whether serial magnetic resonance imaging (MRI), positron emission tomography (PET), other biological markers, and clinical and neuropsychological assessment can be combined to measure the progression of mild cognitive impairment (MCI) and early Alzheimer's disease (AD). For up-to-date information, see

www.adni-info.org. ADNI is an ongoing, longitudinal, multicenter study designed to develop clinical, imaging, genetic, and biochemical biomarkers for the early detection and tracking of AD.

The genetic data comprise SNPs belonging to the top 40 Alzheimer's Disease (AD) candidate genes listed on the AlzGene database as of June 10, 2010. The data presented here are queried from the genome build as of December 2014, from the ADNI-1 data. After quality control and imputation steps, the genetic data used for this study include 486 SNPs from the 33 targeted genes discussed in Szefer et al. (2017). The freely available software package PLINK (Purcell et al., 2007) is used for genomic quality control. Subjects are included if their genotyping data is available, they have a baseline MRI scan, and they have at least one additional follow-up baseline scan. Among all SNPs, only SNPs belonging to the top 40 AD candidate genes listed on the AlzGene database (www.alzgene.org) as of June 10, 2010, are selected after the standard quality control (QC) and imputation steps. The QC criteria for the SNP data include (i) call rate check per subject and per SNP marker, (ii) gender check, (iii) sibling pair identification, (iv) the Hardy-Weinberg equilibrium test, (v) marker removal by the minor allele frequency and (vi) population stratification. Our thresholds for SNP and subject exclusion are the same as in Wang et al. (2012) with three exceptions. In the data quality control step, we used a stricter minimum call rate of 95% on SNPs vs. Wang et al.'s call rate of 90%. To assign SNPs to genes, we use a genome build (Build GRCh38.p2) from December 2014 whereas these authors use a genome build (Build 36.2) from September 2006, and use all subjects with a baseline measurement whereas we choose subjects with a baseline MRI scan and a scan at at least one additional time point in the longitudinal study.

The response measures are obtained by preprocessing the MRI data using the FreeSurfer V4 software which conducts automated parcellation to define volumetric and cortical thickness values from the 28 ROIs considered in Szefer et al. (2017) and Greenlaw et al. (2017) on each hemisphere of the brain, leading to $c = 56$ brain measures in total. These ROIs are chosen based on prior knowledge that they are related to Alzheimer's Disease. Each of the response variables are adjusted for age, gender, education, handedness, baseline total intracranial volume (ICV), potential population stratification and APOE genotype and centered to have zero-sample-mean and unit-sample-variance.

We fit our new spatial model to these data using both Algorithm 1 (Gibbs sampling) and Algorithm 2 (VB). In addition, we fit the non-spatial model using the MCMC sampler derived therein, with the tuning parameters for the non-spatial model set at $\lambda_1^2 = 1000$ and $\lambda_2^2 = 1000$ based on the values selected in Greenlaw et al. (2017). In all cases, MCMC sampling is run for 10,000 iterations with the initial 5,000 iterations discarded. The required computation time for the spatial model (MCMC) is 50 hours on a single core (2.66-GHz Xeon x5650) with 20GB of RAM, while the computation for the non-spatial model is 5hrs. Some trace plots and MCMC convergence diagnostics are presented in Appendix E and these demonstrate rapid convergence and good mixing of the MCMC sampling chains. The VB algorithm is run to convergence and requires 45 minutes with the ELBO converging in approximately 16 iterations. The tuning parameter λ^2 is set based on the moment estimator and we set $\rho = 0.95$ for VB. The convergence of VB based on successive values of the ELBO is depicted in Figure 3.2.

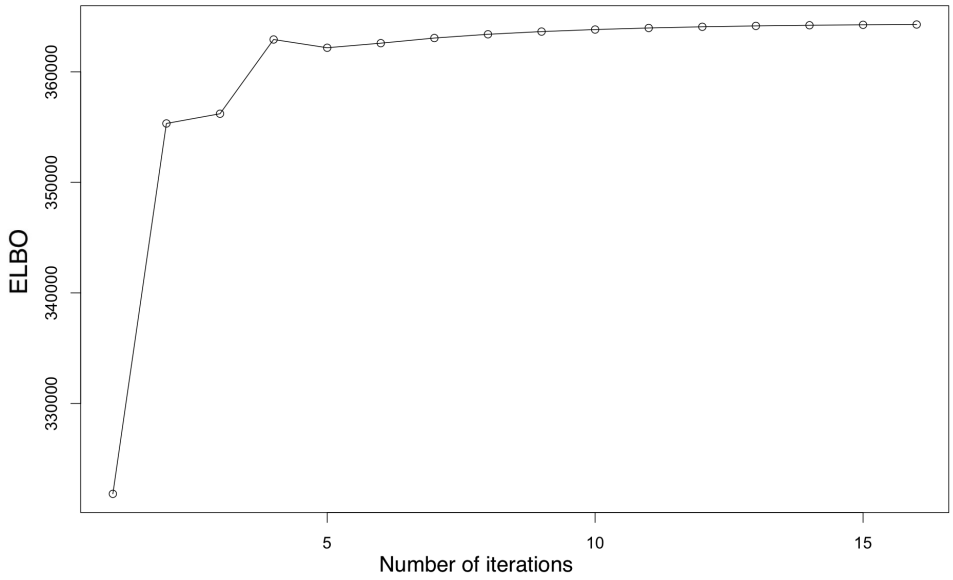


Figure 3.2: Variational Bayes - convergence of the evidence lower bound (ELBO) for the ADNI MRI and genetic data considered in the application.

Figure 3.3 presents the WAIC computed for a number of different choices of the tuning parameters ρ and λ^2. For the values of $\rho = 0.8$ and $\lambda^2 = 10,000$ for the ADNI-1 data, the value of the WAIC is 83, 170. While the WAIC is able to identify a value for ρ, it is monotone decreasing as a function of λ^2. We thus summarize the results over several values of λ^2. The WAIC obtained for the non-spatial model with value of the tuning parameters used in Greenlaw et al. (2017) ($\lambda_1^2 = 1000$ and $\lambda_2^2 = 1000$) is 108, 745.

Figure 3.4 presents the number of SNPs chosen by the spatial model for each ROI using Bayesian FDR (based on a critical value of $c^* = 0.044$) as a function of the tuning parameter λ^2 for both Gibbs sampling and VB. As expected, the curves are monotone decreasing but it is interesting to note that their shapes differ when comparing the algorithms. In particular, VB selects a larger number of SNPs at all values of λ^2. Comparing the figures obtained from MCMC and VB, the VB approximation appears to perform reasonably well in this regard, though MCMC is a gold standard because of simulation consistency guarantees (Robert and Casella, 2013). We suggest that the VB algorithm be used for obtaining starting values to initialize the MCMC as well as a tool to gain some initial insight into the data while the MCMC sampler runs to completion. This is useful because the MCMC sampler requires a relatively long run time, and the VB algorithm can be used initially (requiring 45 minutes in our study) while the MCMC sampler runs (requiring 50 hrs in our study).

For the values of the tuning parameters $\rho = 0.8$, $\lambda^2 = 10,000$, the average number of SNPs selected per ROI is 2, while more than half of the ROIs have no SNPs selected. In total, 75 SNPs across all 56 ROIs are selected and these are listed in Table 3.1 of the Supplementary Material

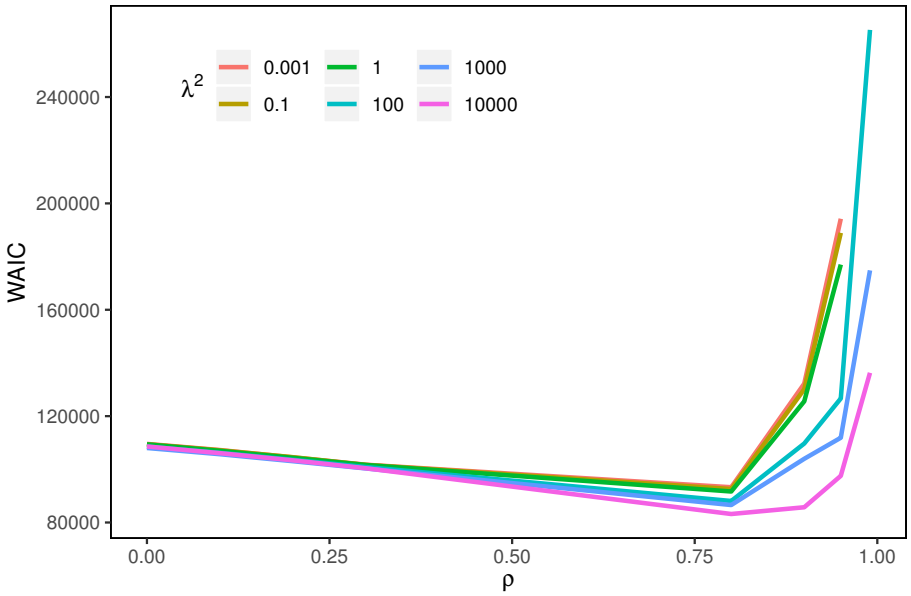


Figure 3.3: ADNI-1 Data - the WAIC for the spatial model implemented with MCMC for various values of λ^2 and ρ.

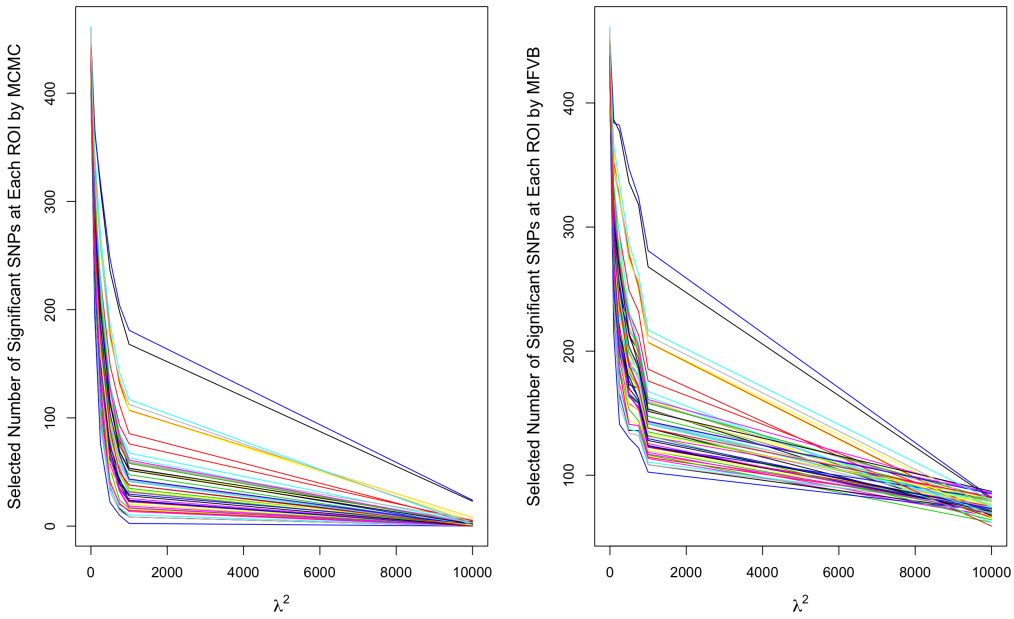


Figure 3.4: ADNI-1 Data - Relationship between the number of selected SNPs in each region and λ^2. Each region is represented with a curve in each panel of the figure. The left panel shows this relationship for MCMC combined with Bayesian FDR ($\alpha = 0.05$) while the right panel shows the same relationship for VB with Bayesian FDR ($\alpha = 0.05$).

(Appendix A) along with the corresponding phenotypes that they are associated with. With the VB approximation, 150 SNPs are selected, and the set of SNP-ROI pairs selected by MCMC is a proper subset of the set selected by VB. In addition, the subset of SNPs and phenotypes also selected by the approach of Greenlaw et al. (2017) where the marginal posterior 95% credible interval is used for SNP selection are also highlighted in bold in Table 3.1 of the Supplementary Material (Appendix A).

Considering all three approaches, a consistent signal is found at the APOE gene, where all three methods select SNP rs405509 and the Bayesian FDR procedure selects this SNP for four phenotypes, namely, the middle temporal gyrus thickness on the right hemisphere, supramarginal gyrus thickness on the right hemisphere, mean thickness of the caudal midfrontal, rostral midfrontal, superior frontal, lateral orbitofrontal, and medial orbitofrontal gyri and frontal pole on the right hemisphere, and the mean thickness of the inferior temporal, middle temporal, and superior temporal gyri on the right hemisphere. It is of interesting to note that the selected associations for this SNP all correspond to ROIs in the right brain hemisphere. The associations between the genetic signal represented in our analysis by APOE SNP rs405509 with phenotypes on the right hemisphere of the brain may be of potential interest for further investigation.

The associated point estimates and 95% equal-tail credible intervals for all 56 phenotypes and APOE SNP rs405509 are presented in Table 3.4 of the supplementary material (Appendix F) and a subset of these results for 26 phenotypes is presented in Table 3.1. Overall, there is broad agreement based on the overlap of the interval estimates, though, we also see are a number of examples where the 95% interval estimate obtained from the spatial model includes the value 0 while the corresponding interval estimate for the non-spatial model does not include zero. The higher bias arising from VB in the simulation studies is also apparent in the results presented in Table 3.1. The interval estimates arising from VB appear to be slightly more narrow than those obtained from MCMC, but not to a large degree, and this may also be reflected in the selection rates depicted in Figure 3.4 which are slightly higher.

Table 3.1: ADNI-1 Study: posterior means and 95% equal-tail credible intervals for a subset of the ROIs and their association with APOE SNP rs405509.

	Spatial Model				**Non-Spatial Model**	
	MCMC		**MFVB**		**MCMC**	
Region	Mean	95% CI	Mean	95% CI	Mean	95% CI
Amygdala volume (L)	0.09	[-0.03,0.22]	0.17	[0.08,0.28]	0.12	[0.02,0.23]
Cerebral white matter volume (L)	0.09	[-0.04,0.22]	0.19	[0.10,0.28]	0.13	[0.03,0.23]
Inferior lateral ventricle volume (L)	-0.13	[-0.24,-0.01]	-0.14	[-0.24,-0.05]	-0.08	[-0.18,0.02]

Continued on next page

Table 3.1 – *Continued from previous page*

	Spatial Model				**Non-Spatial Model**	
	MCMC		**MFVB**		**MCMC**	
Region	Mean	95% CI	Mean	95% CI	Mean	95% CI
Inferior parietal gyrus thickness (R)	0.11	[0.01,0.21]	0.20	[0.11,0.29]	0.13	[0.03,0.23]
Fusiform, parahippocampal, and lingual gyri, temporal pole and transverse temporal pole mean thickness (L)	0.12	[0.01,0.24]	0.22	[0.13,0.32]	0.14	[0.04,0.24]
Inferior temporal, middle temporal, superior temporal, fusiform, parahippocampal, and lingual gyri, temporal pole and transverse temporal pole mean thickness (L)	0.11	[0.01,0.22]	0.20	[0.11,0.28]	0.13	[0.03,0.23]
Postcentral gyrus thickness (L)	0.13	[0.03,0.24]	0.23	[0.13,0.33]	0.14	[0.03,0.23]
Superior frontal gyrus thickness (R)	0.12	[0.02,0.22]	0.19	[0.11,0.28]	0.14	[0.04,0.24]
Supramarginal gyrus thickness (L)	0.13	[0.02,0.23]	0.21	[0.13,0.29]	0.14	[0.04,0.25]
Supramarginal gyrus thickness (R)	0.14	[0.04,0.24]	0.24	[0.17,0.33]	0.15	[0.05,0.25]
Fusiform gyrus thickness (L)	0.11	[-0.02,0.23]	0.19	[0.09,0.28]	0.13	[0.03,0.23]
Hippocampus volume (L)	0.12	[-0.02,0.26]	0.22	[0.14,0.34]	0.15	[0.05,0.26]

Continued on next page

Table 3.1 – *Continued from previous page*

	Spatial Model				Non-Spatial Model	
	MCMC		MFVB		MCMC	
Region	Mean	95% CI	Mean	95% CI	Mean	95% CI
Caudal midfrontal, rostral midfrontal, superior frontal, lateral orbitofrontal, and medial orbitofrontal gyri and frontal pole mean thickness (L)	0.10	[-0.02,0.22]	0.18	[0.10,0.27]	0.13	[0.03,0.23]
Inferior temporal, middle temporal, and superior temporal gyri mean thickness (L)	0.10	[-0.01,0.21]	0.17	[0.08,0.26]	0.12	[0.02,0.22]
Inferior temporal, middle temporal, and superior temporal gyri mean thickness (R)	0.10	[-0.01,0.21]	0.20	[0.11,0.30]	0.13	[0.02,0.22]
Fusiform, parahippocampal, and lingual gyri, temporal pole and transverse temporal pole mean thickness (R)	0.10	[-0.01,0.21]	0.20	[0.11,0.29]	0.12	[0.02,0.22]
Inferior and superior parietal gyri, supramarginal gyrus, and precuneus mean thickness (L)	0.09	[-0.02,0.21]	0.16	[0.07,0.25]	0.12	[0.02,0.22]
Precentral and postcentral gyri mean thickness (R)	0.09	[-0.02,0.20]	0.17	[0.09,0.26]	0.12	[0.01,0.22]

Continued on next page

Table 3.1 – *Continued from previous page*

	Spatial Model				Non-Spatial Model	
	MCMC		MFVB		MCMC	
Region	Mean	95% CI	Mean	95% CI	Mean	95% CI
Precentral and postcentral gyri mean thickness (L)	0.10	[-0.01,0.20]	0.16	[0.08,0.25]	0.12	[0.02,0.22]
Middle temporal gyrus thickness (L)	0.08	[-0.03,0.19]	0.15	[0.06,0.23]	0.11	[0.01,0.21]
Middle temporal gyrus thickness (R)	0.09	[-0.01,0.20]	0.18	[0.11,0.27]	0.12	[0.02,0.22]
Postcentral gyrus thickness (R)	0.09	[-0.01,0.20]	0.18	[0.08,0.28]	0.11	[0.01,0.21]
Precentral gyrus thickness (R)	0.08	[-0.03,0.19]	0.15	[0.06,0.25]	0.11	[0.01,0.21]
Precuneus thickness (R)	0.09	[-0.01,0.19]	0.17	[0.08,0.25]	0.11	[0.01,0.21]
Superior parietal gyrus thickness (R)	0.08	[-0.02,0.18]	0.14	[0.06,0.22]	0.11	[0.01,0.21]
Superior temporal gyrus thickness (R)	0.10	[-0.01,0.22]	0.22	[0.14,0.31]	0.13	[0.03,0.23]

Another consistent signal is found at the ACE gene with SNP rs4311, which is found associated with 12 ROIs. We note that all but one of these ROIs is in the right hemisphere, and three of these ROIs (all of which are in the right hemisphere) are in common with the ROIs selected for this SNP by Greenlaw et al. (2017). In Figure 3.5 we indicate the SNPs chosen for each ROI, where the SNPs are grouped on the x-axis by gene and the ROIs are grouped in left/right pairs on the y-axis. The selected SNPs for each ROI are shown for tuning parameter values $\lambda^2 = 1000$ and $\lambda^2 = 10,000$. In both cases the value of the spatial tuning parameter for the CAR model is set at $\rho = 0.8$ as suggested by Figure 3.3.

Examining Figure 3.5, two ROIs stand out has having a relatively broad genetic signal that persists even as the tuning parameter increases from $\lambda^2 = 1000$ to $\lambda^2 = 10,000$. These are *Left-Supramarg* (thickness of the left supramarginal gyrus) and *Left-SupTemporal* (thickness of the left superior temporal gyrus). For the case where $\lambda^2 = 1000$, phenotype *Left-Supramarg* is associated with 188 SNPs (top panel of Figure 3.5) and this decreases to 24 SNPs (bottom panel of Figure 3.5) when $\lambda^2 = 10,000$. When $\lambda^2 = 1000$ phenotype *Left-SupTemporal* is associated with 188 SNPs and this decreases to 23 SNPs when $\lambda^2 = 10,000$. This is to be compared with the average number of

Figure 3.5: ADNI-1 Data: SNPs chosen with the spatial model fit using Gibbs sampling and Bayesian FDR ($\alpha = 0.05$) are highlighted in red for each phenotype. The black ticks on y-axis indicate the phenotypes from the left/right hemisphere, and the SNPS from same gene are indicated by the ticks on x-axis. The top panel corresponds to the case $\lambda^2 = 1000$ while the bottom panel corresponds to the case $\lambda^2 = 10,000$.

SNPs selected over all ROIs when $\lambda^2 = 10,000$ which is just 2. The regularization paths for these two regions are shown in Figure 3.6 where the SNPs having the most persistent signal across values

of λ^2 are highlighted. These include SNP rs10868609 for the thickness of the left supramarginal gyrus and rs10501426 for the thickness of the left superior temporal gyrus.

3.6 Conclusion

We have developed a spatial multi-task regression model for relating genetic data to multivariate imaging phenotypes. The model uses a shrinkage prior with group penalization for the coefficients of each SNP (rows of $\mathbf{W}$) in the regression structure and the error structure for the imaging phenotype is based on a proper bivariate conditional autoregressive model, which allows us to account for bilateral correlation across brain hemispheres while also allowing us to account for within-hemisphere correlation using a graph structure characterized by a neighbourhood matrix. Ours is one of the first explicitly spatial hierarchical models for imaging genetics and neuroimaging to account for both spatial correlation and bilateral cross-hemisphere correlation.

Within the current context the investigation of alternative shrinkage priors is of interest. Future development will investigate a group spike-and-slab prior as an alternative to the Bayesian group LASSO prior. The development of reduced rank approximations for $\boldsymbol{W}$ in combination with mean-field approximations is also of interest in enabling the application of the bivariate spatial model to much larger scale imaging and genetic data. Future developments may also incorporate the Bayesian false discovery probability proposed by Wakefield (2007) which allows the user to account explicitly for the cost of false discoveries and the cost of false non-discovery.

With regards to the two computational algorithms, we recommend that the approximate VB procedures be used to initialize the MCMC algorithm and also to obtain an initial insight into the data while the MCMC sampler runs. It appears that VB combined with Bayesian FDR tends to be more liberal in the selection of SNPs, and in our application the SNP-ROI pairs selected by MCMC + Bayesian FDR are a proper subset of that selected by VB + Bayesian FDR. Nevertheless, Figure 3.4 suggests that the approximation can be reasonable when good initializations are used. In addition, improving the optimization algorithm by including gradient steps operating on the ELBO directly will likely lead to better VB solutions and potentially improved approximations.

While our current methodology is best suited for situations where the analysis is focussed on a relatively small set of targeted SNPs and a moderate number of ROIs, these are settings in which a full multivariate model for imaging genetics can be specified and fit. Extending the applicability of the methodology to settings with massive numbers of genetic and neuroimaging variables is an avenue for future work. Combining variational Bayes with better optimization schemes, reduced rank regression and the use of parallel matrix computations may be a useful avenue in this regard. Divide and conquer strategies such as the consensus Monte Carlo algorithm (Scott et al., 2016) as well as splitting up the brain into a smaller number of sub-regions might lead to feasible Monte Carlo implementations for such settings. Their design and implementation for imaging genetics and the Bayesian spatial model should lead to greater scalability.

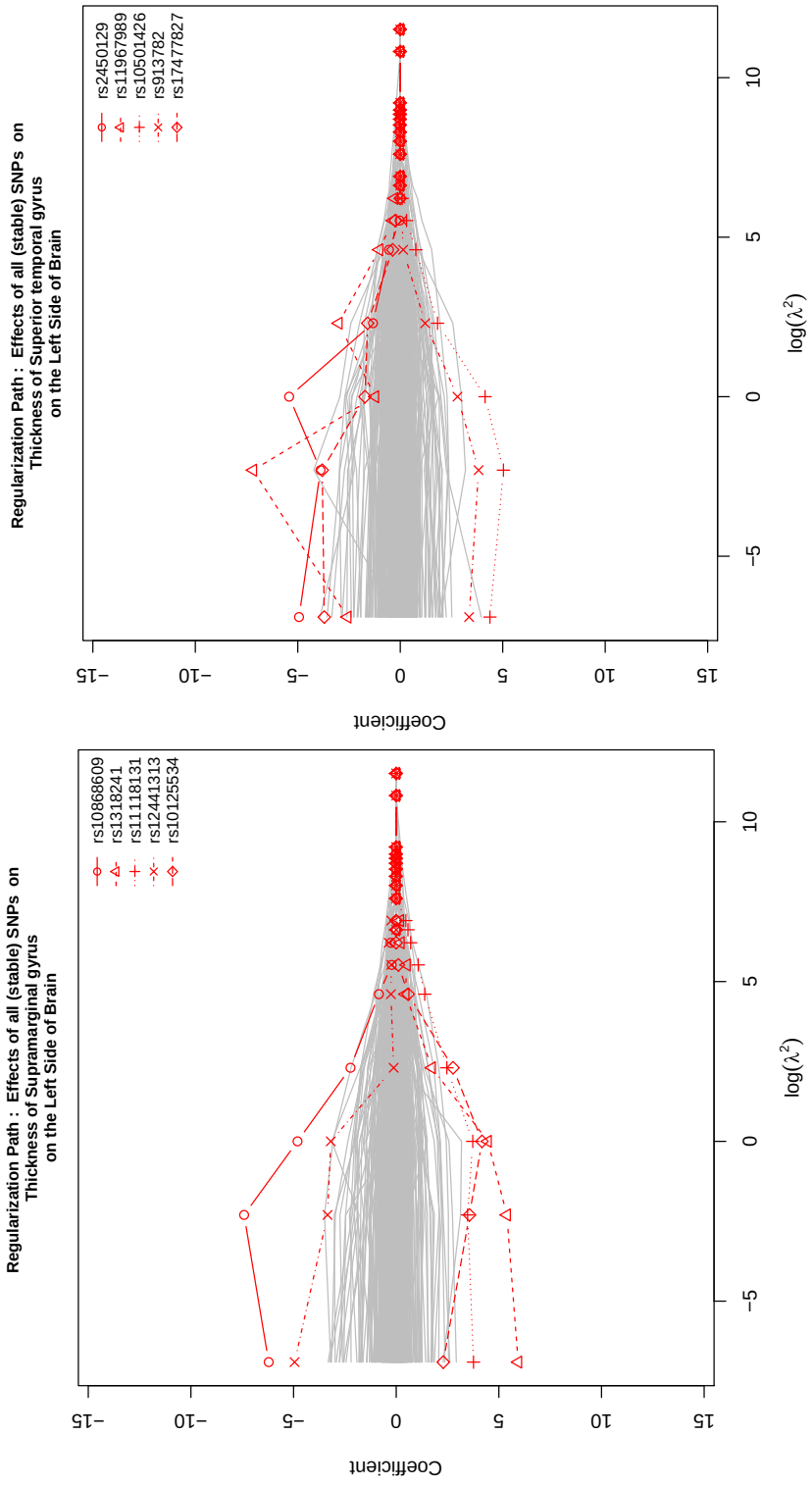


Figure 3.6: ADNI-1 Data: regularization paths showing the posterior mean estimates for varying λ^2 for all SNPs for the thickness of the supramarginal gyrus on the left side of the brain and the thickness of the superior temporal gyrus on the left side of the brain.

3.7 Appendix A

3.7.1 Selected SNPs and the Corresponding Regions of Interest for the ADNI-1 Application

Table 3.2: Application to ADNI-1 data: The 75 SNPs and corresponding phenotypes selected from the proposed Bayesian spatial group LASSO regression model with Gibbs Sampling combined with Bayesian FDR at $\alpha = 0.05$. These same SNP-ROI pairs are also selected by variational Bayes combined with Bayesian FDR at $\alpha = 0.05$. SNPs and phenotypes in bold correspond to those also chosen using 95% credible intervals and the model of Greenlaw et al. (2017).

SNP	Gene	Phenotype ID (hemisphere)
rs4305	ACE	SupTemporal(L), Supramarg(L)
rs4311	ACE	AmygVol(R), **CerebCtx(R)**, HippVol(R), **InfParietal(R)**, Parahipp(R), Precentral(L), **SupFrontal(R)**,**SupParietal(R)**, Supramarg(R), TemporalPole(R), MeanCing(R), MeanMedTemp(R)
rs4353	ACE	Supramarg(R)
rs405509	APOE	**MidTemporal(R)**, **Supramarg(R)**, **MeanFront(R)**, **MeanLatTemp(R)**
rs11191692	CALHM1	SupTemporal(L)
rs3811450	CHRNB2	SupParietal(R)
rs2025935	CR1	Postcentral(L), Supramarg(L)
rs10780849	DAPK1	InfParietal(R)
rs1105384	DAPK1	TemporalPole(R), MeanCing(L)
rs1473180	DAPK1	CerebWM(L)
rs17399090	DAPK1	MeanCing(L)
rs3095747	DAPK1	Postcentral(L)
rs3118853	DAPK1	SupTemporal(L)
rs3124237	DAPK1	InfParietal(R)
rs3124238	DAPK1	SupTemporal(L)
rs4877368	DAPK1	Parahipp(R)
rs4878117	DAPK1	Parahipp(R)
rs913782	DAPK1	InfParietal(R)
rs10916959	ECE1	Supramarg(L)
rs212539	ECE1	SupTemporal(L), Supramarg(L)
rs4654916	ECE1	SupTemporal(L), Supramarg(L)
rs6584307	ENTPD7	InfParietal(R)
rs11601726	GAB2	SupTemporal(L), Supramarg(L)
rs7927923	GAB2	SupTemporal(L)

Continued on next page

Table 3.2 – *Continued from previous page*

SNP	Gene	Phenotype ID (hemisphere)
rs17561	IL1A	InfTemporal(R)
rs16924159	IL33	TemporalPole(L), **MeanCing(L)**
rs928413	IL33	Postcentral(L)
rs1433099	LDLR	CerebWM(L), SupFrontal(R)
rs2228671	LDLR	MidTemporal(R)
rs2569537	LDLR	MeanCing(R)
rs6511720	LDLR	Postcentral(L), Supramarg(L)
rs688	LDLR	Supramarg(L)
rs2184226	MTHFR	SupTemporal(L), Supramarg(L)
rs3737964	MTHFR	MeanSensMotor(R)
rs4846048	MTHFR	Supramarg(L)
rs12209631	NEDD9	CerebWM(L)
rs1475345	NEDD9	InfParietal(R)
rs16871157	NEDD9	SupTemporal(L), Supramarg(L)
rs17496723	NEDD9	MeanFront(R)
rs2072834	NEDD9	Supramarg(L)
rs2182335	NEDD9	Precuneus(R), MeanTemp(R)
rs2182337	NEDD9	SupTemporal(L)
rs2950	NEDD9	SupTemporal(L), Supramarg(L)
rs4713379	NEDD9	InfParietal(R)
rs744970	NEDD9	Supramarg(L)
rs760680	NEDD9	PostCing(L), Postcentral(L), SupTemporal(L), Supramarg(L)
rs10501604	PICALM	Supramarg(L)
rs7938033	PICALM	Supramarg(L)
rs6084833	PRNP	PostCing(L), SupTemporal(L)
rs10748924	SORCS1	InfTemporal(R)
rs10786972	SORCS1	MeanCing(L), MeanTemp(R)
rs10787010	SORCS1	PostCing(L), MeanSensMotor(L)
rs10787011	SORCS1	Supramarg(L)
rs10884399	SORCS1	Supramarg(L)
rs11193198	SORCS1	SupTemporal(L)
rs12240854	SORCS1	Postcentral(L), SupTemporal(L)
rs1269918	SORCS1	**CerebWM(L)**
rs1887635	SORCS1	SupTemporal(L)
rs2149196	SORCS1	MidTemporal(R), Parahipp(R)
rs2243581	SORCS1	SupTemporal(L)

Continued on next page

Table 3.2 – *Continued from previous page*

SNP	Gene	Phenotype ID (hemisphere)
rs2418811	SORCS1	PostCing(L), SupTemporal(L)
rs596577	SORCS1	Supramarg(L)
rs7903481	SORCS1	InfParietal(R), InfTemporal(R)
rs10502262	SORL1	Postcentral(L)
rs1699102	SORL1	PostCing(L), Postcentral(L), SupTemporal(L), Supramarg(L)
rs1699105	SORL1	SupTemporal(L)
rs2276346	SORL1	Supramarg(L)
rs3781832	SORL1	Supramarg(L)
rs4936632	SORL1	SupTemporal(L)
rs661057	SORL1	SupTemporal(L)
rs726601	SORL1	Supramarg(L)
rs762484	TF	MeanCing(L)
rs1568400	THRA	MeanTemp(R)
rs3744805	THRA	HippVol(R), Parahipp(R), Precuneus(R)
rs7219773	TNK1	Parahipp(R)

3.8 Appendix B

3.8.1 Derivations for the Gibbs Sampling and Mean Field Variational Bayes Algorithm

Recall that $\boldsymbol{\omega}^2 = (\omega_1^2, \cdots, \omega_{\mathrm{d}}^2)$, $\tilde{W}_{i,j^*} = (W_{i,j}, W_{i,j+1})$, $i = 1, ..., d,\ j = 2j^* - 1,\ j^* = 1, ..., \frac{c}{2}$. based on the hierarchical prior setting, the joint posterior distribution can be expressed up to a normalizing constant as

$$
\begin{aligned}
p(\boldsymbol{W}, \omega_1^2, \cdots, \omega_{\mathrm{d}}^2, \boldsymbol{\Sigma}, |\boldsymbol{Y}) &\propto p(\boldsymbol{Y}|\boldsymbol{W}, \boldsymbol{\Sigma}) p(\boldsymbol{W}|\boldsymbol{\Sigma}, \boldsymbol{\omega}^2) p(\boldsymbol{\omega}^2) p(\boldsymbol{\Sigma}) \\
&\propto |(D_A - \rho A)^{-1} \otimes \boldsymbol{\Sigma}|^{-\frac{n}{2}} \exp\left\{ -\frac{1}{2} \sum_{\ell=1}^{n} (\boldsymbol{y}_\ell - \boldsymbol{W}^T \boldsymbol{x}_\ell)^T [(D_A - \rho A)^{-1} \otimes \boldsymbol{\Sigma}]^{-1} \right. \\
&\quad \left. (\boldsymbol{y}_\ell - \boldsymbol{W}^T \boldsymbol{x}_\ell) \right\} \\
&\quad \times \prod_{\mathrm{i}=1}^{\mathrm{d}} (\omega_i^2)^{-\frac{c}{2}} |\boldsymbol{\Sigma}|^{-\frac{c}{4}} \exp\{-\frac{1}{2} \sum_{j=1}^{\frac{c}{2}} \tilde{W}_{i,j}^T (\omega_i^2 \boldsymbol{\Sigma})^{-1} \tilde{W}_{i,j}\} \\
&\quad \times \prod_{\mathrm{i}=1}^{\mathrm{d}} \frac{(\frac{\lambda^2}{2})^{\frac{c+1}{2}}}{\Gamma(\frac{c+1}{2})} (\omega_i^2)^{\frac{c}{2} - \frac{1}{2}} \exp\{-\frac{\lambda^2}{2} \omega_i^2\} \\
&\quad \times \frac{|S|^{\frac{\nu}{2}}}{2^\nu \Gamma_2(\frac{\nu}{2})} |\boldsymbol{\Sigma}|^{-\frac{\nu+3}{2}} \exp\left\{ -\frac{1}{2} tr(S \boldsymbol{\Sigma}^{-1}) \right\}.
\end{aligned}
$$

The full conditional distribution of $\boldsymbol{W_{(i)}}$

For any matrix A, let $A_{(i)}$ be the i_{th} row of A, $A_{i,j}$ be i_{th} row and j_{th} column of A if A is a matrix and $A_{(i)}$ be the i_{th} element of A if A is a vector. And let $A_{(-i)}$ be the rest after removing $A_{(i)}$ from A. The full conditional distribution of $\boldsymbol{W_{(i)}},\ i = 1, \ldots, d$ is expressed as

$$\begin{aligned}
&p(\boldsymbol{W}_{(i)}\big|\boldsymbol{Y}, \boldsymbol{W}_{(-i)}, \boldsymbol{\omega}^2, \Sigma, \rho, \lambda^2) \propto p(\boldsymbol{Y}|\boldsymbol{W}_{(i)}, \boldsymbol{W}_{(-i)}, \Sigma, \rho) p(\boldsymbol{W}|\Sigma, \boldsymbol{\omega}^2) \\
&\propto \exp\left\{-\frac{1}{2}\sum_{\ell=1}^{n}(\boldsymbol{y}_\ell - \boldsymbol{W}^T\boldsymbol{x}_\ell)^T[(D_A - \rho A)^{-1} \otimes \Sigma]^{-1}(\boldsymbol{y}_\ell - \boldsymbol{W}^T\boldsymbol{x}_\ell)\right\} \\
&\times \prod |\omega_i^2\Sigma|^{-\frac{c}{4}} \exp\left\{-\frac{1}{2}\sum_{j^*=1}^{\frac{c}{2}} \tilde{w}_{ij^*}^T\ (\omega_i^2\Sigma)^{-1}\ \tilde{w}_{ij^*}\right\} \\
&\propto \exp\left\{-\frac{1}{2}\sum_{\ell=1}^{n}(\boldsymbol{y}_\ell - \boldsymbol{W}^T\boldsymbol{x}_\ell)^T[(D_A - \rho A)^{-1} \otimes \Sigma]^{-1}(\boldsymbol{y}_\ell - \boldsymbol{W}^T\boldsymbol{x}_\ell)\right\} \\
&\times \exp\left\{-\frac{1}{2}\sum_{\mathrm{i}\in\pi_\mathrm{k}}\sum_{j^*=1}^{\frac{c}{2}} \tilde{w}_{ij^*}^T\ (\omega_i^2\Sigma)^{-1}\ \tilde{w}_{ij^*}\right\}.
\end{aligned} \tag{5}$$

Split $\boldsymbol{W}$ into $\boldsymbol{W}_{(i)}$ and $\boldsymbol{W}_{(-i)}$ and rewrite the first exponent of (5) as

$$\exp\left\{-\frac{1}{2}\sum_{\ell=1}^{\mathrm{n}}(\boldsymbol{y}_\ell - \boldsymbol{W}_{(i)}^T\boldsymbol{x}_{\ell(i)} - \boldsymbol{W}_{(-i)}^T\boldsymbol{x}_{\ell(-i)})^T[(D_A - \rho A)^{-1} \otimes \Sigma]^{-1}(\boldsymbol{y}_\ell - \boldsymbol{W}_{(i)}^T\boldsymbol{x}_{\ell(i)} - \boldsymbol{W}_{(-i)}^T\boldsymbol{x}_{\ell(-i)})\right\}. \tag{6}$$

We vectorise the terms that include either $\boldsymbol{W}_{(i)}$ or $\boldsymbol{W}_{(-i)}$ and simplify based on:
1) $\mathrm{vec}(\boldsymbol{W}_{(i)}^T\boldsymbol{x}_{\ell(i)}) = (\boldsymbol{x}_{\ell(i)}^T \otimes I_c)\mathrm{vec}(\boldsymbol{W}_{(i)}^T)$,
2) $\mathrm{vec}(\boldsymbol{W}_{(-i)}^T\boldsymbol{x}_{\ell(-i)}) = (\boldsymbol{x}_{\ell(-i)}^T \otimes I_c)\mathrm{vec}(\boldsymbol{W}_{(-i)}^T)$.

These results give an equivalent expression for (6)

$$\exp\left\{\frac{-1}{2}\sum_{\ell=1}^{\mathrm{n}}\left(\boldsymbol{y}_\ell - (\boldsymbol{x}_{\ell(i)}^T \otimes I_c)\mathrm{vec}(\boldsymbol{W}_{(i)}^T) - (\boldsymbol{x}_{\ell(-i)}^T \otimes I_c)\mathrm{vec}(\boldsymbol{W}_{(-i)}^T)\right)^T\ [(D_A - \rho A)^{-1} \otimes \Sigma]^{-1}\cdot \right.$$
$$\left.\left(\boldsymbol{y}_\ell - (\boldsymbol{x}_{\ell(i)}^T \otimes I_c)\mathrm{vec}(\boldsymbol{W}_{(i)}^T) - (\boldsymbol{x}_{\ell(-i)}^T \otimes I_c)\mathrm{vec}(\boldsymbol{W}_{(-i)}^T)\right)\right\},$$

which can be expressed as

$$\exp\left\{-\frac{1}{2}\sum_{\ell=1}^{\mathrm{n}}\left(\boldsymbol{y}_\ell^T - \mathrm{vec}(\boldsymbol{W}_{(i)}^T)^T(\boldsymbol{x}_{\ell(i)}^T \otimes I_c)^T - \mathrm{vec}(\boldsymbol{W}_{(-i)}^T)^T)(\boldsymbol{x}_{\ell(-i)}^T \otimes I_c)^T\right)\ [(D_A - \rho A)^{-1} \otimes \Sigma]^{-1}\right.$$
$$\left.\left(\boldsymbol{y}_\ell - (\boldsymbol{x}_{\ell(i)}^T \otimes I_c)\mathrm{vec}(\boldsymbol{W}_{(i)}^T) - (\boldsymbol{x}_{\ell(-i)}^T \otimes I_c)\mathrm{vec}(\boldsymbol{W}_{(-i)}^T)\right)\right\},$$

Using $(A \otimes B)^T = (A^T \otimes B^T)$ and $(A \otimes B)^{-1} = (A^{-1} \otimes B^{-1})$ the above is simplified to

$$\exp\left\{-\frac{1}{2\,\sigma^2}\sum_{\ell=1}^{\mathrm{n}}\left(\boldsymbol{y}_\ell^T - \mathrm{vec}(\boldsymbol{W}_{(i)}^T)^T(\boldsymbol{x}_{\ell(i)}\otimes I_c) - \mathrm{vec}(\boldsymbol{W}_{(-i)}^T)^T(\boldsymbol{x}_{\ell(-i)}\otimes I_c)\right)[(D_A-\rho A)\otimes\Sigma^{-1}]\right.$$
$$\left.\left(\boldsymbol{y}_\ell - (\boldsymbol{x}_{\ell(i)}^T\otimes I_c)\mathrm{vec}(\boldsymbol{W}_{(i)}^T) - (\boldsymbol{x}_{\ell(-i)}^T\otimes I_c)\mathrm{vec}(\boldsymbol{W}_{(-i)}^T)\right)\right\}.$$

It is now possible to expand the expression. Only those terms that include $\boldsymbol{W}_{(i)}$ are kept, as the other terms are considered to be constants that can be factored out to become part of the normalising constant. We have

$$\begin{aligned}\exp\left\{-\frac{1}{2}\sum_{\ell=1}^{\mathrm{n}}\left(\right.\right.&-\boldsymbol{y}_\ell^T((D_A-\rho A)\otimes\Sigma^{-1})(\boldsymbol{x}_{\ell(i)}^T\otimes I_c)\mathrm{vec}(\boldsymbol{W}_{(i)}^T)\\
&-\mathrm{vec}(\boldsymbol{W}_{(i)}^T)^T(\boldsymbol{x}_{\ell(i)}\otimes I_c)[(D_A-\rho A)\otimes\Sigma^{-1}]\boldsymbol{y}_\ell\\
&+\mathrm{vec}(\boldsymbol{W}_{(i)}^T)^T(\boldsymbol{x}_{\ell(i)}\otimes I_c)[(D_A-\rho A)\otimes\Sigma^{-1}](\boldsymbol{x}_{\ell(i)}^T\otimes I_c)\mathrm{vec}(\boldsymbol{W}_{(i)}^T)\\
&+\mathrm{vec}(\boldsymbol{W}_{(i)}^T)^T(\boldsymbol{x}_{\ell(i)}\otimes I_c)[(D_A-\rho A)\otimes\Sigma^{-1}](\boldsymbol{x}_{\ell(-i)}^T\otimes I_c)\mathrm{vec}(\boldsymbol{W}_{(-i)}^T)\\
&\left.\left.+\mathrm{vec}(\boldsymbol{W}_{(-i)}^T)^T(\boldsymbol{x}_{\ell(-i)}\otimes I_c)[(D_A-\rho A)\otimes\Sigma^{-1}](\boldsymbol{x}_{\ell(i)}^T\otimes I_c)\mathrm{vec}(\boldsymbol{W}_{(i)}^T)\right)\right\},\end{aligned}$$

which can be expressed as

$$\begin{aligned}\exp\left\{-\frac{1}{2}\right.\Bigg[&\mathrm{vec}(\boldsymbol{W}_{(i)}^T)^T\sum_{\ell=1}^{\mathrm{n}}(\boldsymbol{x}_{\ell(i)}\otimes I_c)[(D_A-\rho A)\otimes\Sigma^{-1}](\boldsymbol{x}_{\ell(i)}^T\otimes I_c)\mathrm{vec}(\boldsymbol{W}_{(i)}^T)\\
&+2\mathrm{vec}(\boldsymbol{W}_{(i)}^T)^T\sum_{\ell=1}^{\mathrm{n}}(\boldsymbol{x}_{\ell(i)}\otimes I_c)[(D_A-\rho A)\otimes\Sigma^{-1}](\boldsymbol{x}_{\ell(-i)}^T\otimes I_c)\mathrm{vec}(\boldsymbol{W}_{(-i)}^T)\\
&-2\mathrm{vec}(\boldsymbol{W}_{(i)}^T)^T\sum_{\ell=1}^{\mathrm{n}}(\boldsymbol{x}_{\ell(i)}\otimes I_c)[(D_A-\rho A)\otimes\Sigma^{-1}]\boldsymbol{y}_\ell\Bigg]\Bigg\}.\end{aligned}$$

Next, consider the second exponent in (5),

$$\exp\left\{-\frac{1}{2}\sum_{\mathrm{i}\in\pi_{\mathrm{k}}}\sum_{j^*=1}^{\frac{c}{2}}\tilde{w}_{ij^*}^T\,(\omega_i^2\Sigma)^{-1}\,\tilde{w}_{ij^*}\right\},$$

and define a matrix, $\boldsymbol{H}_i$, such that $\boldsymbol{H}_i = \left[diag\left\{(\frac{1}{\omega_{\mathrm{i}}^2})I_{\frac{c}{2}}\right\}\otimes\Sigma^{-1}\right] = \left[diag\left\{\frac{1}{\omega_{\mathrm{i}}^2}\right\}\otimes I_{\frac{c}{2}}\otimes\Sigma^{-1}\right]$.

Notice that,

$$\sum_{\mathrm{i}\in\pi_{\mathrm{k}}}\sum_{j^*=1}^{\frac{c}{2}}\tilde{w}_{ij^*}^T\,(\omega_i^2\Sigma)^{-1}\,\tilde{w}_{ij^*} = \mathrm{vec}(\boldsymbol{W}_{(i)}^T)^T\boldsymbol{H}_i\mathrm{vec}(\boldsymbol{W}_{(i)}^T).$$

We can then rewrite (5), up to its normalising constant, as,

$$p(\boldsymbol{W}_{(i)}\big|\boldsymbol{Y}, \boldsymbol{W}_{(-i)}, \boldsymbol{\omega}^2, \Sigma, \lambda^2) \propto$$
$$\exp\left\{-\frac{1}{2}\left[\text{vec}(\boldsymbol{W}_{(i)}^T)^T(H_k + \sum_{\ell=1}^{\text{n}}(\boldsymbol{x}_{\ell(i)} \otimes I_c)[(D_A - \rho A) \otimes \Sigma^{-1}](\boldsymbol{x}_{\ell(i)}^T \otimes I_c))\text{vec}(\boldsymbol{W}_{(i)}^T)\right.\right.$$
$$+2\text{vec}(\boldsymbol{W}_{(i)}^T)^T \sum_{\ell=1}^{\text{n}}(\boldsymbol{x}_{\ell(i)} \otimes I_c)[(D_A - \rho A) \otimes \Sigma^{-1}](\boldsymbol{x}_{\ell(-i)}^T \otimes I_c)\text{vec}(\boldsymbol{W}_{(-i)}^T)$$
$$\left.\left.-2\text{vec}(\boldsymbol{W}_{(i)}^T)^T \sum_{\ell=1}^{\text{n}}(\boldsymbol{x}_{\ell(i)} \otimes I_c)[(D_A - \rho A) \otimes \Sigma^{-1}]\boldsymbol{y}_\ell\right]\right\}. \quad (7)$$

Expression (7) is a quadratic form in $\text{vec}(\boldsymbol{W}_{(i)}^T)$ in the exponent. Therefore, the full conditional distribution of $\text{vec}(\boldsymbol{W}_{(i)}^T)$ is multivariate normal of dimension $m_k c$, with parameters, say, $\underset{\sim}{\boldsymbol{\mu_i}}$ and $\boldsymbol{\Sigma}_i$. After expanding, the exponent of a multivariate normal distribution is of the form,

$$\exp\left\{-\frac{1}{2}\left[\text{vec}(\boldsymbol{W}_{(i)}^T)^T\boldsymbol{\Sigma}_i^{-1}\text{vec}(\boldsymbol{W}_{(i)}^T) - 2\text{vec}(\boldsymbol{W}_{(i)}^T)^T\boldsymbol{\Sigma}_i^{-1}\underset{\sim}{\boldsymbol{\mu_i}} + \textit{constant}\right]\right\}. \quad (8)$$

The next steps involve matching (7) to (8).

Solving for $\boldsymbol{\Sigma}_i$*:*

Consider the terms of (7) that are quadratic in $\text{vec}(\boldsymbol{W}_{(i)}^T)$. We have

$$\exp\left\{-\frac{1}{2}\left[\text{vec}(\boldsymbol{W}_{(i)}^T)^T(\boldsymbol{H}_i + \sum_{\ell=1}^{\text{n}}(\boldsymbol{x}_{\ell(i)} \otimes I_c)[(D_A - \rho A) \otimes \Sigma^{-1}](\boldsymbol{x}_{\ell(i)}^T \otimes I_c))\text{vec}(\boldsymbol{W}_{(i)}^T)\right]\right\}.$$

Rearrange to obtain

$$\exp\left\{-\frac{1}{2}\left[\text{vec}(\boldsymbol{W}_{(i)}^T)^T\left(\sum_{\ell=1}^{\text{n}}(\boldsymbol{x}_{\ell(i)} \otimes I_c)(\boldsymbol{x}_{\ell(i)}^T \otimes I_c) + \boldsymbol{H}_i\right)\text{vec}(\boldsymbol{W}_{(i)}^T)\right]\right\}.$$

We now observe that

$$\boldsymbol{\Sigma}_i^{-1} = \left(\boldsymbol{H}_i + \sum_{\ell=1}^{\text{n}}(\boldsymbol{x}_{\ell(i)} \otimes I_c)[(D_A - \rho A) \otimes \Sigma^{-1}](\boldsymbol{x}_{\ell(i)}^T \otimes I_c)\right),$$
$$\boldsymbol{\Sigma}_i = \left(\boldsymbol{H}_i + \sum_{\ell=1}^{\text{n}}(\boldsymbol{x}_{\ell(i)} \otimes I_c)[(D_A - \rho A) \otimes \Sigma^{-1}](\boldsymbol{x}_{\ell(i)}^T \otimes I_c)\right)^{-1}.$$

Solving for $\underset{\sim}{\boldsymbol{\mu_i}}$:

Consider the term $-\frac{1}{2}\left(-2\text{vec}(\boldsymbol{W}_{(i)}^T)^T\boldsymbol{\Sigma}_i^{-1}\underset{\sim}{\boldsymbol{\mu_i}}\right)$ within the density of the multivariate normal density.

We have the expression,

$$-\frac{1}{2}\Big(2\text{vec}(\boldsymbol{W}_{(i)}^T)^T \sum_{\ell=1}^{\text{n}}(\boldsymbol{x}_{\ell(i)}\otimes I_c)[(D_A-\rho A)\otimes\Sigma^{-1}](\boldsymbol{x}_{\ell(-i)}^T\otimes I_c)\text{vec}(\boldsymbol{W}_{(-i)}^T) - 2\sum_{\ell=1}^{\text{n}}(\boldsymbol{x}_{\ell(i)}\otimes I_c)[(D_A-\rho A)\otimes\Sigma^{-1}]\boldsymbol{y}_\ell\Big)$$

$$= \text{vec}(\boldsymbol{W}_{(i)}^T)^T\Big(-\sum_{\ell=1}^{\text{n}}(\boldsymbol{x}_{\ell(i)}\otimes I_c)[(D_A-\rho A)\otimes\Sigma^{-1}](\boldsymbol{x}_{\ell(-i)}^T\otimes I_c)\text{vec}(\boldsymbol{W}_{(-i)}^T) + \sum_{\ell=1}^{\text{n}}(\boldsymbol{x}_{\ell(i)}\otimes I_c)[(D_A-\rho A)\otimes\Sigma^{-1}]\boldsymbol{y}_\ell\Big).$$

Match up the expressions,

$$\boldsymbol{\Sigma}_i^{-1}\underset{\sim}{\boldsymbol{\mu_i}} = \Big(-\sum_{\ell=1}^{\text{n}}(\boldsymbol{x}_{\ell(i)}\otimes I_c)[(D_A-\rho A)\otimes\Sigma^{-1}](\boldsymbol{x}_{\ell(-i)}^T\otimes I_c)\text{vec}(\boldsymbol{W}_{(-i)}^T) + \sum_{\ell=1}^{\text{n}}(\boldsymbol{x}_{\ell(i)}\otimes I_c)[(D_A-\rho A)\otimes\Sigma^{-1}]\boldsymbol{y}_\ell\Big).$$

Isolate $\underset{\sim}{\boldsymbol{\mu_i}}$ to obtain

$$\underset{\sim}{\boldsymbol{\mu_i}} = \boldsymbol{\Sigma}_i\Big(-\sum_{\ell=1}^{\text{n}}(\boldsymbol{x}_{\ell(i)}\otimes I_c)[(D_A-\rho A)\otimes\Sigma^{-1}](\boldsymbol{x}_{\ell(-i)}^T\otimes I_c)\text{vec}(\boldsymbol{W}_{(-i)}^T) + \sum_{\ell=1}^{\text{n}}(\boldsymbol{x}_{\ell(i)}\otimes I_c)[(D_A-\rho A)\otimes\Sigma^{-1}]\boldsymbol{y}_\ell\Big).$$

Finally, the full conditional distribution of $\boldsymbol{W_{(i)}}$ is expressed as

$$\text{vec}(\boldsymbol{W}_{(i)}^T)\big|\boldsymbol{Y},\boldsymbol{W}_{(-i)},\boldsymbol{\omega},\Sigma,\rho\lambda^2 \sim MVN_{m_k c}(\underset{\sim}{\boldsymbol{\mu_i}},\ \boldsymbol{\Sigma}_i), \tag{9}$$

where

$$\underset{\sim}{\boldsymbol{\mu_i}} = \boldsymbol{\Sigma}_i\Big(-\sum_{\ell=1}^{\text{n}}(\boldsymbol{x}_{\ell(i)}\otimes I_c)[(D_A-\rho A)\otimes\Sigma^{-1}](\boldsymbol{x}_{\ell(-i)}^T\otimes I_c)\text{vec}(\boldsymbol{W}_{(-i)}^T) + \sum_{\ell=1}^{\text{n}}(\boldsymbol{x}_{\ell(i)}\otimes I_c)[(D_A-\rho A)\otimes\Sigma^{-1}]\boldsymbol{y}_\ell\Big),$$

$$\boldsymbol{\Sigma}_i = \left(\boldsymbol{H}_i + \sum_{\ell=1}^{\text{n}}(\boldsymbol{x}_{\ell(i)}\otimes I_c)[(D_A-\rho A)\otimes\Sigma^{-1}](\boldsymbol{x}_{\ell(i)}^T\otimes I_c)\right)^{-1},\ \text{and}\ \boldsymbol{H}_i = \left[diag\left\{\frac{1}{\omega_{\text{i}}^2}\right\}\otimes I_{\frac{c}{2}}\otimes\Sigma^{-1}\right].$$

Since we already have the full conditional distribution, the coordinate-wise updates for mean field variational Bayes can be derived as:

$$
\begin{aligned}
q((\boldsymbol{W}_{(i)})^T) &\propto exp\Big\{\mathbf{E}_{-i}(logP((\boldsymbol{W}_{(i)})^T|rest))\Big\} \\
&\propto exp\Big\{\mathbf{E}_{-i}\Big(-\frac{c}{2}log(2\pi)-\frac{1}{2}log(det|\boldsymbol{\Sigma}_i|)-\frac{1}{2}((\boldsymbol{W}_{(i)})^T-\underset{\sim}{\boldsymbol{\mu_i}})^T\boldsymbol{\Sigma}_i^{-1}((\boldsymbol{W}_{(i)})^T-\underset{\sim}{\boldsymbol{\mu_i}})\Big)\Big\} \\
&\propto exp\Big\{\mathbf{E}_{-i}\Big(-const-\frac{1}{2}(\boldsymbol{W}_{(i)})\boldsymbol{\Sigma}_i^{-1}(\boldsymbol{W}_{(i)})^T+(\boldsymbol{W}_{(i)})\boldsymbol{\Sigma}_i^{-1}\underset{\sim}{\boldsymbol{\mu_i}}\Big)\Big\} \\
&\propto exp\Big\{const-\frac{1}{2}(\boldsymbol{W}_{(i)})\mathbf{E}_{-i}(\boldsymbol{\Sigma}_i^{-1})(\boldsymbol{W}_{(i)})^T+(\boldsymbol{W}_{(i)})\mathbf{E}_{-i}(\boldsymbol{\Sigma}_i^{-1}\underset{\sim}{\boldsymbol{\mu_i}})\Big\}.
\end{aligned}
$$

We still can see that $q((\boldsymbol{W}_{(i)})^T)$ is still MVN with

$$
\begin{aligned}
\boldsymbol{\Sigma}^{-1}_{q(\boldsymbol{W}_{(i)})} &= \mathbf{E}_{-i}(\boldsymbol{\Sigma}_i^{-1}) \\
&= E_{-i}\left(\left[\left\{\frac{1}{\omega_i^2}\right\}\otimes I_{\frac{c}{2}}\otimes\boldsymbol{\Sigma}^{-1}\right]+\sum_{\ell=1}^{\mathrm{n}}(\boldsymbol{x}_{\ell(i)}\otimes I_c)[(D_A-\rho A)\otimes\boldsymbol{\Sigma}^{-1}](\boldsymbol{x}^T_{\ell(i)}\otimes I_c)\right) \\
&= \left(\left[\left\{\frac{1}{\omega_i^2}\right\}\otimes I_{\frac{c}{2}}\otimes E_{q(\boldsymbol{\Sigma})}(\boldsymbol{\Sigma}^{-1})\right]+\sum_{\ell=1}^{\mathrm{n}}(\boldsymbol{x}_{\ell(i)}\otimes I_c)[(D_A-\rho A)\otimes E_{q(\boldsymbol{\Sigma})}(\boldsymbol{\Sigma}^{-1})](\boldsymbol{x}^T_{\ell(i)}\otimes I_c)\right).
\end{aligned}
$$

Also, we can find that:

$$
\begin{aligned}
\mathbf{E}_{-i}(\boldsymbol{\Sigma}_i^{-1}\underset{\sim}{\boldsymbol{\mu_i}}) = \boldsymbol{\Sigma}^{-1}_{q(\boldsymbol{W}_{(i)})}\boldsymbol{\mu}_{q(\boldsymbol{W}_{(i)})} &= \mathbf{E}_{-i}\Big(-\sum_{\ell=1}^{\mathrm{n}}(\boldsymbol{x}_{\ell(i)}\otimes I_c)[(D_A-\rho A)\otimes\boldsymbol{\Sigma}^{-1}](\boldsymbol{x}^T_{\ell(-i)}\otimes I_c)(\boldsymbol{W}^T_{(-i)}) \\
&\quad+\sum_{\ell=1}^{\mathrm{n}}(\boldsymbol{x}_{\ell(i)}\otimes I_c)[(D_A-\rho A)\otimes\boldsymbol{\Sigma}^{-1}]\boldsymbol{y}_\ell\Big), \\
\Rightarrow \boldsymbol{\Sigma}^{-1}_{q(\boldsymbol{W}_{(i)})}\boldsymbol{\mu}_{q(\boldsymbol{W}_{(i)})} &= \Big(-\sum_{\ell=1}^{\mathrm{n}}(\boldsymbol{x}_{\ell(i)}\otimes I_c)[(D_A-\rho A)\otimes E_{q(\boldsymbol{\Sigma})}(\boldsymbol{\Sigma}^{-1})](\boldsymbol{x}^T_{\ell(-i)}\otimes I_c)(\boldsymbol{\mu}_{q(\boldsymbol{W}_{(-i)})}) \\
&\quad+\sum_{\ell=1}^{\mathrm{n}}(\boldsymbol{x}_{\ell(i)}\otimes I_c)[(D_A-\rho A)\otimes E_{q(\boldsymbol{\Sigma})}(\boldsymbol{\Sigma}^{-1})]\boldsymbol{y}_\ell\Big), \\
\Rightarrow \boldsymbol{\mu}_{q(\boldsymbol{W}_{(i)})} &= \boldsymbol{\Sigma}_{q(\boldsymbol{W}_{(i)})}\Big(-\sum_{\ell=1}^{\mathrm{n}}(\boldsymbol{x}_{\ell(i)}\otimes I_c)[(D_A-\rho A)\otimes E_{q(\boldsymbol{\Sigma})}(\boldsymbol{\Sigma}^{-1})](\boldsymbol{x}^T_{\ell(-i)}\otimes I_c)(\boldsymbol{\mu}_{q(\boldsymbol{W}_{(-i)})}) \\
&\quad+\sum_{\ell=1}^{\mathrm{n}}(\boldsymbol{x}_{\ell(i)}\otimes I_c)[(D_A-\rho A)\otimes E_{q(\boldsymbol{\Sigma})}(\boldsymbol{\Sigma}^{-1})]\boldsymbol{y}_\ell\Big).
\end{aligned}
$$

Then, we can also compute:

$$
\begin{aligned}
E_q[log(q(vec\boldsymbol{W}_{(i)}^T))] = E_q\Big[& -\frac{1}{2}log|2\pi\boldsymbol{\Sigma}_{q(\boldsymbol{W}_{(i)})}| - \frac{1}{2}\left(vec(\boldsymbol{W}_{(i)}^T) - \boldsymbol{\mu}_{q(\boldsymbol{W}_{(i)})}\right)^T \boldsymbol{\Sigma}^{-1}_{q(\boldsymbol{W}_{(i)})} \\
& \left(vec(\boldsymbol{W}_{(i)}^T) - \boldsymbol{\mu}_{q(\boldsymbol{W}_{(i)})}\right)\Big] \\
= & -\frac{1}{2}log|2\pi\boldsymbol{\Sigma}_{q(\boldsymbol{W}_{(i)})}| - \frac{1}{2}\boldsymbol{\mu}^T_{q(\boldsymbol{W}_{(i)})}\boldsymbol{\Sigma}^{-1}_{q(\boldsymbol{W}_{(i)})}\boldsymbol{\mu}_{q(\boldsymbol{W}_{(i)})} \\
& + E_q\Big[-\frac{1}{2}(\boldsymbol{W}_{(i)})\boldsymbol{\Sigma}^{-1}_{q(\boldsymbol{W}_{(i)})}(\boldsymbol{W}_{(i)})^T + (\boldsymbol{W}_{(i)})\boldsymbol{\Sigma}^{-1}_{q(\boldsymbol{W}_{(i)})}\boldsymbol{\mu}_{q(\boldsymbol{W}_{(i)})} \Big] \\
= & -\frac{1}{2}log|2\pi\boldsymbol{\Sigma}_{q(\boldsymbol{W}_{(i)})}| - \frac{1}{2}\boldsymbol{\mu}^T_{q(\boldsymbol{W}_{(i)})}\boldsymbol{\Sigma}^{-1}_{q(\boldsymbol{W}_{(i)})}\boldsymbol{\mu}_{q(\boldsymbol{W}_{(i)})} \\
& - \frac{1}{2}\Big(\boldsymbol{\mu}^T_{q(\boldsymbol{W}_{(i)})}\boldsymbol{\Sigma}^{-1}_{q(\boldsymbol{W}_{(i)})}\boldsymbol{\mu}_{q(\boldsymbol{W}_{(i)})} + tr(\boldsymbol{\Sigma}^{-1}_{q(\boldsymbol{W}_{(i)})}\boldsymbol{\Sigma}_{q(\boldsymbol{W}_{(i)})})\Big) + \boldsymbol{\mu}^T_{q(\boldsymbol{W}_{(i)})}\boldsymbol{\Sigma}^{-1}_{q(\boldsymbol{W}_{(i)})}\boldsymbol{\mu}_{q(\boldsymbol{W}_{(i)})} \\
= & -\frac{1}{2}log|2\pi\boldsymbol{\Sigma}_{q(\boldsymbol{W}_{(i)})}| - \frac{1}{2}tr(\boldsymbol{\Sigma}^{-1}_{q(\boldsymbol{W}_{(i)})}\boldsymbol{\Sigma}_{q(\boldsymbol{W}_{(i)})}).
\end{aligned}
$$

Full conditional distribution of $\boldsymbol{\Sigma}$

$$
\begin{aligned}
p(\boldsymbol{\Sigma}\big|\boldsymbol{Y}, \boldsymbol{W}, \boldsymbol{\omega}^2) &\propto p(\boldsymbol{Y}|\boldsymbol{W}, \boldsymbol{\Sigma})p(\boldsymbol{W}|\boldsymbol{\Sigma}, \boldsymbol{\omega}^2)p(\boldsymbol{\Sigma}) \\
&\propto |(D_A - \rho A)^{-1} \otimes \boldsymbol{\Sigma}|^{-\frac{n}{2}} \exp\left\{ -\frac{1}{2}\sum_{\ell=1}^{n}(\boldsymbol{y}_\ell - \boldsymbol{W}^T\boldsymbol{x}_\ell)^T[(D_A - \rho A) \otimes \boldsymbol{\Sigma}^{-1}](\boldsymbol{y}_\ell - \boldsymbol{W}^T\boldsymbol{x}_\ell)\right\} \\
&\times \prod_{\mathrm{i}=1}^{\mathrm{d}} |\omega_i^2\boldsymbol{\Sigma}|^{-\frac{c}{4}} \exp\left\{-\frac{1}{2}\sum_{j^*=1}^{\frac{c}{2}} \tilde{W}^T_{i,j^*}\ (\omega_i^2\boldsymbol{\Sigma})^{-1}\ \tilde{W}_{i,j^*}\right\} \\
&\times \frac{|S|^{\frac{v}{2}}}{2^v\Gamma_2(\frac{v}{2})}|\boldsymbol{\Sigma}|^{-\frac{v+3}{2}} \exp\left\{-\frac{1}{2}tr(S\boldsymbol{\Sigma}^{-1})\right\}.
\end{aligned}
$$

Denote $y_l^* = y_l - \boldsymbol{W}^T x_l$, $\tilde{y^*}^T_{l,j^*} = (y^*_{l,j}, y^*_{l,j+1})$, $j = 2j^* - 1$, $j^* = 1, ..., \frac{c}{2}$, $l = 1, ..., n$. Let $B = D_A - \rho A$, then dim(B)=$\frac{c}{2} \times \frac{c}{2}$. Denote $b_{i,j}$ be i_{th} row and j_{th} column of B, $b_{i,j}$ is a scalar, where $i = 1, ..., \frac{c}{2}$, $j = 1, ..., \frac{c}{2}$. Using $|E \otimes F| = |E|^n|F|^m$, where $dim(E) = n \times n$ and $dim(F) = m \times m$. $tr(G) + tr(Q) = tr(G + Q)$ where dim(G)=dim(Q) and $tr(JK) = tr(KJ)$ where dim(J)=dim(K^T). This can be simplified as:

$$
\begin{aligned}
p(\boldsymbol{\Sigma}\big|\boldsymbol{Y}, \boldsymbol{W}, \boldsymbol{\omega}^2) \propto |D_A - \rho A|^{\frac{nc}{4}}|\boldsymbol{\Sigma}|^{-n} &\exp\left\{-\frac{1}{2}tr\left(\sum_{l=1}^{n}\sum_{i=1}^{\frac{c}{2}}\sum_{j=1}^{\frac{c}{2}} b_{i,j}\tilde{y^*}_{l,i}\tilde{y^*}_{l,i}^T\boldsymbol{\Sigma}^{-1}\right)\right\} \\
&\times \prod_{\mathrm{i}=1}^{\mathrm{d}} |\omega_i^2\boldsymbol{\Sigma}|^{-\frac{c}{4}} \exp\left\{-\frac{1}{2}\sum_{j^*=1}^{\frac{c}{2}} \tilde{W}^T_{i,j^*}\ (\omega_i^2\boldsymbol{\Sigma})^{-1}\ \tilde{W}_{i,j^*}\right\} \\
&\times \frac{|S|^{\frac{v}{2}}}{2^v\Gamma_2(\frac{v}{2})}|\boldsymbol{\Sigma}|^{-\frac{v+3}{2}} \exp\left\{-\frac{1}{2}tr(S\boldsymbol{\Sigma}^{-1})\right\}.
\end{aligned}
$$

Since $|D_A - \rho A|$, $\prod_{\mathrm{i=1}}^{\mathrm{d}} |\omega_i^2|^{-\frac{c}{2}}$ and $\frac{|S|^{\frac{v}{2}}}{2^v \Gamma_2(\frac{v}{2})}$ do not depend on $\mathbf{\Sigma}$, they can be factored out of the expression. This leaves,

$$
\begin{aligned}
p(\mathbf{\Sigma}\big|\boldsymbol{Y}, \boldsymbol{W}, \boldsymbol{\omega}^2) &\propto |\mathbf{\Sigma}|^{-n} \exp\left\{-\frac{1}{2} tr\left(\sum_{l=1}^{n}\sum_{i=1}^{\frac{c}{2}}\sum_{j=1}^{\frac{c}{2}} b_{i,j}\tilde{y^*_{l,i}}\tilde{y^*_{l,i}}^T \mathbf{\Sigma}^{-1}\right)\right\} \\
&\times |\mathbf{\Sigma}|^{-\frac{cd}{4}} \exp\left\{-\frac{1}{2} tr\left(\sum_{i=1}^{d}\sum_{j^*=1}^{\frac{c}{2}} \frac{\tilde{W_{i,j^*}}\tilde{W_{i,j^*}}^T}{\omega_i^2}\mathbf{\Sigma}^{-1}\right)\right\} \\
&\times |\mathbf{\Sigma}|^{-\frac{v+3}{2}} \exp\left\{-\frac{1}{2} tr(S\,\mathbf{\Sigma}^{-1})\right\} \\
&\propto |\mathbf{\Sigma}|^{-\frac{2n+\frac{cd}{2}+v+3}{2}} \exp\left\{-\frac{1}{2} tr\left[\left(\sum_{l=1}^{n}\sum_{i=1}^{\frac{c}{2}}\sum_{j=1}^{\frac{c}{2}} b_{i,j}\tilde{y^*_{l,i}}\tilde{y^*_{l,i}}^T + \sum_{i=1}^{d}\sum_{j^*=1}^{\frac{c}{2}} \frac{\tilde{W}_{i,j^*}\tilde{W}^T_{i,j^*}}{\omega_i^2} + S\right)\mathbf{\Sigma}^{-1}\right]\right\}.
\end{aligned}
$$

Therefore

$$
\mathbf{\Sigma} \sim Inverse - Wishart(S^*, v^*), \tag{10}
$$

where

$$
\begin{aligned}
S^* &= \sum_{l=1}^{n}\sum_{i=1}^{\frac{c}{2}}\sum_{j=1}^{\frac{c}{2}} b_{i,j}\tilde{y^*_{l,i}}\tilde{y^*_{l,i}}^T + \sum_{i=1}^{d}\sum_{j^*=1}^{\frac{c}{2}} \frac{\tilde{W}_{i,j^*}\tilde{W}^T_{i,j^*}}{\omega_i^2} + S, \\
v^* &= 2n + \frac{cd}{2} + v.
\end{aligned}
$$

Here S^* is a 2×2 matrix and v^* is a scalar. Similarly, we can derive the mean field approximation for $\mathbf{\Sigma}$ based on the full conditional distribution as follows:

$$
\begin{aligned}
q(\mathbf{\Sigma}) &\propto exp\Big\{\mathbf{E}_{rest}\Big(logP(\mathbf{\Sigma}|rest)\Big)\Big\} \\
&\propto exp\Big\{\mathbf{E}_{rest}\Big(- \frac{2n + \frac{cd}{2} + v + 3}{2} log(|\mathbf{\Sigma}|) \\
&\quad - \frac{1}{2} tr\Big[\Big(\sum_{l=1}^{n}\sum_{i=1}^{\frac{c}{2}}\sum_{j=1}^{\frac{c}{2}} b_{i,j}\tilde{y^*_{l,i}}\tilde{y^*_{l,i}}^T + \sum_{i=1}^{d}\sum_{j^*=1}^{\frac{c}{2}} \frac{\tilde{W_{i,j^*}}\tilde{W_{i,j^*}}^T}{\omega_i^2} + S\Big)\mathbf{\Sigma}^{-1}\Big]\Big)\Big\} \\
&\propto exp\Big\{ - \frac{2n + \frac{cd}{2} + v + 3}{2} log(|\mathbf{\Sigma}|) \\
&\quad - \mathbf{E}_{rest}\Big(\frac{1}{2} tr\Big[\Big(\sum_{l=1}^{n}\sum_{i=1}^{\frac{c}{2}}\sum_{j=1}^{\frac{c}{2}} b_{i,j}\tilde{y^*_{l,i}}\tilde{y^*_{l,i}}^T + \sum_{i=1}^{d}\sum_{j^*=1}^{\frac{c}{2}} \frac{\tilde{W_{i,j^*}}\tilde{W_{i,j^*}}^T}{\omega_i^2} + S\Big)\mathbf{\Sigma}^{-1}\Big]\Big)\Big\}.
\end{aligned}
$$

This is still a Inverse-Wishart distribution. That is

$$
q(\mathbf{\Sigma}) \sim Inverse - Wishart(S_{q(\mathbf{\Sigma})}, v_{q(\mathbf{\Sigma})}).
$$

For $\tilde{W}_{i,j^*} = (W_{i,j}, W_{i,j+1}),\ j = 2j^* - 1,\ j^* = 1, ..., \frac{c}{2}$.

$$E_{q(\boldsymbol{W})}(\tilde{W}_{i,j^*}\tilde{W}_{i,j^*}^T) = \begin{bmatrix} E_{q(\boldsymbol{W})}(W_{i,j}^2) & E_{q(\boldsymbol{W})}(W_{i,j}W_{i,j+1}) \\ E_{q(\boldsymbol{W})}(W_{i,j}W_{i,j+1}) & E_{q(\boldsymbol{W})}(W_{i,j+1}^2) \end{bmatrix}.$$

Now, for $E_{q(\boldsymbol{W})}(W_{i,j}^2)$ and $E_{q(\boldsymbol{W})}(W_{i,j}W_{i,j+1})$, we can get that:

$$\begin{aligned} E_{q(\boldsymbol{W})}(W_{i,j}^2) &= (\boldsymbol{\mu}_{q(\boldsymbol{W}_{(i)})_{(j)}})^2 + \boldsymbol{\Sigma}_{q(\boldsymbol{W}_{(i)})\,j,j}\,, \\ E_{q(\boldsymbol{W})}(W_{i,j+1}^2) &= (\boldsymbol{\mu}_{q(\boldsymbol{W}_{(i)})_{(j+1)}})^2 + \boldsymbol{\Sigma}_{q(\boldsymbol{W}_{(i)})\,j+1,j+1}\,, \\ E_{q(\boldsymbol{W})}(W_{i,j}W_{i,j+1}) &= \boldsymbol{\mu}_{q(\boldsymbol{W}_{(i)})_{(j)}}\boldsymbol{\mu}_{q(\boldsymbol{W}_{(i)})_{(j+1)}} + \boldsymbol{\Sigma}_{q(\boldsymbol{W}_{(i)})\,j,j+1}\,, \end{aligned}$$

$$\begin{aligned} S_{q(\boldsymbol{\Sigma})} &= E_{rest}\left(\sum_{l=1}^{n}\sum_{i=1}^{\frac{c}{2}}\sum_{j=1}^{\frac{c}{2}} b_{i,j}\tilde{y}_{l,i}^{*}\tilde{y}_{l,i}^{*\,T} + \sum_{i=1}^{d}\sum_{j^*=1}^{\frac{c}{2}}\frac{\tilde{W_{i,j^*}}\tilde{W_{i,j^*}}^T}{\omega_i^2} + S\right) \\ &= \left(\sum_{l=1}^{n}\sum_{i=1}^{\frac{c}{2}}\sum_{j=1}^{\frac{c}{2}} b_{i,j}\tilde{y}_{l,i}^{*}\tilde{y}_{l,i}^{*\,T} + \sum_{i=1}^{d}\sum_{j^*=1}^{\frac{c}{2}} E_q\Big(\tilde{W_{i,j^*}}\tilde{W_{i,j^*}}^T\Big)E_q\Big(\frac{1}{\omega_i^2}\Big) + S\right), \end{aligned}$$

$$v_{q(\boldsymbol{\Sigma})} = 2n + \frac{cd}{2} + v.$$

Now for $log(q(\boldsymbol{\Sigma}))$:

$$\begin{aligned} E_q(log(q(\boldsymbol{\Sigma}))) &= E_q\left[\frac{v_{q(\boldsymbol{\Sigma})}}{2}log|S_{q(\boldsymbol{\Sigma})}| - log(2^{v_{q(\boldsymbol{\Sigma})}}) - log\Gamma_2(\frac{v_{q(\boldsymbol{\Sigma})}}{2}) - \frac{v+3}{2}log(|\boldsymbol{\Sigma}|) - \frac{1}{2}tr(S_{q(\boldsymbol{\Sigma})}\boldsymbol{\Sigma}^{-1})\right] \\ &= \frac{v_{q(\boldsymbol{\Sigma})}}{2}log|S_{q(\boldsymbol{\Sigma})}| - log(2^{v_{q(\boldsymbol{\Sigma})}}) - log\Gamma_2(\frac{v_{q(\boldsymbol{\Sigma})}}{2}) - E_q\left[\frac{v+3}{2}log(|\boldsymbol{\Sigma}|) - \frac{1}{2}tr(S_{q(\boldsymbol{\Sigma})}\boldsymbol{\Sigma}^{-1})\right]. \end{aligned}$$

Full Conditional of ω^2

We consider a joint update of the scale mixing variable based on the corresponding full conditional distribution. We have:

$$\begin{aligned} p(\boldsymbol{\omega}^2\big|\boldsymbol{Y}, \boldsymbol{W}, \boldsymbol{\Sigma}) &\propto p(\boldsymbol{W}|\boldsymbol{\Sigma}, \boldsymbol{\omega}^2)p(\boldsymbol{\omega}^2|\lambda^2) \\ &\propto \prod_{\mathrm{i=1}}^{\mathrm{d}}(\omega_i^2)^{-\frac{c}{2}}|\boldsymbol{\Sigma}|^{-\frac{c}{4}}\exp\left\{-\frac{1}{2}\sum_{j^*=1}^{\frac{c}{2}}\tilde{w}_{i,j^*}^T\,(\omega_i^2\boldsymbol{\Sigma})^{-1}\,\tilde{w}_{i,j^*}\right\} \\ &\times \prod_{\mathrm{i=1}}^{\mathrm{d}}\frac{(\frac{\lambda^2}{2})^{\frac{c+1}{2}}}{\Gamma(\frac{c+1}{2})}(\omega_i^2)^{\frac{c+1}{2}-1}\exp\{-\frac{\lambda^2}{2}\omega_i^2\} \\ &\propto \prod_{\mathrm{i=1}}^{\mathrm{d}}(\omega_i^2)^{-\frac{1}{2}}\exp\left\{-\left(\frac{\lambda^2}{2}\right)\omega_i^2 - \frac{c_i^*}{2\omega_i^2}\right\}, \end{aligned}$$

where:

$$c_i^* = \sum_{j^*=1}^{\frac{c}{2}} \tilde{W}_{i,j^*}^T \boldsymbol{\Sigma}^{-1} \tilde{W}_{i,j^*} = tr(\sum_{j^*=1}^{\frac{c}{2}} \tilde{W}_{i,j^*} \tilde{W}_{i,j^*}^T \boldsymbol{\Sigma}^{-1}).$$

The above expression shows that the scale mixing variables are conditionally independent given $\boldsymbol{Y}, \boldsymbol{W}, \boldsymbol{\Sigma}, \rho, \lambda^2$. We next apply a transformation of variable $\eta_i = (\omega_i^2)^{-1}$, Jacobian $= \left|\frac{d}{d\eta_i}\omega_i^2\right| = \eta_i^{-2}$ which yields:

$$p(\boldsymbol{\omega}^2 \big| \boldsymbol{Y}, \boldsymbol{W}, \boldsymbol{\Sigma}, \rho, \lambda^2) \propto \prod_{\mathrm{i}=1}^{\mathrm{d}} (\eta_i)^{-\frac{3}{2}} \exp\left\{-\left(\frac{\lambda^2}{2\eta_i}\right) - \frac{\eta_i c_i^*}{2}\right\},$$

and from this we see that the conditional distributions lie within the Inverse Gaussian family. More specifically we have

$$\eta_i = \frac{1}{\omega_{\mathrm{i}}^2} \;\Big|\; \boldsymbol{Y}, \boldsymbol{W}, \boldsymbol{\Sigma}, \rho, \lambda^2 \sim \textit{Inverse-Gaussian}\left(\sqrt{\frac{\lambda^2}{c_i^*}}\,,\;\; \lambda^2\right),\;\; i = 1, \ldots, d. \tag{11}$$

Now, since we already know the full conditional distribution of ω_i, we have:

$$\begin{aligned}
q(\eta_i) &\propto exp\Big\{\mathbf{E}_q\Big(logP(\eta_i|rest)\Big)\Big\} \\
&\propto exp\Big\{\mathbf{E}_q\Big(-\frac{3}{2}log(\eta_i) - \left(\frac{\lambda^2}{2\omega_i^2}\right) - \frac{\omega_i^2 c_i^*}{2}\Big)\Big\} \\
&\propto exp\left\{\Big(-\frac{3}{2}log(\eta_i) - \left(\frac{\lambda^2}{2\eta_i}\right) - \mathbf{E}_q\left(\frac{\eta_i c_i^*}{2}\right)\Big)\right\} \\
&\propto exp\left\{\Big(-\frac{3}{2}log(\eta_i) - \left(\frac{\lambda^2}{2}\right)\frac{1}{\eta_i} - \mathbf{E}_{c_i^*}(c_i^*)\left(\frac{\eta_i}{2}\right)\Big)\right\}.
\end{aligned}$$

Therefore, $q(\eta_i)$ is still an Inverse-Gaussian distribution with

$$\mu_{q(\eta_i)} = \sqrt{\frac{\lambda^2}{E_{c_i^*}(c_i^*)}},$$

$$E_{c_i^*}(c_i^*) = \sum_{j^*=1}^{\frac{c}{2}} E(tr(\tilde{W}_{i,j^*} \tilde{W}_{i,j^*}^T \boldsymbol{\Sigma}^{-1})) = \sum_{j^*=1}^{\frac{c}{2}} (E_{q(\boldsymbol{W})}(W_{i,j}^2) + E_{q(\boldsymbol{W})}(W_{i,j+1}^2)) tr(S_{q(\boldsymbol{\Sigma})}^{-1}).$$

Now, since η_i is an Inverse Gaussian, then $\omega_i^2 = 1/\eta_i$ will be a reciprocal of inverse gaussian. where we have:

$$\mu_{q(\omega_i^2)} = \frac{1}{\mu_{q(\eta_i)}} + \frac{1}{\lambda^2},$$
$$Var_{q(\omega_i^2)} = \frac{1}{\mu_{q(\eta_i)}\lambda^2} + \frac{2}{(\lambda^2)^2}.$$

For $E_q(log(q(\omega_i^2)))$, we can compute as:

$$\begin{aligned} E_q(log(q(\omega_i^2))) &= E_q\left[\frac{1}{2}\left(log(\lambda^2) - log(2\pi) - log(\omega_i^2)\right) - \frac{\lambda^2(1-\omega_i^2\mu_{q(\eta_i)})^2}{2\mu_{q(\eta_i)}^2\omega_i^2}\right] \\ &= \frac{1}{2}\left(log(\lambda^2) - log(2\pi)\right) - E_q\left[log(\omega_i^2)\right] - \lambda^2 E_q\left[\frac{1}{2\mu_{q(\eta_i)}^2}\left(\frac{1}{\omega_i^2}\right) - \frac{1}{\mu_{q(\eta_i)}^2} + \frac{\omega_i^2}{2}\right]. \end{aligned}$$

For $E_q(log(\omega_i^2))$, we can use Taylor series to approximate as:

$$E_q(log(\omega_i^2)) = log(\mu_{q(\omega_i^2)}) - \frac{1}{2\mu_{\omega_i^2}} Var_q[\omega_i^2].$$

Then, we can have:

$$\begin{aligned} E_q(log(q(\omega_i^2))) &= \frac{1}{2}\left(log(\lambda^2) - log(2\pi)\right) - log(\mu_{q(\omega_i^2)}) - \frac{1}{2\mu_{\omega_i^2}} Var_q[\omega_i^2] \\ &\quad - \lambda^2 E_q\left[\frac{1}{2\mu_{q(\eta_i)}^2}\left(\frac{1}{\omega_i^2}\right) - \frac{1}{\mu_{q(\eta_i)}^2} + \frac{\omega_i^2}{2}\right] \\ &= \frac{1}{2}\left(log(\lambda^2) - log(2\pi)\right) - log(\mu_{q(\omega_i^2)}) - \frac{1}{2\mu_{\omega_i^2}} Var_q[\omega_i^2] \\ &\quad - \lambda^2\left[\frac{1}{2\mu_{q(\eta_i)}^2}\mu_{q(\eta_i)} - \frac{1}{\mu_{q(\eta_i)}^2} + \frac{\mu_{q(\omega_i^2)}}{2}\right]. \end{aligned}$$

Therefore, the posterior distribution can be approximated by mean field variational bayes as:

$$P(\boldsymbol{\Theta}|Y) \approx \prod_{i=1}^{d}\left[q_{\boldsymbol{W}_{(i)}}(\boldsymbol{W}_{(i)})\right]\prod_{i=1}^{d}\left[q_{\omega_i^2}(\omega_i^2)\right]q_{\boldsymbol{\Sigma}}(\boldsymbol{\Sigma}),$$

where

$$q_{\boldsymbol{W}_{(i)}}(\boldsymbol{W}_{(i)}) \equiv MVN(\boldsymbol{\mu}_{q_{\boldsymbol{W}_{(i)}}}, \boldsymbol{\Sigma}_{q_{\boldsymbol{W}_{(i)}}}),\ q_{\omega_i^2}(\omega_i^2) \quad \equiv \text{reciprocal of Inverse Gaussian}(\mu_{q(\eta_i)}, \lambda^2),\ \text{and}$$
$$q_{\boldsymbol{\Sigma}}(\boldsymbol{\Sigma}) \equiv Inverse - Wishart(S_{q(\boldsymbol{\Sigma})}, v_{q(\boldsymbol{\Sigma})}).$$

Lower Bound $\mathcal{L}(q)$ for Variational Bayes

We now have derived the optimal q distributions. The logarithm lower bound takes following explicit form:

$$\begin{aligned}\mathcal{L}(q) &= E_q[log(P(\boldsymbol{Y},\boldsymbol{\theta}))] - E_q[log(q(\boldsymbol{\theta}))] \\ &= E_q[log(p(\boldsymbol{Y}|\boldsymbol{W},\boldsymbol{\Sigma},\rho)) + log(p(\boldsymbol{W}|\boldsymbol{\Sigma},\boldsymbol{\omega}^2)) + log(p(\boldsymbol{\omega}^2|\lambda^2)) + log(p(\boldsymbol{\Sigma}))] - E_q[log(q(\boldsymbol{\theta}))].\end{aligned}$$

Now, taking the expectation with respect to q for each component in above, we have:

$$\begin{aligned}E_q(log(p(\boldsymbol{Y}|\boldsymbol{W},\boldsymbol{\Sigma},\rho))) &= E_q\Big[-\frac{n}{2}log|(D_A-\rho A)^{-1}\otimes\boldsymbol{\Sigma}| - \frac{1}{2}\sum_{\ell=1}^{n}(\boldsymbol{y}_\ell - \boldsymbol{W}^T\boldsymbol{x}_\ell)^T[(D_A-\rho A)\otimes\boldsymbol{\Sigma}^{-1}] \\ &\quad (\boldsymbol{y}_\ell - \boldsymbol{W}^T\boldsymbol{x}_\ell)\Big] \\ &= -\frac{n}{2}log|(D_A-\rho A)^{-1}\otimes\mu_{q(\boldsymbol{\Sigma})}| - E_q\Big[\frac{1}{2}\sum_{\ell=1}^{n}(\boldsymbol{y}_\ell - \boldsymbol{W}^T\boldsymbol{x}_\ell)^T[(D_A-\rho A) \\ &\quad \otimes\boldsymbol{\Sigma}^{-1}](\boldsymbol{y}_\ell - \boldsymbol{W}^T\boldsymbol{x}_\ell)\Big], \\ E_q(log(p(\boldsymbol{W}|\boldsymbol{\Sigma},\boldsymbol{\omega}^2))) &= E_q\Big[\sum_{\mathrm{i}=1}^{\mathrm{d}}\sum_{j=1}^{\frac{c}{2}}\Big(-\frac{1}{2}log|\omega_i^2\boldsymbol{\Sigma}| - \frac{1}{2}\tilde{W}_{i,j}^T(\omega_i^2\boldsymbol{\Sigma})^{-1}\tilde{W}_{i,j}\Big)\Big] \\ &= \Big[\sum_{\mathrm{i}=1}^{\mathrm{d}}\sum_{j=1}^{\frac{c}{2}} -\frac{1}{2}log|\mu_{q(\omega_i^2)}\mu_{q(\boldsymbol{\Sigma})}| - E_q\Big(\frac{1}{2}\tilde{W}_{i,j}^T(\omega_i^2\boldsymbol{\Sigma})^{-1}\tilde{W}_{i,j}\Big)\Big], \\ E_q(log(p(\boldsymbol{\omega}^2|\lambda^2))) &= E_q\Big[\sum_{\mathrm{i}=1}^{\mathrm{d}}\Big(\frac{c+1}{2}log(\lambda^2) - log(\Gamma(\frac{c+1}{2})) + (\frac{c+1}{2}-1)log(\omega_i^2) - \frac{\lambda^2}{2}\omega_i^2\Big)\Big] \\ &= \Big[\sum_{\mathrm{i}=1}^{\mathrm{d}}\Big(\frac{c+1}{2}(log(\lambda^2) - log(2)) - log(\Gamma(\frac{c+1}{2})) + (\frac{c+1}{2}-1)E_q(log(\omega_i^2)) \\ &\quad - \frac{1}{2}\lambda^2 E_q(\omega_i^2)\Big)\Big],\end{aligned}$$

$$\begin{aligned}E_q(log(p(\boldsymbol{\Sigma}))) &= E_q\Big[const - (\frac{v+3}{2})log|\boldsymbol{\Sigma}| - \frac{1}{2}tr(S\,\boldsymbol{\Sigma}^{-1})\Big] \\ &= \Big[const - (\frac{v+3}{2})log|\mu_{q(\boldsymbol{\Sigma})}| - \frac{1}{2}tr(S\,E_{q(\boldsymbol{\Sigma})}(\boldsymbol{\Sigma}^{-1}))\Big].\end{aligned}$$

Now, let's take a look at the $E_q[log(q(\boldsymbol{\theta}))]$, which could be written as:

$$\begin{aligned}E_q[log(q(\boldsymbol{\theta}))] &= E_q[log(q(\boldsymbol{W}))] + E_q[log(q(\boldsymbol{\omega}^2))] + E_q[log(q(\boldsymbol{\Sigma}))] \\ &= \sum_{i=1}^{d}E_q[log(q(vec(\boldsymbol{W}_{(i)}^T)))] + \sum_{i=1}^{d}E_q[log(q(\omega_i^2))] + E_q[log(q(\boldsymbol{\Sigma}))].\end{aligned}$$

Where these expectations are evaluated from the forms derived above. Then the lower bound can be written as :

$$
\begin{aligned}
\mathcal{L}(q) &= E_q[log(p(\boldsymbol{Y}|\boldsymbol{W},\boldsymbol{\Sigma},\rho)) + log(p(\boldsymbol{W}|\boldsymbol{\Sigma},\boldsymbol{\omega}^2)) + log(p(\boldsymbol{\omega}^2|\lambda^2)) + log(p(\boldsymbol{\Sigma}))] - E_q[log(q(\boldsymbol{\theta}))] \\
&= E_q\Big[log(p(\boldsymbol{Y}|\boldsymbol{W},\boldsymbol{\Sigma},\rho)) + log(p(\boldsymbol{W}|\boldsymbol{\Sigma},\boldsymbol{\omega}^2)) + log(p(\boldsymbol{\omega}^2|\lambda^2)) + log(p(\boldsymbol{\Sigma}))\Big] \\
&- \Big[\sum_{i=1}^{d} E_q[log(q(vec(\boldsymbol{W}_{(i)}^T)))] + \sum_{i=1}^{d} E_q[log(q(\omega_i^2)] + E_q[log(q(\boldsymbol{\Sigma}))]\Big] \\
&= -\frac{n}{2} log\Big|(D_A - \rho A)^{-1} \otimes (S_{q(\boldsymbol{\Sigma})}/(v_{q(\boldsymbol{\Sigma})} - 3)\Big| \\
&- \frac{1}{2}\sum_{\ell=1}^{n} tr\Big((\boldsymbol{y}_\ell - \boldsymbol{\mu}_{\boldsymbol{q(W)}}\boldsymbol{x}_\ell)^T(\boldsymbol{y}_\ell - \boldsymbol{\mu}_{\boldsymbol{q(W)}}^T\boldsymbol{x}_\ell)^T[(D_A - \rho A) \otimes (v_{q(\boldsymbol{\Sigma})} * S_{q(\boldsymbol{\Sigma})}^{-1})]\Big) \\
&+ \sum_{\mathrm{i}=1}^{\mathrm{d}}\sum_{j=1}^{\frac{c}{2}}\Big[-\frac{1}{2}log\Big|(\frac{1}{\mu_{q(\eta_i)}} + \frac{1}{\lambda^2})(S_{q(\boldsymbol{\Sigma})}/(v_{q(\boldsymbol{\Sigma})} - 3)\Big| - E_q\left(\frac{1}{2}\mu_{q(\tilde{\boldsymbol{W}})}\mu_{q(\tilde{\boldsymbol{W}})}{}_{i,j}^{T}\mu_{q(\eta_i)}(v_{q(\boldsymbol{\Sigma})} * S_{q(\boldsymbol{\Sigma})}^{-1})\right)\Big] \\
&+ \sum_{\mathrm{i}=1}^{\mathrm{d}}\Big[\Big(\frac{c+1}{2}(log(\lambda^2) - log(2)) - log(\Gamma(\frac{c+1}{2}))\Big) \\
&+ \Big((\frac{c+1}{2} - 1)(log(\mu_{q(\omega_i^2)}) - \frac{1}{2\mu_{q_{\omega_i^2}}}Var_q[\omega_i^2]) - \frac{1}{2}\lambda^2\mu_{q(\omega_i^2)}\Big)\Big] \\
&- \Big[(\frac{v+3}{2})log|(S_{q(\boldsymbol{\Sigma})}/(v_{q(\boldsymbol{\Sigma})} - 3))| + \frac{1}{2}tr(S(v_{q(\boldsymbol{\Sigma})} * S_{q(\boldsymbol{\Sigma})}^{-1}))\Big] \\
&- \Big[\sum_{i=1}^{d}\Big(-\frac{1}{2}log|2\pi\boldsymbol{\Sigma}_{q(\boldsymbol{W}_{(i)})}| - \frac{1}{2}tr(\boldsymbol{\Sigma}_{q(\boldsymbol{W}_{(i)})}^{-1}\boldsymbol{\Sigma}_{q(\boldsymbol{W}_{(i)})})\Big) \\
&+ \frac{1}{2}\Big(log(\lambda^2) - log(2\pi)\Big) - log(\mu_{q(\omega_i^2)}) - \frac{1}{2\mu_{\omega_i^2}}Var_q[\omega_i^2]\lambda^2\left[\frac{1}{2\mu_{q(\eta_i)}^2}\mu_{q(\eta_i)} - \frac{1}{\mu_{q(\eta_i)}^2} + \frac{\mu_{q(\omega_i^2)}}{2}\right] \\
&+ \frac{v_{q(\boldsymbol{\Sigma})}}{2}log|S_{q(\boldsymbol{\Sigma})}| - log(2^{v_{q(\boldsymbol{\Sigma})}}) - log\Gamma_2(\frac{v_{q(\boldsymbol{\Sigma})}}{2}) \\
&- \left[\frac{v+3}{2}log(|(S_{q(\boldsymbol{\Sigma})}/(v_{q(\boldsymbol{\Sigma})} - 3)|) - \frac{1}{2}tr(S_{q(\boldsymbol{\Sigma})}(v_{q(\boldsymbol{\Sigma})} * S_{q(\boldsymbol{\Sigma})}^{-1}))\right]\Big].
\end{aligned}
$$

3.9 Appendix C

3.9.1 Derivation of the Moment Estimator for λ^2

We have developed a simple approach for tuning the algorithm that is based on a simple moment estimator of λ^2 and does not require multiple runs of the algorithm over different values of λ^2. It is thus well suited to computation on a single processor. Beginning with the model for $\boldsymbol{W}$ we have

$$\tilde{W}_{i,j^*}|\omega_i^2, \boldsymbol{\Sigma} \overset{ind}{\sim} \text{BVN}(\boldsymbol{0}, \omega_i^2\boldsymbol{\Sigma}),$$

$$\omega_i^2|\lambda^2 \overset{iid}{\sim} \text{Gamma}(\frac{c+1}{2}, \lambda^2/2),$$

$$\boldsymbol{\Sigma} \sim \text{Inv-Wishart}(\nu, \boldsymbol{S}).$$

From this we have

$$E[W_{i,j}^2] = E[E[W_{i,j}^2|\omega_i^2, \boldsymbol{\Sigma}]] = E[VAR[W_{i,j}|\omega_i^2, \boldsymbol{\Sigma}] + E^2[W_{i,j}|\omega_i^2, \boldsymbol{\Sigma}]]$$

$$= E[VAR[W_{i,j}|\omega_i^2, \boldsymbol{\Sigma}]] = \begin{cases} E[\omega_i^2\boldsymbol{\Sigma}_{11}], & \text{if coefficient is for the left hemisphere,} \\ E[\omega_i^2\boldsymbol{\Sigma}_{22}], & \text{if coefficient is for the right hemisphere.} \end{cases}$$

When $\boldsymbol{S} = \boldsymbol{I}$, we have $E[W_{i,j}^2] = \frac{c+1}{\lambda^2}\frac{1}{\nu-3}$. We then use the ridge estimator $\hat{\boldsymbol{W}}_R$ obtained to initialize the VB algorithm to setup a moment equation

$$\frac{c+1}{\lambda^2}\frac{1}{\nu-3} = \frac{1}{dc}\sum_{i,j}\hat{\boldsymbol{W}}_{R_{i,j}}^2,$$

Solving for λ^2 yields

$$\lambda^2 = \frac{dc(c+1)}{\nu-3}\left(\sum_{i,j}\hat{\boldsymbol{W}}_{R_{i,j}}^2\right)^{-1}.$$

We modify this equation so that it can apply more generally by replacing $\nu - 3$ with $\max\{1, \nu - 3\}$ which yields

$$\lambda^2 = \frac{dc(c+1)}{\max\{1,\nu-3\}}\left(\sum_{i,j}\hat{\boldsymbol{W}}_{R_{i,j}}^2\right)^{-1}.$$

3.10 Appendix D

3.10.1 Simulation Study Examining Empirical FDR

Table 3.3: Empirical proportion of false discoveries in 100 simulation replicates when the expected Bayesian FDR is controlled at $\alpha = 0.05$ leading to the threshold for posterior probabilities $\phi_\alpha = 0.05$. Note that the values of c^* are obtained after transforming the response matrix $\mathbb{Y}$ so that its columns are centered and scaled.

c^*	MCMC	VB
0.001	1.0000	1.0000
0.002	1.0000	1.0000
0.003	1.0000	1.0000
0.004	1.0000	0.9884
0.005	0.9412	0.8134
0.006	0.5640	0.5619
0.007	0.1997	0.3739
0.008	0.0579	0.2496
0.009	0.0185	0.1699
0.010	0.0067	0.1177
0.020	0.0000	0.0065
0.030	0.0000	0.0012
0.040	0.0000	3×10^{-4}
0.050	0.0000	1×10^{-4}
0.060	0.0000	0.0000

3.11 Appendix E

3.11.1 MCMC Trace Plots and Convergence Diagnostics

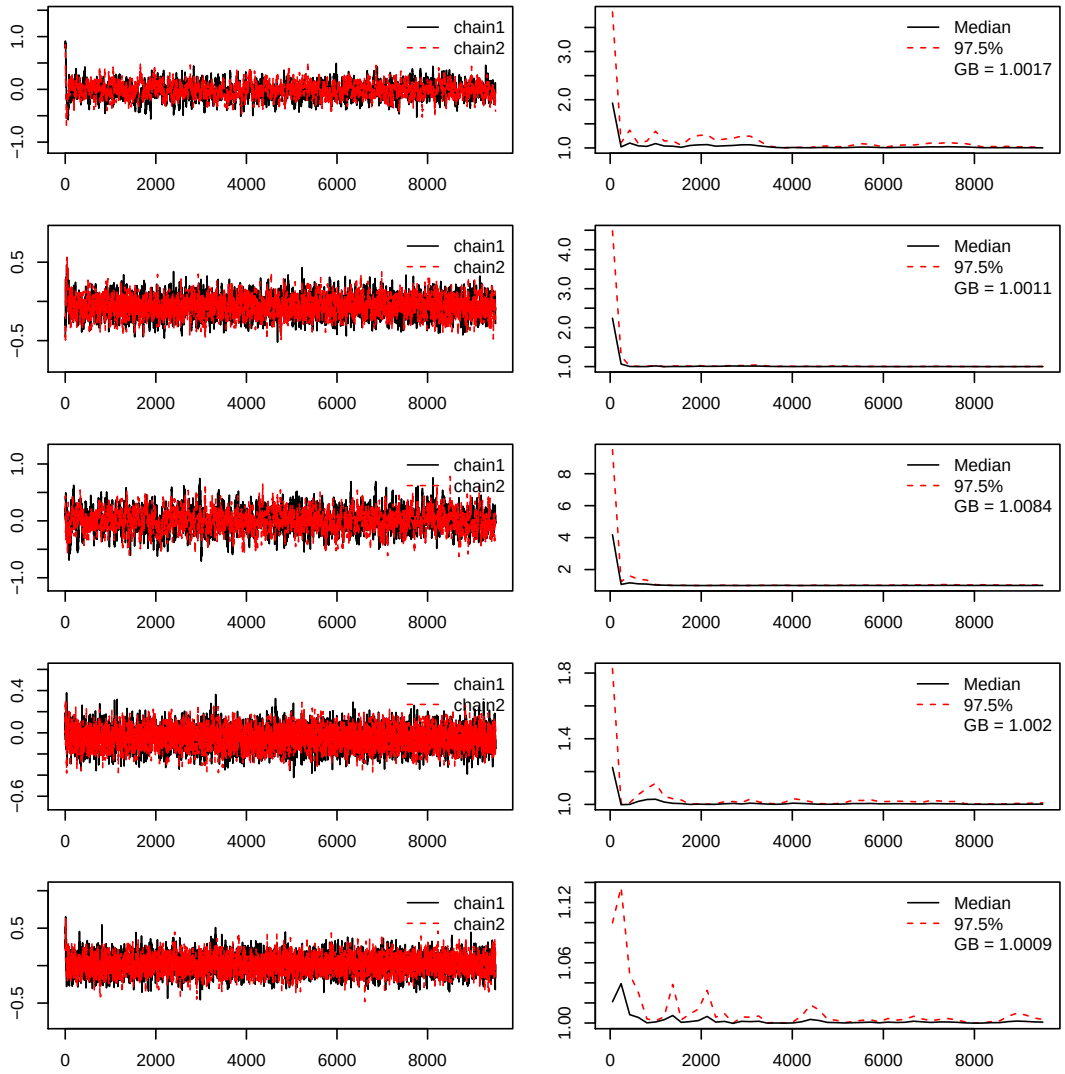


Figure 3.7: Left Panels displays MCMC trace plots representing 5 randomly selected elements of **W**. The right panels display the evolution of Gelman and Rubin shrink factor as the number of iterations increase and GB is the point estimate of Gelman and Rubin's convergence diagnostic statistic.

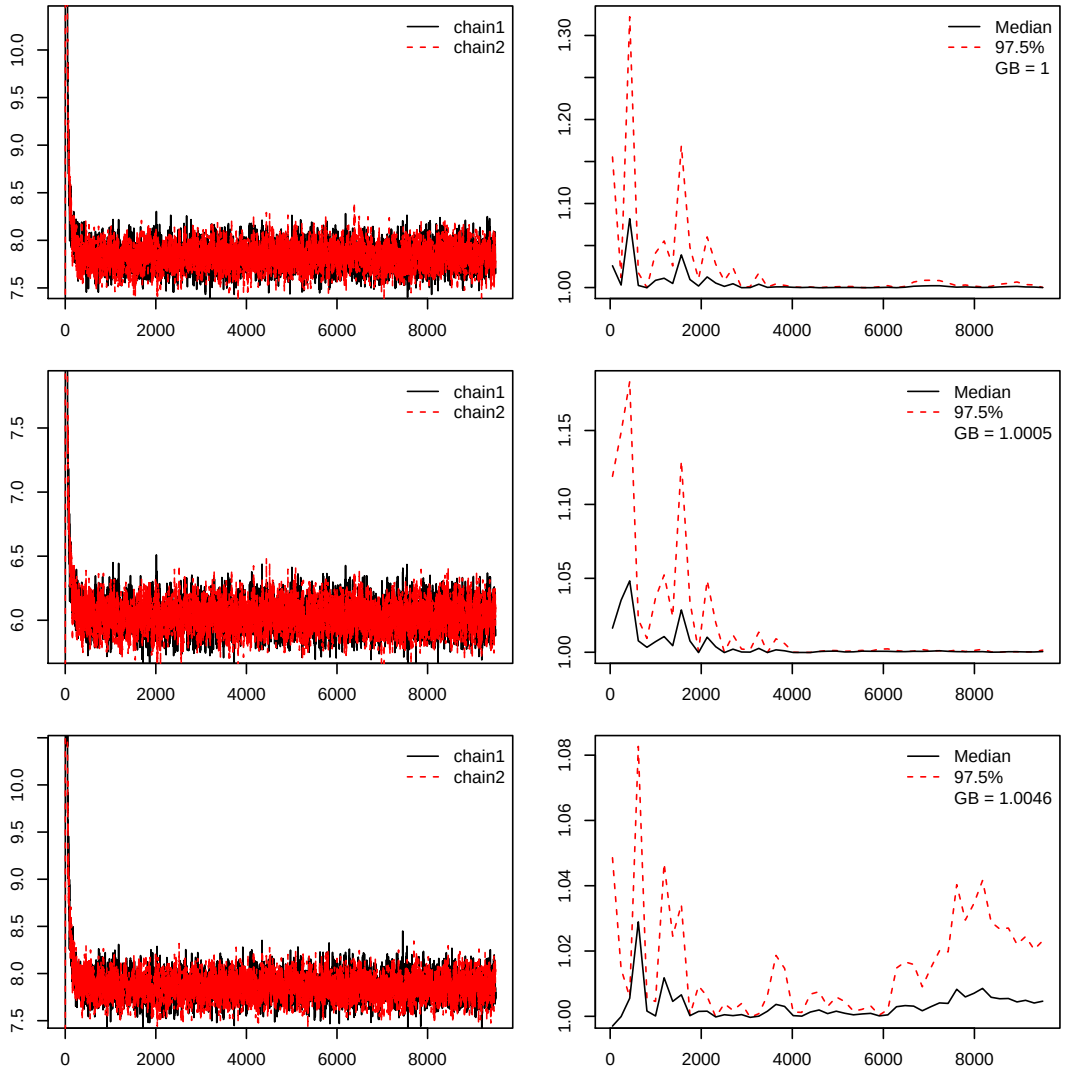


Figure 3.8: Left Panels displays MCMC trace plots representing $\mathbf{\Sigma}$. The right panels display the evolution of Gelman and Rubin's shrink factor as the number of iterations increase and GB is the point estimate of Gelman and Rubin's convergence diagnostic statistic.

3.12 Appendix F

3.12.1 Posterior Summaries for APOE SNP rs405509

Table 3.4: ADNI-1 Study: posterior means and 95% equal-tail credible intervals for all the ROIs and their association with APOE SNP rs405509.

	Spatial Model				**Non-Spatial Model**	
	MCMC		**MFVB**		**MCMC**	
Region	Mean	95% CI	Mean	95% CI	Mean	95% CI
Amygdala volume (L)	0.09	[-0.03,0.22]	0.17	[0.08,0.28]	0.12	[0.02,0.23]
Amygdala volume (R)	0.03	[-0.09,0.16]	0.10	[0.02,0.21]	0.07	[-0.03,0.17]
Cerebral cortex volume (L)	0.03	[-0.08,0.13]	0.10	[0.02,0.21]	0.06	[-0.04,0.16]
Cerebral cortex volume (R)	0.03	[-0.07,0.14]	0.13	[0.03,0.23]	0.06	[-0.03,0.16]
Cerebral white matter volume (L)	0.09	[-0.04,0.22]	0.19	[0.10,0.28]	0.13	[0.03,0.23]
Cerebral white matter volume (R)	0.02	[-0.11,0.15]	0.09	[0.00,0.18]	0.05	[-0.04,0.16]
Entorhinal cortex thickness (L)	0.03	[-0.08,0.15]	0.10	[0.01,0.19]	0.06	[-0.04,0.16]
Entorhinal cortex thickness (R)	0.05	[-0.07,0.17]	0.10	[0.00,0.18]	0.08	[-0.02,0.18]
Fusiform gyrus thickness (L)	0.11	[-0.02,0.23]	0.19	[0.09,0.28]	0.13	[0.03,0.23]
Fusiform gyrus thickness (R)	0.07	[-0.06,0.19]	0.17	[0.06,0.27]	0.10	[-0.01,0.20]
Hippocampus volume (L)	0.12	[-0.02,0.26]	0.22	[0.14,0.34]	0.15	[0.05,0.26]
Hippocampus volume (R)	0.07	[-0.08,0.21]	0.16	[0.06,0.26]	0.10	[0.00,0.20]
Inferior lateral ventricle volume (L)	-0.13	[-0.24,-0.01]	-0.14	[-0.24,-0.05]	-0.08	[-0.18,0.02]
Inferior lateral ventricle volume (R)	-0.08	[-0.20,0.04]	-0.08	[-0.17,0.01]	-0.03	[-0.14,0.07]

Continued on next page

Table 3.4 – *Continued from previous page*

	Spatial Model				Non-Spatial Model	
	MCMC		MFVB		MCMC	
Region	Mean	95% CI	Mean	95% CI	Mean	95% CI
Inferior parietal gyrus thickness (L)	0.08	[-0.02,0.19]	0.15	[0.05,0.24]	0.10	[0.00,0.21]
Inferior parietal gyrus thickness (R)	0.11	[0.01,0.21]	0.20	[0.11,0.29]	0.13	[0.03,0.23]
Inferior temporal gyrus thickness (L)	0.10	[0.00,0.21]	0.18	[0.09,0.26]	0.12	[0.02,0.22]
Inferior temporal gyrus thickness (R)	0.07	[-0.04,0.17]	0.15	[0.06,0.23]	0.10	[-0.01,0.19]
Lateral ventricle volume (L)	-0.07	[-0.17,0.03]	-0.05	[-0.13,0.02]	-0.02	[-0.12,0.08]
Lateral ventricle volume (R)	-0.04	[-0.14,0.06]	-0.01	[-0.09,0.08]	-0.00	[-0.10,0.10]
Caudal anterior cingulate, isthmus cingulate, posterior cingulate, and rostral anterior cingulate mean thickness (L)	0.04	[-0.06,0.14]	0.09	[0.00,0.18]	0.07	[-0.03,0.17]
Caudal anterior cingulate, isthmus cingulate, posterior cingulate, and rostral anterior cingulate mean thickness (R)	0.01	[-0.09,0.11]	0.06	[-0.02,0.15]	0.05	[-0.05,0.15]
Caudal midfrontal, rostral midfrontal, superior frontal, lateral orbitofrontal, and medial orbitofrontal gyri and frontal pole mean thickness (L)	0.10	[-0.02,0.22]	0.18	[0.10,0.27]	0.13	[0.03,0.23]

Continued on next page

Table 3.4 – *Continued from previous page*

	Spatial Model				Non-Spatial Model	
	MCMC		MFVB		MCMC	
Region	Mean	95% CI	Mean	95% CI	Mean	95% CI
Caudal midfrontal, rostral midfrontal, superior frontal, lateral orbitofrontal, and medial orbitofrontal gyri and frontal pole mean thickness (R)	0.12	[0.00,0.25]	0.23	[0.15,0.32]	0.15	[0.05,0.26]
Inferior temporal, middle temporal, and superior temporal gyri mean thickness (L)	0.10	[-0.01,0.21]	0.17	[0.08,0.26]	0.12	[0.02,0.22]
Inferior temporal, middle temporal, and superior temporal gyri mean thickness (R)	0.10	[-0.01,0.21]	0.20	[0.11,0.30]	0.13	[0.02,0.22]
Fusiform, parahippocampal, and lingual gyri, temporal pole and transverse temporal pole mean thickness (L)	0.12	[0.01,0.24]	0.22	[0.13,0.32]	0.14	[0.04,0.24]
Fusiform, parahippocampal, and lingual gyri, temporal pole and transverse temporal pole mean thickness (R)	0.10	[-0.01,0.21]	0.20	[0.11,0.29]	0.12	[0.02,0.22]

Continued on next page

Table 3.4 – *Continued from previous page*

	Spatial Model				Non-Spatial Model	
	MCMC		MFVB		MCMC	
Region	Mean	95% CI	Mean	95% CI	Mean	95% CI
Inferior and superior parietal gyri, supramarginal gyrus, and precuneus mean thickness (L)	0.09	[-0.02,0.21]	0.16	[0.07,0.25]	0.12	[0.02,0.22]
Inferior and superior parietal gyri, supramarginal gyrus, and precuneus mean thickness (R)	0.11	[0.00,0.22]	0.20	[0.11,0.29]	0.13	[0.03,0.23]
Precentral and postcentral gyri mean thickness (L)	0.10	[-0.01,0.20]	0.16	[0.08,0.25]	0.12	[0.02,0.22]
Precentral and postcentral gyri mean thickness (R)	0.09	[-0.02,0.20]	0.17	[0.09,0.26]	0.12	[0.01,0.22]
Inferior temporal, middle temporal, superior temporal, fusiform, parahippocampal, and lingual gyri, temporal pole and transverse temporal pole mean thickness (L)	0.11	[0.01,0.22]	0.20	[0.11,0.28]	0.13	[0.03,0.23]

Continued on next page

Table 3.4 – *Continued from previous page*

	Spatial Model				Non-Spatial Model	
	MCMC		MFVB		MCMC	
Region	Mean	95% CI	Mean	95% CI	Mean	95% CI
Inferior temporal, middle temporal, superior temporal, fusiform, parahippocampal, and lingual gyri, temporal pole and transverse temporal pole mean thickness (R)	0.10	[0.00,0.21]	0.21	[0.12,0.29]	0.12	[0.02,0.23]
Middle temporal gyrus thickness (L)	0.08	[-0.03,0.19]	0.15	[0.06,0.23]	0.11	[0.01,0.21]
Middle temporal gyrus thickness (R)	0.09	[-0.01,0.20]	0.18	[0.11,0.27]	0.12	[0.02,0.22]
Parahippocampal gyrus thickness (L)	0.01	[-0.10,0.11]	0.09	[-0.00,0.21]	0.03	[-0.07,0.14]
Parahippocampal gyrus thickness (R)	0.07	[-0.03,0.18]	0.18	[0.08,0.29]	0.09	[-0.01,0.19]
Postcentral gyrus thickness (L)	0.13	[0.03,0.24]	0.23	[0.13,0.33]	0.14	[0.03,0.23]
Postcentral gyrus thickness (R)	0.09	[-0.01,0.20]	0.18	[0.08,0.28]	0.11	[0.01,0.21]
Posterior cingulate thickness (L)	0.05	[-0.07,0.16]	0.11	[0.02,0.20]	0.07	[-0.03,0.17]
Posterior cingulate thickness (R)	0.07	[-0.04,0.18]	0.15	[0.05,0.24]	0.10	[0.00,0.20]
Precentral gyrus thickness (L)	0.06	[-0.05,0.17]	0.10	[0.02,0.21]	0.09	[-0.01,0.19]
Precentral gyrus thickness (R)	0.08	[-0.03,0.19]	0.15	[0.06,0.25]	0.11	[0.01,0.21]
Precuneus thickness (L)	0.08	[-0.02,0.19]	0.14	[0.06,0.23]	0.10	[0.00,0.20]
Precuneus thickness (R)	0.09	[-0.01,0.19]	0.17	[0.08,0.25]	0.11	[0.01,0.21]

Continued on next page

Table 3.4 – *Continued from previous page*

	Spatial Model				**Non-Spatial Model**	
	MCMC		**MFVB**		**MCMC**	
Region	Mean	95% CI	Mean	95% CI	Mean	95% CI
Superior frontal gyrus thickness (L)	0.10	[0.00,0.20]	0.16	[0.08,0.25]	0.12	[0.02,0.22]
Superior frontal gyrus thickness (R)	0.12	[0.02,0.22]	0.19	[0.11,0.28]	0.14	[0.04,0.24]
Superior parietal gyrus thickness (L)	0.07	[-0.02,0.18]	0.13	[0.05,0.21]	0.10	[0.00,0.20]
Superior parietal gyrus thickness (R)	0.08	[-0.02,0.18]	0.14	[0.06,0.22]	0.11	[0.01,0.21]
Supramarginal gyrus thickness (L)	0.13	[0.02,0.23]	0.21	[0.13,0.29]	0.14	[0.04,0.25]
Supramarginal gyrus thickness (R)	0.14	[0.04,0.24]	0.24	[0.17,0.33]	0.15	[0.05,0.25]
Superior temporal gyrus thickness (L)	0.07	[-0.03,0.18]	0.14	[0.06,0.22]	0.10	[0.00,0.20]
Superior temporal gyrus thickness (R)	0.10	[-0.01,0.22]	0.22	[0.14,0.31]	0.13	[0.03,0.23]
Temporal Pole thickness (L)	0.03	[-0.07,0.13]	0.09	[0.02,0.18]	0.06	[-0.04,0.16]
Temporal Pole thickness (R)	-0.02	[-0.11,0.08]	0.06	[-0.01,0.15]	0.02	[-0.08,0.12]

Chapter 4

Random Tessellation Forests

4.1 Introduction

Bayesian nonparametric models provide flexible and accurate priors by allowing the dimensionality of the parameter space to scale with dataset sizes (Ferguson, 1973). The Mondrian process (MP) is a Bayesian nonparametric prior for space partitioning and provides a Bayesian view of decision trees and random forests (Roy and Teh, 2008; Kemp et al., 2006). Inference for the MP is conducted by recursive and random axis-aligned cuts in the domain of the observed data, partitioning the space into a hierarchical tree of hyper-rectangles.

The MP is appropriate for multi-dimensional data, and it is self-consistent (*i.e.*, it is a projective distribution), meaning that the prior distribution it induces on a subset of a domain is equal to the marginalisation of the prior over the complement of that subset. Self-consistency is required in Bayesian nonparametric models in order to insure correct inference, and prevent any bias arising from sampling and sample population size. Recent advances in MP methods for Bayesian nonparametric space partitioning include online methods (Lakshminarayanan et al., 2014), and particle Gibbs inference for MP additive regression trees (Lakshminarayanan et al., 2015). These methods achieve high predictive accuracy, with improved efficiency. However, the axis-aligned nature of the decision boundaries of the MP restricts its flexibility, which could lead to failure in capturing inter-dimensional dependencies in the domain.

Recently, advances in Bayesian nonparametrics have been developed to allow more flexible non-axis aligned partitioning. The Ostomachion process (OP) was introduced to generalise the MP and allow non-axis aligned cuts. The OP is defined for two dimensional data domains (Fan et al., 2016). In the OP, the angle and position of each cut is randomly sampled from a specified distribution. However, the OP is not self-consistent, and so the binary space partitioning-tree (BSP) process (Fan et al., 2018) was introduced to modify the cut distribution of the OP in order to recover self-consistency. The main limitation of the OP and the BSP is that they are not defined for dimensions larger than two (*i.e.*, they are restricted to data with two predictors). To relax this constraint, in Fan et al. (2019) a self-consistent version of the BSP was extended to arbitrarily dimensioned space (called the BSP-forest). But for this process each cutting hyperplane is axis-aligned in all but two

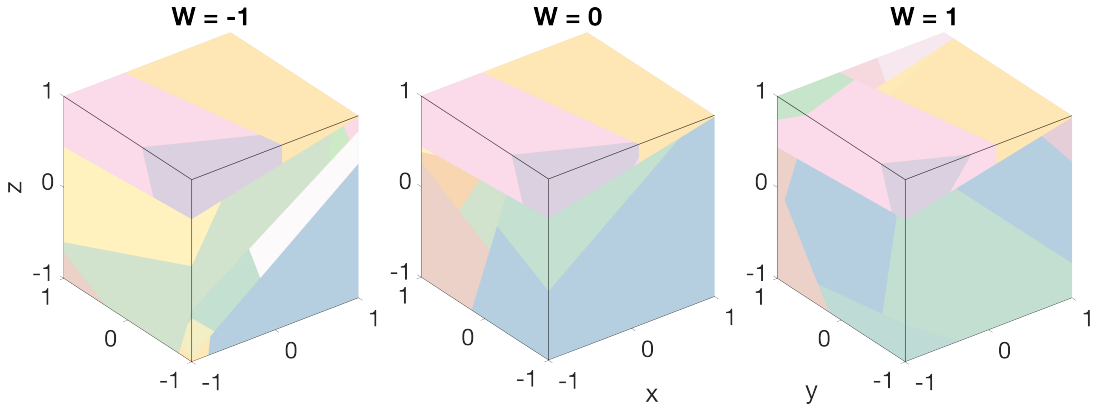


Figure 4.1: A draw from the uRTP prior with domain given by a four dimensional hypercube $(x, y, z, w) \in [-1, 1]^4$. Intersections of the draw and the three dimensional cube are shown for $w = -1, 0, 1$. Colours indicate polytope identity, and are randomly assigned.

dimensions (with non-axis alignment allowed only in the remaining two dimensions, following the specification of the two dimensional BSP). Alternative constructions of non-axis aligned partitioning for two dimensional spaces and non-Bayesian methods involving sparse linear combinations of predictors or canonical correlation have also been proposed as random forest generalisations (George, 1987; Tomita et al., 2015; Rainforth and Wood, 2015).

In this work, we propose the Random Tessellation Process (RTP), a framework for describing Bayesian nonparametric models based on cutting multi-dimensional Euclidean space. We consider four versions of the RTP, including a generalisation of the Mondrian process with non-axis aligned cuts (a sample from this prior is shown in Figure 4.1), a formulation of the Mondrian process as an RTP, and weighted versions of these two methods (shown in Figure 4.2). By virtue of their construction, all versions of the RTP are self-consistent, and are based on the theory of stable iterated tessellations in stochastic geometry (Nagel and Weiss, 2005). The partitions induced by the RTP prior are described by a set of polytopes.

We derive a sequential Monte Carlo (SMC) algorithm (Doucet et al., 2000) for RTP inference which takes advantage of the hierarchical structure of the generating process for the polytope tree. We also propose a random forest version of RTPs, which we refer to as Random Tessellation Forests (RTFs). We apply our proposed model to simulated data and several gene expression datasets, and demonstrate its effectiveness compared to other modern machine learning methods.

4.2 Methods

Suppose we observe a dataset $(\boldsymbol{v}_1, z_1), \ldots, (\boldsymbol{v}_n, z_n)$, for a classification task in which $\boldsymbol{v}_i \in \mathbb{R}^d$ are predictors and $z_i \in \{1, \ldots, K\}$ are labels (with K levels, $K \in \mathbb{N}_{>1}$). Bayesian nonparametric models based on partitioning the predictors proceed by placing a prior on aspects of the partition, and associating likelihood parameters with the blocks of the partition. Inference is then done on the joint posterior of the parameters and the structure of the partition. In this section, we develop the

RTP: a unifying framework that covers and extends such Bayesian nonparametric models through a prior on partitions of $(\boldsymbol{v}_1, z_1), \ldots, (\boldsymbol{v}_n, z_n)$ induced by tessellations.

4.2.1 The Random Tessellation Process

A tessellation Y of a bounded domain $W \subset \mathbb{R}^d$ is a finite collection of closed polytopes such that the union of the polytopes is all of W, and such that the polytopes have pairwise disjoint interiors (Chiu et al., 2013). We denote tessellations of W by $Y(W)$ or the symbol $\triangle$. A polytope is an intersection of finitely many closed half-spaces. In this work we will assume that all polytopes are bounded and have nonempty interior. An RTP $Y_t(W)$ is a tessellation-valued right-continuous Markov jump process (MJP) defined on $[0, \tau]$ (we refer to the t-axis as time), in which events are cuts (specified by hyperplanes) of the tessellation's polytopes, and τ is a prespecified budget (Berger, 2012). In this work we assume that all hyperplanes are affine (i.e., , they need not pass through the origin).

The initial tessellation $Y_0(W)$ contains a single polytope given by the convex hull of the observed predictors in the dataset: $W = \text{hull}\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n\}$ (the operation hull A denotes the convex hull of the set A). In the MJP for the random tessellation process, each polytope has an exponentially distributed lifetime, and at the end of a polytope's lifetime, the polytope is replaced by two new polytopes. The two new polytopes are formed by drawing a hyperplane that intersects the interior of the old polytope, and then intersecting the old polytope with each of the two closed half-spaces bounded by the drawn hyperplane. We refer to this operation as cutting a polytope according to the hyperplane. These cutting events continue until the prespecified budget τ is reached.

Let H be the set of hyperplanes in $\mathbb{R}^d$. Every hyperplane $h \in H$ can be written uniquely as the set of points $\{P : \langle \vec{n}, P - u\ \vec{n} \rangle = 0\}$, such that $\vec{n} \in S^{d-1}$ is a normal vector of h, and $u \in \mathbb{R}_{\geq 0}$ ($u \geq 0$). Here S^{d-1} is the unit $(d-1)$-sphere (i.e., , $S^{d-1} = \{\vec{n} \in \mathbb{R}^d : \| \vec{n} \| = 1\}$). Thus, there is a bijection $\varphi : S^{d-1} \times \mathbb{R}_{\geq 0} \longmapsto H$ by $\varphi(\vec{n}, u) = \{P : \langle \vec{n}, P - u\ \vec{n} \rangle = 0\}$, and therefore a measure Λ on H is induced by any measure $\Lambda \circ \varphi$ on $S^{d-1} \times \mathbb{R}_{\geq 0}$ through this bijection (Kingman, 1996; Chiu et al., 2013).

Nagel and Weiss (2005) describe a random tessellation associated with a measure Λ on H through a tessellation-valued MJP Y_t such that the rate of the exponential distribution for the lifetime of a polytope $a \in Y_t$ is $\Lambda([a])$ (here and throughout this work, $[a]$ denotes the set of hyperplanes in $\mathbb{R}^d$ that intersect the interior of a), and the hyperplane for the cutting event for a polytope a is sampled according to the probability measure $\Lambda(\cdot \cap [a])/\Lambda([a])$. We use this construction as the prior for RTPs, and describe their generative process in Algorithm 5. This algorithm is equivalent to the first algorithm listed in Nagel and Weiss (2005).

Self-consistency of Random Tessellation Processes

From *Theorem 1* in Nagel and Weiss (2005), if the measure Λ is invariant with respect to translation (i.e., , $\Lambda(A) = \Lambda(\{h + x : h \in A\})$ for all measurable subsets $A \subset H$ and $x \in \mathbb{R}^d$), and if a set of d hyperplanes with orthogonal normal vectors is contained in the support of Λ, then

Algorithm 5 Generative Process for RTPs

Inputs: a) Bounded domain W, b) RTP measure Λ on H,
c) prespecified budget τ.
Outputs: A realisation of the Random Tessellation Process $(Y_t)_{0 \le t \le \tau}$.
$\tau_0 \leftarrow 0$.
$Y_0 \leftarrow \{W\}$.
while $\tau_0 \le \tau$ **do**
    Sample $\tau' \sim \text{Exp}\Big(\sum_{a \in Y_{\tau_0}} \Lambda([a])\Big)$.
    Set $Y_t \leftarrow Y_{\tau_0}$ for all $t \in (\tau_0, \min\{\tau, \tau_0 + \tau'\}]$.
    Set $\tau_0 \leftarrow \tau_0 + \tau'$.
    if $\tau_0 \le \tau$ **then**
        Sample a polytope a from the set Y_{τ_0} with probability
        proportional to (*w.p.p.t.*) $\Lambda([a])$.
        Sample a hyperplane h from $[a]$ according to the probability
        measure $\Lambda(\cdot \cap [a])/\Lambda([a])$.
        $Y_{\tau_0} \leftarrow (Y_{\tau_0}/\{a\}) \cup \{a \cap h^-, a \cap h^+\}$.
        h^- and h^+ are the h-bounded closed half planes.
    else
        return the tessellation-valued right-continuous MJP sample $(Y_t)_{0 \le t \le \tau}$.

for all bounded domains $W' \subseteq W$, $Y_t(W')$ is equal in distribution to $Y_t(W) \cap W'$. This means that self-consistency holds for the random tessellations associated with such Λ. (Here, for a hyperplane h, $h + x$ refers to the set $\{y + x : y \in h\}$, and for a tessellation Y and a domain W', $Y \cap W'$ is the tessellation $\{a \cap W' : a \in Y\}$.) In Nagel and Weiss (2005), such tessellations are referred to as stable iterated tessellations.

If we assume that $\Lambda \circ \varphi$ is the product measure $\lambda^d \times \lambda_+$, with λ^d symmetric (i.e., , $\lambda^d(A) = \lambda^d(-A)$ for all measurable sets $A \subseteq S^{d-1}$) and further that λ_+ is given by the Lebesgue measure on $\mathbb{R}_{\ge 0}$, then Λ is translation invariant (a proof of this statement is given in Appendix A, *Lemma 1* of the *Supplementary Material*). So, through Algorithm 5 and *Theorem 1* in Nagel and Weiss (2005), any distribution λ^d on the sphere S^{d-1} that is supported on a set of d hyperplanes with orthogonal normal vectors gives rise to a self-consistent random tessellation, and we refer to models based on this self-consistent prior as Random Tessellation Processes (RTPs). We refer to any product measure $\Lambda \circ \varphi = \lambda^d \times \lambda_+$ such these conditions hold (i.e., , λ^d symmetric, Λ supported on d orthogonal hyperplanes and λ_+ given by the Lebesgue measure) as an RTP measure.

Relationship to cutting nonparametric models

If λ^d is the measure associated with a *uniform distribution* on the sphere with respect to the usual Borel sets on S^{d-1} (Halmos, 1974), then the resulting RTP is a generalisation of the Mondrian process with non-axis aligned cuts. We refer to this RTP as the uRTP (for uniform RTP). In this case, λ^d is a probability measure and a normal vector $\vec{n}$ may be sampled according to λ^d by sampling $n_i \sim N(0, 1)$

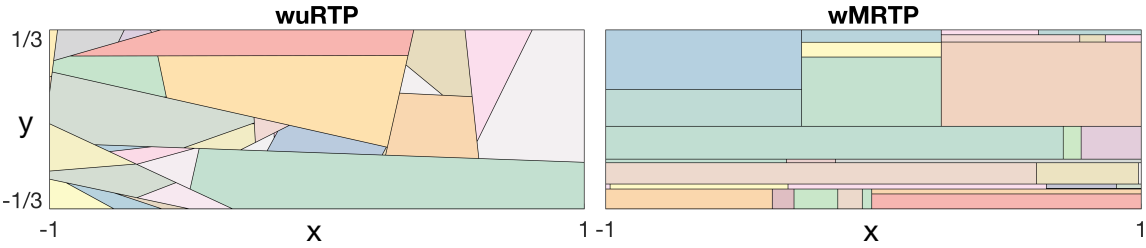


Figure 4.2: Draws from priors *Left)* wuRTP, and *Right)* wMRTP, for the domain given by the rectangle $W = [-1, 1] \times [-1/3, 1/3]$. Weights are given by $\omega_x = 14, \omega_y = 1$, leading to horizontal *(x-axis heavy)* structure in the polygons. Colours are randomly assigned and indicate polygon identity, and black delineates polygon boundaries.

and then setting $\vec{n}_i = n_i/\|n\|$. A draw from the uRTP prior supported on a four dimensional hypercube is displayed in Figure 4.1.

We consider a weighted version of the uniform RTP found by setting λ^d to the measure associated with the distribution on $\vec{n}$ induced by the scheme $n_i \sim N(0, \omega_i^2)$, $\vec{n}_i = n_i/\|n\|$. We refer to this RTP as the wuRTP (weighted uniform RTP), and the wuRTP is parameterised by d weights $\omega_i \in \mathbb{R}_{>0}$. Note that the isotropy of the multivariate Gaussian n implies symmetry of λ^d. Setting the weight ω_i increases the prior probability of cuts orthogonal to the i-th predictor dimension, allowing prior information about the importance of each of the predictors to be incorporated. Figure 4.2*(left)* shows a draw from the wuRTP prior supported on a rectangle.

The Mondrian process itself is an RTP with $\lambda^d = \sum_{v \in \text{poles}(d)} \delta^v$. Here, δ_x is the Dirac delta supported on x, and poles(d) is the set of normal vectors with zeros in all coordinates, except for one of the coordinates (the non-zero coordinate of these vectors can be either -1 or $+1$). We refer to this view of the Mondrian process as the MRTP. If $\lambda^d = \sum_{v \in \text{poles}(d)} \omega_{i(v)} \delta_v$, where ω_i are d axis weights, and $i(v)$ is the index of the nonzero element of v, then we arrive at a weighted version of the MRTP, which we refer to as the wMRTP. Figure 4.2*(right)* displays a draw from the wMRTP prior. The horizontal organisation of the lines in Figure 4.2 arise from the uneven axis weights.

Other nonparametric models based on cutting polytopes may also be viewed in this way. For example, Binary Space Partitioning-Tree Processes (Fan et al., 2018) are uRTPs and wuRTPs restricted to two dimensions, and the generalization of the Binary Space Partitioning-Tree Process in Fan et al. (2019) is an RTP for which λ^d is a sum of delta functions convolved with smooth measures on S^1 projected onto pairs of axes. The Ostomachion process (Fan et al., 2016) does not arise from an RTP: it is not self-consistent, and so by *Theorem 1* in Nagel and Weiss (2005), the OP cannot arise from an RTP measure.

Likelihoods for Random Tessellation Processes

In this section, we illustrate how RTPs can be used to model categorical data. For example, in gene expression data of tissues, the predictors are the amounts of expression of each gene in the tissue (the vector $\boldsymbol{v}_i$ for sample i), and our goal is to predict disease condition (labels z_i). Let $\triangle_t$ be an RTP

on the domain $W = \text{hull}\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n\}$. Let J_t denote the number of polytopes in $\triangle_t$. We let $h(\boldsymbol{v}_i)$ denote a mapping function, which matches the i-th data item to the polytope in the tessellation containing that data item. Hence, $h(\boldsymbol{v}_i)$ takes a value in the set $\{1, \ldots, J_t\}$. We will consider the likelihood arising at time t from the following generative process:

$$\triangle_t \sim \boldsymbol{Y}_t(\boldsymbol{W}), \qquad \boldsymbol{\phi}_j \sim \text{Dirichlet}(\boldsymbol{\alpha}) \text{ for } 1 \leq j \leq J_t, \tag{1}$$

$$z_i | \triangle_t, \boldsymbol{\phi}_{h(\boldsymbol{v}_i)} \sim \text{Multinomial}(\boldsymbol{\phi}_{h(\boldsymbol{v}_i)}) \text{ for } 1 \leq i \leq n. \tag{2}$$

Here $\boldsymbol{\phi}_j = (\phi_{j1}, \ldots, \phi_{jK})$ are parameters of the multinomial distribution with a Dirichlet prior with hyperparameters $\boldsymbol{\alpha} = (\alpha_k)_{1 \leq k \leq K}$. The likelihood function for $\boldsymbol{Z} = (z_i)_{1 \leq i \leq n}$ conditioned on the tessellation $\triangle_t$ and $\boldsymbol{V} = (v_i)_{1 \leq i \leq n}$ and given the hyperparameter $\boldsymbol{\alpha}$ is as follows:

$$P(\boldsymbol{Z} | \triangle_t, \boldsymbol{V}, \boldsymbol{\alpha}) = \int \cdots \int P(\boldsymbol{Z}, \phi | \triangle_t, \boldsymbol{\alpha}) d\boldsymbol{\phi}_1 \cdots d\boldsymbol{\phi}_{J_t} = \prod_{j=1}^{J_t} \frac{B(\boldsymbol{\alpha} + \boldsymbol{m}_j)}{B(\boldsymbol{\alpha})}. \tag{3}$$

Here $B(\cdot)$ is the multivariate beta function, $\boldsymbol{m}_j = (m_{jk})_{1 \leq k \leq K}$ and $m_{jk} = \sum_{i:h(\boldsymbol{v}_i)=j} \delta(z_i = k)$, and $\delta(\cdot)$ is an indicator function with $\delta(z_i = k) = 1$ if $z_i = k$ and $\delta(z_i = k) = 0$ otherwise. We refer to Appendix A, *Lemma 2* of the *Supplementary Material* for the derivation of (3).

4.2.2 Inference for Random Tessellation Processes

Our objective is to infer the posterior of the tessellation at time t, denoted $\pi(\triangle_t | \boldsymbol{V}, \boldsymbol{Z}, \boldsymbol{\alpha})$. We let $\pi_0(\triangle_t)$ denote the prior distribution of $\triangle_t$. Let $P(\boldsymbol{Z} | \triangle_t, \boldsymbol{V}, \boldsymbol{\alpha})$ denote the likelihood (3) of the data given the tessellation $\triangle_t$ and the hyperparameter $\boldsymbol{\alpha}$. By Bayes' rule, the posterior of the tessellation at time t is $\pi(\triangle_t | \boldsymbol{V}, \boldsymbol{Z}, \boldsymbol{\alpha}) = \pi_0(\triangle_t) P(\boldsymbol{Z} | \triangle_t, \boldsymbol{V}, \boldsymbol{\alpha}) / P(\boldsymbol{Z} | \boldsymbol{V}, \boldsymbol{\alpha})$.

Here $P(\boldsymbol{Z} | \boldsymbol{V}, \boldsymbol{\alpha})$ is the marginal likelihood given data $\boldsymbol{Z}$. This posterior distribution is intractable, and so in *Section* 4.2.2 we introduce an efficient SMC algorithm for conducting inference on $\pi(\triangle_t)$. The proposal distribution for this SMC algorithm involves draws from the RTP prior, and so in *Section* 4.2.2, we describe a rejection sampling scheme for drawing a hyperplane from the probability measure $\Lambda(\cdot \cap [a]) / \Lambda([a])$ for a polytope a. We provide some optimizations and approximations used in this SMC algorithm in *Section* 4.2.2.

Sampling cutting hyperplanes for Random Tessellation Processes

Suppose that a is a polytope, and $\Lambda \circ \varphi$ is an RTP measure such that $\Lambda(\varphi(\cdot)) = (\lambda^d \times \lambda_+)(\cdot)$. We wish to sample a hyperplane according to the probability measure $\Lambda(\cdot \cap [a]) / \Lambda([a])$. We note that if $B_r(x)$ is the smallest closed d-ball containing a (with radius r and centre x), then $[a]$ is contained in $[B_r(x)]$. If we can sample a normal vector $\vec{n}$ according to λ^d, then we can sample a hyperplane according to $\Lambda(\cdot \cap [a]) / \Lambda([a])$ through the following rejection sampling scheme.

- Step 1) Sample $\vec{n}$ according to λ^d.

- Step 2) Sample $u \sim \text{Uniform}[0, r]$.
- Step 3) If the hyperplane $h = x + \{P : \langle \vec{n}, P - u\ \vec{n} \rangle\}$ intersects a, then `RETURN` h. Otherwise, `GOTO` Step 1).

Note that in Step 3, the set $\{P : \langle \vec{n}, P - u\ \vec{n} \rangle\}$ is a hyperplane intersecting the ball $B_r(0)$ centred at the origin, and so translation of this hyperplane by x yields a hyperplane intersecting the ball $B_r(x)$. When this scheme is applied to the uRTP or wuRTP, in Step 1 $\vec{n}$ is sampled from the uniform distribution on the sphere or the appropriate isotropic Gaussian. And for the MRTP or wMRTP, $\vec{n}$ is sampled from the discrete distributions given in *Section* 4.2.1.

Optimizations and approximations

We use three methods to decrease the computational requirements and complexity of inference based on RTP posteriors. First, we replace all polytopes with convex hulls formed by intersecting the polytopes with the dataset predictors. Second, in determining the rates of the lifetimes of polytopes, we approximate $\Lambda([a])$ with the measure $\Lambda([\cdot])$ applied to the smallest closed ball containing a. Third, we use a pausing condition so that no cuts are proposed for polytopes for which the labels of all predictors in that polytope are the same label.

Convex hull replacement. In our posterior inference, if $a \in Y$ is cut according to the hyperplane h, we consider the resulting tessellation to be $Y/\{a\} \cup \{\text{hull}(a \cap \boldsymbol{V} \cap h^+), \text{hull}(a \cap \boldsymbol{V} \cap h^-)\}$. Here h^+ and h^- are the two closed half planes bounded by h, and / is the set minus operation. This requires a slight loosening of the definition of a tessellation Y of a bounded domain W to allow the union of the polytopes of a tessellation Y to be a strict subset of W such that $\boldsymbol{V} \subseteq \cup_{a \in Y} a$.

In our computations, we do not need to explicitly compute these convex hulls, and instead for any polytope b, we store only $b \cap \boldsymbol{V}$, as this is enough to determine whether or not a hyperplane h intersects $\text{hull}(b \cap \boldsymbol{V})$. This membership check is the only geometric operation required to sample hyperplanes intersecting b according to the rejection sampling scheme from *Section* 4.2.2. By the self-consistency of RTPs, this has the effect of marginalizing out MJP events involving cuts that do not further separate the predictors in the dataset. This also obviates the need for explicit computation of the facets of polytopes, significantly simplifying the codebase of our implementation of inference.

After this convex hull replacement operation, a data item in the test dataset may not be contained in any polytope, and so to conduct posterior inference we augment the training dataset with a version of the testing dataset in which the label is missing, and then marginalise the missing label in the likelihood described in *Section* 4.2.1.

Spherical approximation. Every hyperplane intersecting a polytope a also intersects a closed ball containing a. Therefore, for any RTP measure Λ, $\Lambda([a])$ is upper bounded by $\Lambda([B(r_a)])$. Here r_a is the radius of the smallest closed ball containing a. We approximate $\Lambda([a]) \simeq \Lambda([B(r_a)])$ for use in polytope lifetime calculations in our uRTP inference and we do not compute $\Lambda([a])$ exactly. For the uRTP and wRTP, $\Lambda([B(r_a)]) = r_a$. A proof of this is given in Appendix A, *Lemma 3* of the

Supplementary Material. For the MRTP and wMRTP, $\Lambda([a])$ can be computed exactly (Roy and Teh, 2008).

Pausing condition. In our posterior inference, if $z_i = z_j$ for all i, j such that $\boldsymbol{v}_i, \boldsymbol{v}_j \in a$, then we *pause* the polytope a and no further cuts are performed on this polytope. This improves computational efficiency without affecting inference, as cutting such a polytope cannot further separate labels. This was done in recent work for Mondrian processes (Lakshminarayanan et al., 2014) and was originally suggested in the Random Forest reference implementation (Breiman et al., 1984).

Sequential Monte Carlo for Random Tessellation Process inference

Algorithm 6 SMC for inferring RTP posteriors

1: **Inputs:** a) Training dataset $\boldsymbol{V}, \boldsymbol{Z}$, b) RTP measure Λ on H, c) prespecified budget τ, d) likelihood hyperparameter α.
2: **Outputs:** Approximate RTP posterior $\sum_{m=1}^{M} \varpi_m \delta_{\triangle_{\tau,m}}$ at time τ. (ϖ_m are particle weights.)
3: Set $\tau_m \leftarrow 0$, for $m = 1, \ldots, M$.
4: Set $\triangle_{0,m} \leftarrow \{\text{hull } \boldsymbol{V}\}$, $\varpi_m \leftarrow 1/M$, for $m = 1, \ldots, M$.
5: **while** $\min\{\tau_m\}_{m=1}^{M} < \tau$ **do**
6: Resample $\triangle'_{\tau_m,m}$ from $\{\triangle_{\tau_m,m}\}_{m=1}^{M}$ *w.p.p.t.* $\{\varpi_m\}_{m=1}^{M}$, for $m = 1, \ldots, M$.
7: Set $\triangle_{\tau_m,m} \leftarrow \triangle'_{\tau_m,m}$, for $m = 1, \ldots, M$.
8: Set $\varpi_m \leftarrow 1/M$, for $m = 1, \ldots, M$.
9: **for** $m \in \{m : m = 1, \ldots, M \text{ and } \tau_m < \tau\}$ **do**
10: Sample $\tau' \sim \text{Exp}\left(\sum_{a \in \triangle_{\tau_m,m}} r_a\right)$. ($r_a$ is the radius of the smallest closed ball containing a.)
11: Set $\triangle_{t,m} \leftarrow \triangle_{\tau_m,m}$, for all $t \in (\tau_m, \min\{\tau, \tau_m + \tau'\}]$.
12: **if** $\tau_m + \tau' \leq \tau$ **then**
13: Sample a from the set $\triangle_{\tau_m,m}$ *w.p.p.t.* r_a.
14: Sample h from $[a]$ according to $\Lambda(\cdot \cap [a])/\Lambda([a])$ using *Section* 4.2.2.
15: Set $\triangle_{\tau_m,m} \leftarrow (\triangle_{\tau_m,m}/\{a\}) \cup \{\text{hull}(\boldsymbol{V} \cap a \cap h^-), \text{hull}(\boldsymbol{V} \cap a \cap h^+)\}$.
16: Set $\varpi_m \leftarrow \varpi_m P(\boldsymbol{Z}|\triangle_{\tau_m,m}, \boldsymbol{V}, \alpha)/P(\boldsymbol{Z}|\triangle'_{\tau_m,m}, \boldsymbol{V}, \alpha)$ according to (3).
17: **else**
18: Set $\triangle_{t,m} \leftarrow \triangle_{\tau_m,m}$, for $t \in (\tau_m, \tau]$.
19: Set $\tau_m \leftarrow \tau_m + \tau'$.
20: Set $\mathcal{Z} \leftarrow \sum_{m=1}^{M} \varpi_m$.
21: Set $\varpi_m \leftarrow \varpi_m/\mathcal{Z}$, for $m = 1, \ldots, M$.
22: **return** the particle approximation $\sum_{m=1}^{M} \varpi_m \delta_{\triangle_{\tau,m}}$.

We provide an SMC method (Algorithm 6) with M particles, to approximate the posterior distribution $\pi(\triangle_t)$ of an RTP conditioned on $\boldsymbol{Z}, \boldsymbol{V}$, and given an RTP measure Λ and a hyperparameter $\boldsymbol{\alpha}$ and a prespecified budget τ. Our algorithm iterates between three steps: resampling particles (Algorithm 6, line 7), propagation of particles (Algorithm 6, lines 13-15) and weighting of particles (Algorithm 6, line 16). At each SMC iteration, we sample the next MJP events using the spherical approximation of $\Lambda([\cdot])$ described in *Section* 4.2.2. For brevity, the pausing condition described in *Section* 4.2.2 is omitted from Algorithm 6.

In our experiments, after using Algorithm 6 to yield a posterior estimate $\sum_{m=1}^{M} \varpi_m \delta_{\triangle_{\tau,m}}$, we select the tessellation $\triangle_{\tau,m}$ with the largest weight ϖ_m (i.e., , we do not conduct resampling at the last SMC iteration). We then compute posterior probabilities of the test dataset labels using the particle $\triangle_{\tau,m}$. This method of not resampling after the last SMC iteration is recommended in (Chopin, 2004) for lowering asymptotic variance in SMC estimates.

The computational complexity of Algorithm 6 depends on the number of polytopes in the tessellations, and the organization of the labels within the polytopes. The more linearly separable the dataset is, the sooner the pausing conditions are met. The complexity of computing the spherical approximation in *Section* 4.2.2 (the radius r_a) for a polytope a is $O(|V \cap a|^2)$, where $|\cdot|$ denotes set cardinality.

Prediction with Random Tessellation Forests

Random forests are commonly used in machine learning for classification and regression problems (Breiman, 2001). A random forest is represented by an ensemble of decision trees, and predictions of test dataset labels are combined over all decision trees in the forest. To improve the performance of our methods, we consider random forest versions of RTPs (which we refer to as RTFs: uRTF, wuRTF, MRTF, wMRTF are random forest versions of the uRTP, MRTP and their weighted versions *resp*.). We run Algorithm 6 independently T times, and predict labels using the modes. Differing from Breiman (2001), we do not use bagging.

Lakshminarayanan et al. (2015) consider an efficient Mondrian forest in which likelihoods are dropped from the SMC sampler and cutting is done independent of likelihood. This method follows recent theory for random forests (Geurts et al., 2006). We consider this method (by dropping line 16 of Algorithm 6) and refer to the implementations of this method as the uRTF.i and MRTF.i (*i* for likelihood *i*ndependence).

4.3 Experiments

In *Section* 4.3.1, we explore a simulation study that shows differences among uRTP and MRTP, and some standard machine learning methods. Variations in gene expression across tissues in brain regions play an important role in disease conditions. In *Section* 4.3.2, we examine predictions of a variety of RTF models for gene expression data. For all our experiments, we set the likelihood hyperparameters for the RTPs and RTFs to the empirical estimates α_k to $n_k/1000$. Here $n_k = \sum_{i=1}^{n} \delta(z_i = k)$. In all of our experiments, for each train/test split, we allocate 60% of the data items at random to the training set.

An implementation of our methods (released under the open source BSD 2-clause license) and a software manual are provided in the *Supplementary Material*.

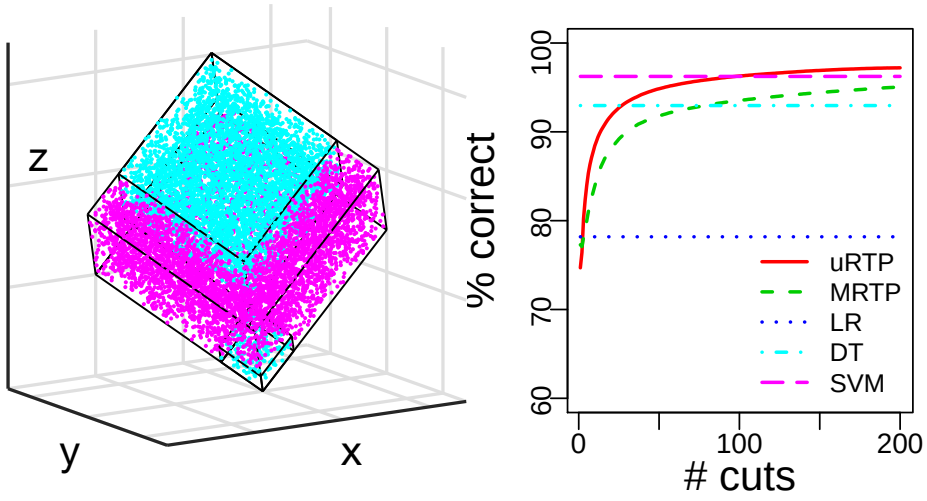


Figure 4.3: *Left)* A view of the Mondrian cube, with cyan indicating label 1, magenta indicating label 2, and black delineating label boundaries. *Right)* Percent correct versus number of cuts for predicting Mondrian cube test dataset, with uRTP, MRTP and a variety of baseline methods.

4.3.1 Simulations on the Mondrian cube

We consider a simulated three dimensional dataset designed to exemplify the difference between axis-aligned and non-axis aligned models. We refer to this dataset as the Mondrian cube, and we investigate the performance of uRTP and the MRTP on this dataset, along with some standard machine learning approaches, varying the number of cuts in the processes. The Mondrian cube dataset is simulated as follows: first, we sample 10,000 points uniformly in the cube $[0, 1]^3$. Points falling in the cube $[0, 0.25]^3$ or the cube $[0.25, 1]^3$ are given label 1, and the remaining points are given label 2. Then, we centre the points and rotate all of the points by the angles $\frac{\pi}{4}$ and $-\frac{\pi}{4}$ about the x-axis and y-axis respectively, creating a dataset organised on diagonals. In Figure 4.3*(left)*, we display a visualization of the Mondrian cube dataset, wherein points are colored by their label. We apply the SMC algorithm to the Mondrian cube data, with 50 random train/test splits. For each split, we run 10 independent copies of the uRTP and MRTP and take the mode of their results, and we also examine the accuracy of logistic regression (LR), a decision tree (DT) and a support vector machine (SVM).

Figure 4.3*(right)* shows that the percent correct for the uRTP and MRTP both increase as the number of cuts increases, and plateaus when the number of cuts becomes larger (greater than 25). Even though the uRTP has lower accuracy at the first cut, it starts dominating the MRTP after the second cut. Overall, in terms of percent correct, with any number of cuts > 105, a sign test indicates that the uRTP performs significant better than all other methods at nominal significance, and the MRTP performs significant better than DT and LR for any number of cuts > 85 at nominal significance.

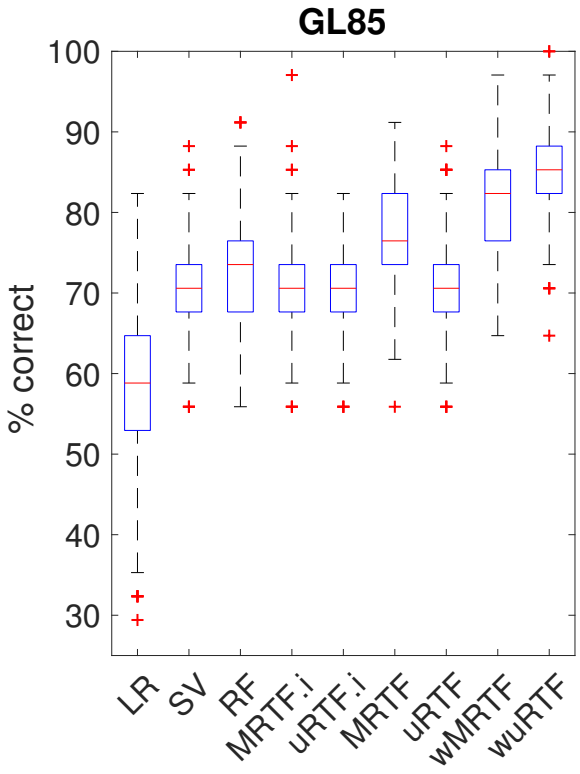


Figure 4.4: Box plot showing wuRTF and wMRTF improvements for *GL85*, and generally best performance for wuRTF method (with sign test p-value of 3.2×10^{-9} *vs* wMRTF). Medians, quantiles and outliers beyond 99.3% coverage are indicated.

4.3.2 Experiment on gene expression data in brain tissue

We evaluate a variety of RTPs and some standard machine learning methods on a glioblastoma tissue dataset *GSE83294* (Freije et al., 2004), which includes 22,283 gene expression profiles for 85 astrocytomas (26 diagnosed as grade III and 59 as grade IV). We also examine schizophrenia brain tissue datasets: *GSE21935* (Barnes et al., 2011), in which 54,675 gene expression in the superior temporal cortex is recorded for 42 subjects, with 23 cases (with schizophrenia), and 19 controls (without schizophrenia), and dataset *GSE17612* (Maycox et al., 2009), a collection of 54,675 gene expressions from samples in the anterior prefrontal cortex (i.e., , a different brain area from *GSE21935*) with 28 schizophrenic subjects, and 23 controls. We refer to these datasets as *GL85*, *SCZ42* and *SCZ51*, respectively.

We also consider a combined version of *SCZ42* and *SCZ51* (in which all samples are concatenated), which we refer to as *SCZ93*. For *GL85* the labels are the astrocytoma grade, and for *SCZ42*, *SCZ51* and *SCZ93* the labels are schizophrenia status. We use principal components analyais (PCA) in preprocessing to replace the predictors of each data item (a set of gene expressions) with its scores on a full set of principal components (PCs): i.e., , 85 PCs for *GL85*, 42 PCs in *SCZ42* and 51 PCs in *SCZ51*. We then scale the PCs. We consider 200 test/train splits for each dataset. These datasets

were acquired from NCBI's Gene Expression Omnibus and were are released under the *Open Data Commons Open Database License*. We provide test/train splits of the PCA preprocessed datasets in the *Supplementary Material*.

Through this preprocessing, the j-th predictor is the score vector of the j-th principal component. For the weighted RTFs (the wuRTF and wMRTF), we set the weight of the j-th predictor to be proportional to the variance explained by the j-th PC (σ_j^2): $\omega_j = \sigma_j^2$. We set the number of trees in all of the random forests to 100, which is the default in R's `randomForest` package (Liaw and Wiener, 2002). For the all RTFs, we set the budget $\tau = \infty$, as is done in Lakshminarayanan et al. (2014).

4.4 Results

We compare percent correct for the wuRTF, uRTF, uRTF.i, and the Mondrian Random Tessellation Forests wMRTF, MRTF and MRTF.i, a random forest (RF), logistic regression (LR), a support vector machine (SVM) and a baseline (BL) in which the mode of the training set label is always predicted (Meyer et al., 2019; Liaw and Wiener, 2002). Mean percent correct and sign tests for all of these experiments are reported in Table 4.1 and box plots for the *GL85* experiment reported in Figure 4.4. We observe that the increase in accuracy of wuRTFs achieves nominal significance over other methods on *GL85*. For datasets *SCZ42*, *SCZ51* and *SCZ93*, the performance of the RTFs is comparable to that of logistic regression and random forests. For all datasets we consider, RTFs have higher accuracy than SVMs (with nominal significance). Boxplots with accuracies for the datasets *SCZ42*, *SCZ51* and *SCZ93* are provided in Appendix B, Supplementary Figure 1 of the *Supplementary Material*. Results of a conservative pairwise sign test performed between each pair of methods on each dataset, standard deviation of the percent correct, and mean runtime across different methods for all these four datasets are reported in Supplementary Tables 1, 2, and 3 in Appendix B of the *Supplementary Material*.

4.5 Discussion

Rejection sampling, and computation of the radii of the approximating spheres are computational bottlenecks of RTP methods. This leads to longer running times as dimension grows as indicated in Supplementary Table 3. Supplementary Algorithm 1 replaces the spherical approximation by a Monte Carlo approximation. This algorithm may improve runtime and accuracy.

There are many directions for future work in RTPs including improved SMC sampling for MJPs as in Hajiaghayi et al. (2014), hierarchical likelihoods and online methods as in Lakshminarayanan et al. (2015), analysis of minimax convergence rates (Mourtada et al., 2018), and extensions using Bayesian additive regression trees (Chipman et al., 2010; Lakshminarayanan et al., 2015). We could

Table 4.1: Comparison of mean percent correct on gene expression datasets over 200 random train/test splits across different methods. Nominal statistical significance (p-value < 0.05) is ascertained by a sign test in which ties are broken in a conservative manner. Tests are performed between the top method and each other method. Bold values indicate the top method and all methods statistically indistinguishable from the top method according to nominal significance. Largest nominally significant improvement is seen for wuRTF on *GL85*, and wuRTF is significantly better than other methods for this dataset. The wMRTF and wuRTF have largest mean percent correct for *SCZ51* and *SCZ93* but are not statistically distinguishable from *RF* or *LR* for those datasets.

Dataset	BL	LR	SVM	RF	MRTF.i	uRTF.i	MRTF	uRTF	wMRTF	wuRTF
GL85	70.34	58.13	70.34	73.01	70.74	70.06	77.09	70.60	80.57	**84.90**
SCZ42	46.68	**57.65**	46.79	51.76	49.56	48.50	49.91	47.71	**53.12**	**53.97**
SCZ51	46.55	51.15	46.67	**57.38**	52.55	48.58	**57.95**	44.70	**58.12**	49.05
SCZ93	48.95	**53.05**	50.15	52.45	50.23	50.24	51.80	50.34	**53.12**	**54.99**

also consider applications of RTPs to data that naturally displays tessellation and cracking, such as sea ice (Godlovitch, 2011).

4.6 Conclusion

We have described a framework for viewing Bayesian nonparametric methods based on space partitioning as Random Tessellation Processes. This framework includes the Mondrian process as a special case, and includes extensions of the Mondrian process allowing non-axis aligned cuts in high dimensional space. The processes are self-consistent, and we derive inference using sequential Monte Carlo and random forests. To our knowledge, this is the first work to provide self-consistent Bayesian nonparametric hierarchical partitioning with non-axis aligned cuts that is defined for more than two dimensions. As demonstrated by our simulation study and experiments on gene expression data, these non-axis aligned cuts can improve performance over the Mondrian process and other machine learning methods such as support vector machines, and random forests.

4.7 Appendix A

4.7.1 Theorems and proofs

Lemma 1. *With the notation established in* Section *2.1 of the main text, let $\Lambda \circ \varphi$ be a product measure $\lambda^d \times \lambda_+$ on $S^{d-1} \times \mathbb{R}_{\geq 0}$. If λ^d is symmetric and if λ_+ is the Lebesgue measure, then Λ is a translation invariant measure on the set H of hyperplanes in $\mathbb{R}^d$.*

Proof: We will first show that Λ is also symmetric (i.e., , invariant under reflection through the origin). Suppose that $A \subseteq H$ is a Λ-measurable set of hyperplanes.

$$\Lambda(A) = (\lambda^d \times \lambda_+)(\varphi^{-1}A) \tag{4}$$

$$= \int_0^\infty \lambda^d \left(\{\vec{n} \colon (\vec{n}, u) \in \varphi^{-1}A\}\right) du \tag{5}$$

$$= \int_0^\infty \lambda^d \left(\{-\vec{n} \colon (\vec{n}, u) \in \varphi^{-1}A\}\right) du = \Lambda(-A). \tag{6}$$

We use the symmetry of λ^d in (6), and in (5) we use the definition of product measures (Halmos, 1974).

Let $x \in \mathbb{R}^d$. If $h \in A$, and $\varphi^{-1}h = (\vec{n}, u)$, then $h = \{P : \langle \vec{n}, P - u\,\vec{n}\rangle = 0\}$ and $h + x = \{P : \langle \vec{n}, P - (u + \langle x, \vec{n}\rangle)\,\vec{n}\rangle = 0\}$. Let $A^+ + x = \{h + x \in A + x : \varphi^{-1}h = (\vec{n}, u)$ and $u + \langle x, \vec{n}\rangle \geq 0\}$ and let $A^- + x = \{h + x \in A + x : \varphi^{-1}h = (\vec{n}, u)$ and $u + \langle x, \vec{n}\rangle < 0\}$. By these definitions, $(A^+ + x) \cup (A^- + x) = A + x$.

$$\Lambda(A + x) = \Lambda(A^+ + x) + \Lambda(A^- + x) = \Lambda(A^+ + x) + \Lambda(-(A^- + x)) \tag{7}$$

$$= \int_{\vec{n} \in S^{d-1}} \lambda_+(\{u : (\vec{n}, u) \in \varphi^{-1}(A^+ + x)\}) + \lambda_+(\{u : (\vec{n}, u) \in \varphi^{-1} - (A^- + x)\}) d\lambda^d(\vec{n}) \tag{8}$$

$$= \int_{\vec{n} \in S^{d-1}} \lambda_+(\{u + \langle x, \vec{n}\rangle : (\vec{n}, u) \in \varphi^{-1}A^+\}) + \lambda_+(\{u + \langle x, \vec{n}\rangle : (\vec{n}, u) \in \varphi^{-1} - A^-)\}) d\lambda^d(\vec{n}) \tag{9}$$

$$= \int_{\vec{n} \in S^{d-1}} \lambda_+(\{u : (\vec{n}, u) \in A^+\}) + \lambda_+(\{u : (\vec{n}, u) \in -A^-\}) d\lambda^d(\vec{n}) \tag{10}$$

$$= \Lambda(A^+) + \Lambda(-A^-) = \Lambda(A^+) + \Lambda(A^-) = \Lambda(A). \tag{11}$$

Thus, Λ is translation invariant. In (7) and (11), we use the symmetry of Λ (from the first paragraph of this proof). In (10), we use the translation invariance of λ_+ and in (8), we use the definition of product measures (Halmos, 1974). In (9), we use the definitions of $A^+ + x$ and $A^- + x$. Note that $(\vec{n}, u) \in A^+ + x \Leftrightarrow (\vec{n}, u + \langle x, \vec{n}\rangle) \in A^+$ and $(\vec{n}, u) \in -(A^- + x) \Leftrightarrow (\vec{n}, u + \langle x, \vec{n}\rangle) \in -A^-$.

The intuition behind this proof is that translation by any x in A induces a measure preserving shear in $\Lambda \circ \varphi$, as in the stochastic geometry chapter of Kingman (1996) (this is shown for $d = 2$ in that chapter). In addition, the problem that arises from the fact that translation of a hyperplane through the origin in H flips the sign of $\vec{n}$ is handled by lifting the symmetry of λ^d in S^{d-1} to symmetry of Λ, which requires the decomposition of $A + x$ into a subsets with and without the sign flip (the subsets $A^+ + x$, and $A^- + x$ *resp.*).

Lemma 2. The likelihood function of the labels $\boldsymbol{Z}$ given the tessellation $\diamondsuit_t$ and the labels $\boldsymbol{V}$ and the hyperparameter $\boldsymbol{\alpha}$ can be computed as follows:

$$P(\boldsymbol{Z}|\diamondsuit_t, \boldsymbol{V}, \boldsymbol{\alpha}) = \int \cdots \int P(\boldsymbol{Z}, \{\boldsymbol{\phi}_j\}_{1\le j\le J_t}|\diamondsuit_t, \boldsymbol{\alpha})\, d\boldsymbol{\phi}_1 \cdots d\boldsymbol{\phi}_{J_t} \tag{12}$$

$$= \int \cdots \int \prod_{j=1}^{J_t} P(\boldsymbol{\phi}_j) \prod_{i=1}^{n} P(z_i|\diamondsuit_t, \boldsymbol{\phi}_{h(\boldsymbol{v}_i)})\, d\boldsymbol{\phi}_1 \cdots d\boldsymbol{\phi}_{J_t} \tag{13}$$

$$= \int \cdots \int \prod_{j=1}^{J_t} P(\boldsymbol{\phi}_j) \prod_{i:h(\boldsymbol{v}_i)=j,} \prod_{k=1}^{K} \phi_{jk}^{\delta(z_i=k)}\, d\boldsymbol{\phi}_1 \cdots d\boldsymbol{\phi}_{J_t}. \tag{14}$$

In (12), we use the conditional independence between $\boldsymbol{Z}$ and $\diamondsuit_t$. Let $m_{jk} = \sum_{i:h(\boldsymbol{v}_i)=j} \delta(z_i = k)$, and let $\boldsymbol{m}_j = (m_{jk})_{1\le k\le K}$.

$$P(\boldsymbol{Z}|\diamondsuit_t) = \int \cdots \int \prod_{j=1}^{J_t} P(\boldsymbol{\phi}_j) \prod_{k=1}^{K} \phi_{jk}^{m_{jk}}\, d\boldsymbol{\phi}_1 \cdots d\boldsymbol{\phi}_{J_t} \tag{15}$$

$$= \int \cdots \int \prod_{j=1}^{J_t} \frac{1}{B(\boldsymbol{\alpha})} \prod_{k=1}^{K} \phi_{jk}^{\alpha_k - 1} \prod_{k=1}^{K} \phi_{jk}^{m_{jk}}\, d\boldsymbol{\phi}_1 \cdots d\boldsymbol{\phi}_{J_t} \tag{16}$$

$$= \prod_{j=1}^{J_t} \frac{B(\boldsymbol{\alpha} + \boldsymbol{m}_j)}{B(\boldsymbol{\alpha})}. \tag{17}$$

Here $B(\cdot)$ is the multivariate beta function (Sivazlian, 1981).

Lemma 3. *In the notation from* Section *2.1 of the main text, if* Λ *is the uRTP measure such that* $\Lambda \circ \varphi = \lambda^d \times \lambda_+$ *and* λ^d *is measure associated with the uniform distribution on* λ^d*, and if* $B(x, r)$ *is the closed ball centred at* x *with radius* r*, then* $\Lambda([B(x, r)]) = r$.

Proof: By *Lemma 1*, the measure Λ is translation invariant, and so $\Lambda([B(x, r)]) = \Lambda([B(0, r)])$. By the definition of product measures (Halmos, 1974), $\varphi^{-1}[B(0, r)] = S^{d-1} \times [0, r]$, and so $\Lambda([B(0, r)])$ can be evaluated: $\Lambda([B(0, r)]) = \lambda^d(S^{d-1}) \cdot \lambda_+([0, r]) = r$.

4.8 Appendix B

4.8.1 Supplementary figures and tables

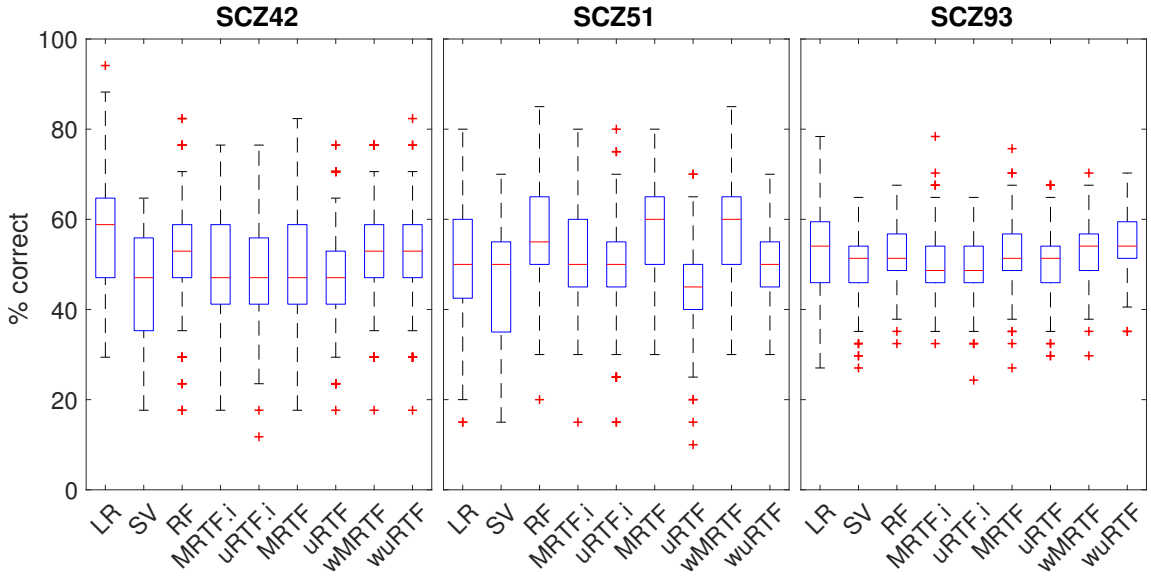


Figure 4.5: Boxplots for percent correct on all methods considered for the *SCZ42*, *SCZ51* and *SCZ93* datasets. Variance and disorganisation of these results may be due to noise, disorder and difficulty in the mapping from gene expression to schizophrenia. Despite this variance and disorganisation, many of the boxplots show significant improvements over the baseline. For example, a conservative sign test for the improvement of the wuRTF over the baseline in *SCZ93* is significant with a p-value of 5.1×10^{-10}.

Table 4.2: Pairwise sign tests among all methods considered on *GL85*, *SCZ42*, *SCZ51* and *SCZ93* datasets. Each table shows raw sign test p-values indicated by methods in row and column headers (not corrected for multiple testing). Sign tests are conducted in a conservative manner, in which the sign test is one-tailed towards the better method, and ties are assigned in favour of the method that is not better. Bolding in Table 1 of the main text is found by examining the column of this table corresponding to the best method for a dataset, and then bolding every method in that column with p-value > 0.05 (i.e., , nominal significance). BL indicates a baseline method in which the mode label in the training dataset is predicted for all data items.

GL85	BL	LR	SVM	RF	MRTF.i	uRTF.i	MRTF	uRTF	wMRTF	wuRTF
BL		6.7e-18	1.0e+00	7.0e-04	1.0e+00	1.0e+00	1.7e-24	1.0e+00	2.2e-28	5.3e-50
LR	6.7e-18		6.7e-18	1.7e-24	2.3e-20	2.6e-17	2.6e-43	9.9e-20	5.3e-50	8.3e-55
SVM	1.0e+00	6.7e-18		7.0e-04	1.0e+00	1.0e+00	1.7e-24	1.0e+00	2.2e-28	5.3e-50
RF	7.0e-04	1.7e-24	7.0e-04		1.1e-01	4.6e-05	2.0e-07	2.0e-02	1.4e-14	1.2e-37
MRTF.i	1.0e+00	2.3e-20	1.0e+00	1.1e-01		1.0e+00	2.3e-20	1.0e+00	1.4e-27	1.5e-44
uRTF.i	1.0e+00	2.6e-17	1.0e+00	4.6e-05	1.0e+00		5.4e-26	1.0e+00	4.6e-30	5.3e-50
MRTF	1.7e-24	2.6e-43	1.7e-24	2.0e-07	2.3e-20	5.4e-26		2.3e-22	7.0e-04	2.3e-20
uRTF	1.0e+00	9.9e-20	1.0e+00	2.0e-02	1.0e+00	1.0e+00	2.3e-22		1.4e-27	1.5e-48
wMRTF	2.2e-28	5.3e-50	2.2e-28	1.4e-14	1.4e-27	4.6e-30	7.0e-04	1.4e-27		3.2e-09
wuRTF	5.3e-50	8.3e-55	5.3e-50	1.2e-37	1.5e-44	5.3e-50	2.3e-20	1.5e-48	3.2e-09	

SCZ42	BL	LR	SVM	RF	MRTF.i	uRTF.i	MRTF	uRTF	wMRTF	wuRTF
BL		4.0e-08	1.0e+00	1.1e-01	9.9e-01	1.0e+00	8.2e-01	1.0e+00	1.8e-03	1.3e-05
LR	4.0e-08		4.0e-08	3.8e-02	4.6e-05	4.2e-04	4.4e-03	6.9e-06	5.3e-01	3.1e-01
SVM	1.0e+00	4.0e-08		1.1e-01	9.9e-01	1.0e+00	8.2e-01	1.0e+00	1.8e-03	1.3e-05
RF	1.1e-01	3.8e-02	1.1e-01		5.8e-01	2.6e-01	8.9e-01	3.1e-01	9.6e-01	5.3e-01
MRTF.i	9.9e-01	4.6e-05	9.9e-01	5.8e-01		1.0e+00	1.0e+00	9.6e-01	3.8e-02	5.2e-02
uRTF.i	1.0e+00	4.2e-04	1.0e+00	2.6e-01	1.0e+00		6.4e-01	1.0e+00	6.9e-02	7.0e-04
MRTF	8.2e-01	4.4e-03	8.2e-01	8.9e-01	1.0e+00	6.4e-01		7.4e-01	7.4e-01	1.8e-01
uRTF	1.0e+00	6.9e-06	1.0e+00	3.1e-01	9.6e-01	1.0e+00	7.4e-01		6.9e-02	8.2e-05
wMRTF	1.8e-03	5.3e-01	1.8e-03	9.6e-01	3.8e-02	6.9e-02	7.4e-01	6.9e-02		9.5e-01
wuRTF	1.3e-05	3.1e-01	1.3e-05	5.3e-01	5.2e-02	7.0e-04	1.8e-01	8.2e-05	9.5e-01	

SCZ51	BL	LR	SVM	RF	MRTF.i	uRTF.i	MRTF	uRTF	wMRTF	wuRTF
BL		1.1e-01	1.0e+00	7.6e-11	2.8e-03	1.0e+00	1.4e-13	1.0e+00	1.2e-15	4.7e-01
LR	1.1e-01		1.1e-01	2.8e-02	9.3e-01	3.1e-01	1.1e-03	1.4e-04	8.2e-05	2.6e-01
SVM	1.0e+00	1.1e-01		7.6e-11	2.8e-03	1.0e+00	1.4e-13	1.0e+00	4.1e-15	4.7e-01
RF	7.6e-11	2.8e-02	7.6e-11		8.9e-02	4.0e-08	9.8e-01	1.4e-14	9.3e-01	6.9e-06
MRTF.i	2.8e-03	9.3e-01	2.8e-03	8.9e-02		1.8e-01	2.8e-03	3.5e-06	8.2e-05	4.2e-01
uRTF.i	1.0e+00	3.1e-01	1.0e+00	4.0e-08	1.8e-01		2.8e-11	8.2e-01	3.2e-09	9.6e-01
MRTF	1.4e-13	1.1e-03	1.4e-13	9.8e-01	2.8e-03	2.8e-11		4.2e-19	1.0e+00	2.0e-07
uRTF	1.0e+00	1.4e-04	1.0e+00	1.4e-14	3.5e-06	8.2e-01	4.2e-19		3.4e-16	2.8e-02
wMRTF	1.2e-15	8.2e-05	4.1e-15	9.3e-01	8.2e-05	3.2e-09	1.0e+00	3.4e-16		4.4e-14
wuRTF	4.7e-01	2.6e-01	4.7e-01	6.9e-06	4.2e-01	9.6e-01	2.0e-07	2.8e-02	4.4e-14	

SCZ93	BL	LR	SVM	RF	MRTF.i	uRTF.i	MRTF	uRTF	wMRTF	wuRTF
BL		8.9e-02	1.0e+00	3.8e-02	9.6e-01	9.5e-01	6.9e-02	9.7e-01	6.6e-03	5.1e-10
LR	8.9e-02		3.1e-01	7.8e-01	1.8e-01	9.7e-03	3.6e-01	8.9e-02	8.2e-01	3.6e-01
SVM	1.0e+00	3.1e-01		3.6e-01	9.9e-01	1.0e+00	5.8e-01	1.0e+00	1.4e-01	4.0e-08
RF	3.8e-02	7.8e-01	3.6e-01		3.1e-01	3.1e-01	9.1e-01	1.4e-01	9.3e-01	6.9e-02
MRTF.i	9.6e-01	1.8e-01	9.9e-01	3.1e-01		9.1e-01	3.6e-01	7.8e-01	1.8e-03	2.0e-07
uRTF.i	9.5e-01	9.7e-03	1.0e+00	3.1e-01	9.1e-01		2.6e-01	8.6e-01	3.8e-02	8.2e-05
MRTF	6.9e-02	3.6e-01	5.8e-01	9.1e-01	3.6e-01	2.6e-01		2.2e-01	6.9e-01	2.8e-03
uRTF	9.7e-01	8.9e-02	1.0e+00	1.4e-01	7.8e-01	8.6e-01	2.2e-01		1.4e-02	9.0e-08
wMRTF	6.6e-03	8.2e-01	1.4e-01	9.3e-01	1.8e-03	3.8e-02	6.9e-01	1.4e-02		2.6e-01
wuRTF	5.1e-10	3.6e-01	4.0e-08	6.9e-02	2.0e-07	8.2e-05	2.8e-03	9.0e-08	2.6e-01	

Table 4.3: Standard deviation of percent correct on gene expression datasets over 200 random train/test splits across different methods.

Dataset	BL	LR	SVM	RF	MRTF.i	uRTF.i	MRTF	uRTF	wMRTF	wuRTF
GL85	5.58	10.52	5.58	7.18	6.09	5.16	6.43	5.78	6.16	6.05
SCZ42	10.62	13.89	10.60	12.17	10.93	11.10	11.76	11.40	11.70	11.41
SCZ51	10.65	12.77	10.67	11.74	11.36	11.12	9.84	10.21	11.07	9.69
SCZ93	7.74	9.65	7.24	7.42	7.20	6.94	7.65	7.24	6.86	6.90

Table 4.4: Comparison of mean running time (in minutes) on gene expression datasets over 200 random train/test splits across different methods. The experiments were run on an Intel Xeon CPU E5-2683v4@2.10GHz.

Dataset	LR	SVM	RF	MRTF.i	uRTF.i	MRTF	uRTF	wMRTF	wuRTF
GL85	<.08	<.08	0.006	0.003	43.221	21.681	22.614	15.595	14.679
SCZ42	<.08	<.08	0.005	0.001	2.118	5.468	4.675	5.180	5.066
SCZ51	<.08	<.08	0.004	0.002	5.305	8.420	7.764	8.161	8.641
SCZ93	0.006	0.012	0.006	0.005	89.106	43.534	35.802	41.759	41.068

4.9 Appendix C

4.9.1 Algorithm for Monte Carlo based approximate inference

The spherical approximation introduced in *Section* 2.2.2 of the main text requires the computation of the radius of smallest closed ball of each polytope in the tessellation. This may be a costly $O(\sum_{a \in \triangle} |a \cap \boldsymbol{V}|^2)$ operation. Further, this approximation may introduce bias into the sampling procedure. In this appendix, we provide a sampling strategy based on Monte Carlo approximation within SMC that may be more precise and efficient. This strategy is described in Supplementary Algorithm 1.

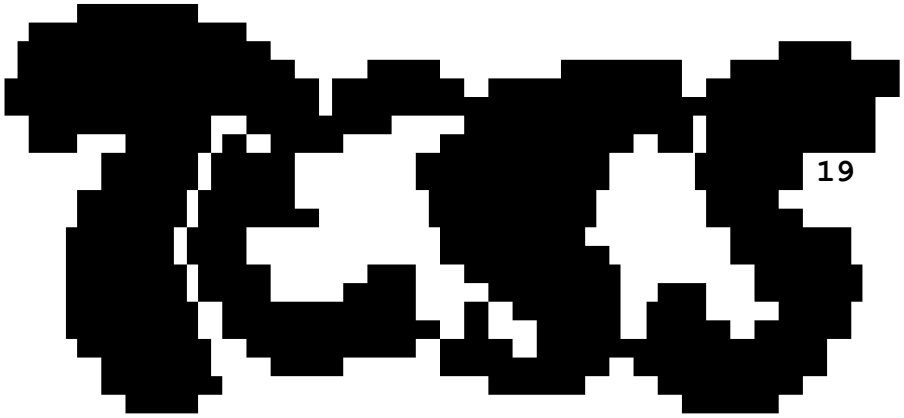
We derive this algorithm for the case discussed in the main text in which the budget for the MJP is $\tau = \infty$ by making use of the 'pausing condition' described in *Section* 2.2.2 of the main text. We say that a polytope a is 'paused' if the set of labels $\{\boldsymbol{Z}_i : \boldsymbol{V_i} \in a\}$ contains only one label, and we say that a tessellation $\triangle$ has met the pausing condition if a is paused for all $a \in \triangle$.

Algorithm 7 Monte Carlo approximations for RTPs

Inputs: a) Training dataset $\boldsymbol{V}$, $\boldsymbol{Z}$, b) RTP measure Λ on H, c) likelihood hyperparameter α, d) A radius r such that $B(r)$ is the smallest closed ball containing $\boldsymbol{V}$, e) The number of Monte Carlo iterations N (our suggested default is $N = 1,000$).
Outputs: Approximate RTP posterior $\sum_{m=1}^{M} \varpi_m \delta_{\diamondsuit_m}$ at time ∞. (ϖ are particle weights.)
Set $\diamondsuit_m \leftarrow \{\text{hull } \boldsymbol{V}\}$, $\varpi_m \leftarrow 1/M$, for $m = 1, \ldots, M$.
while the 'pausing condition' is not met for all $\diamondsuit_m$ **do**
    Resample $\diamondsuit'_m$ from $\{\diamondsuit_m\}_{m=1}^{M}$ *w.p.p.t.* $\{\varpi_m\}_{m=1}^{M}$, for $m = 1, \ldots, M$.
    Set $\diamondsuit_m \leftarrow \diamondsuit'_m$, for $m = 1, \ldots, M$.
    Set $\varpi_m \leftarrow 1/M$, for $m = 1, \ldots, M$.
    for m : the 'pausing condition' is not met for $\diamondsuit_m$ **do**
        Sample N hyperplanes $H = \{h_1, \ldots, h_N\}$ from $\Lambda(\cdot \cap [B(r)])/\Lambda([B(r)])$.
        Set $H_a = \{h \in H : h \cap a \neq \varnothing\}$, for each polygon $a \in \diamondsuit_m$ such that a is not 'paused'.
        if $\sum_{a \in \diamondsuit_m} |H_a| > 0$ **then**
            Sample a polytope a from $\diamondsuit_m$ with probability $|H_a| / \sum_{a' \in \diamondsuit_m} |H_{a'}|$.
            Sample a hyperplane h uniformly from H_a.
            Set $\diamondsuit_m \leftarrow (\diamondsuit_m / \{a\}) \cup \{\text{hull}(\boldsymbol{V} \cap a \cap h^-), \text{hull}(\boldsymbol{V} \cap a \cap h^+)\}$.
        Set $\varpi_m \leftarrow \varpi_m P(\boldsymbol{Z} | \diamondsuit_m, \boldsymbol{V}, \alpha) / P(\boldsymbol{Z} | \diamondsuit'_m, \boldsymbol{V}, \alpha)$.
    Set $\mathcal{Z} \leftarrow \sum_{m=1}^{M} \varpi_m$.
    Set $\varpi_m \leftarrow \varpi_m / \mathcal{Z}$, for $m = 1, \ldots, M$.
return the particle approximation $\sum_{m=1}^{M} \varpi_m \delta_{\diamondsuit_m}$.

4.10 Appendix D

4.10.1 Manual for `tess19 v1.0`



Copyright (c) 2019. Shufei Ge and Lloyd T. Elliott

The `tess19` software implements the Bayesian nonparametric methods described in Ge et al., *Random Tessellation Forests*, 2019. This software constructs a random forest for posterior prediction of categorical data based on real valued predictors. The trees of the random forest are found through

SMC inference. This manual is for version v1.0. This software requires the following R packages: `optparse`, `purrr`. This software is released under the open source BSD 2-clause license.

1. Basic Usage

```
tess19 <IFILE.txt> <OFILE.txt>
```

Predictions for the missing labels in the `Levels` column of the file <IFILE.txt> are made using the predictors in the file <IFILE.txt> and the uRTF model, with 100 trees, with the prespecified budget $\tau = \infty$, and with the hyperparameter settings $\alpha_k = n_k/1000$. The predictions are saved in the `Levels` column of the file <OFILE.txt>.

The format of the file <IFILE.txt> is as follows. The file is space separated. The first line is a header line with one column name for each of predictor (for example, `V1 V2 ...`), followed by a column named `Levels`. Subsequent lines are given with one line per data item, with the predictors for the data item followed by the items level. The predictors are real numbers and the levels must be positive integers in the set $1, \ldots K$, where K is the number of levels. Both test and train data must be provided in <IFILE.txt>, and the test data items must have missing labels indicated by the string `NaN`. The predictors may not have missing data.

The format of the file <OFILE.txt> is as follows. The first line is a header line naming the column `Levels` (i.e., , one column). Subsequent lines are given with one line for data item. If the data item is a training data item, then the value `NaN` is recorded in the corresponding line. If the data item is a testing data item, then a predicted label is recorded.

2. Advanced Usage

```
tess19 --usage

tess19 --license

tess19 --version

tess19 [--Mondrian] [--weights <WFILE.txt>] [--cuts <MAX-CUTS>]
       [--tau <PRESPECIFIED-BUDGET>] [--alpha <HYPER-PARAMETER>]
       [--trees <NUMBER-OF-TREES>] [--particles <PARTICLES>]
       [--seed <SEED>] <IFILE.txt> <OFILE.txt>
```

`--usage`. Prints this manual to the standard output.

`--license`. Prints the open source BSD 2-clause license for this software.

`--version`. Prints the software version information.

`--Mondrian`. Instructs `tess19` to conduct axis aligned cuts, yielding the MRTF model, or (if the `--weights` flag is provided) the wMRTF model. By default, axis aligned cuts are not used.

`--weights <WFILE.txt>`. Instructs `tess19` to use a weighted version of the uniform distribution for the measure λ^d, yielding the wuRTF or wMRTF model. The weights ω_j are read from the file `<WFILE.txt>` which must contain d lines corresponding to the prior weight for each predictor, with one real number per line.

`--cuts <MAX-CUTS>`. This flag sets a stopping condition wherein SMC particles will return after `<MAX-CUTS>` cuts, regardless of the budget. By default, `<MAX-CUTS>` is set to 100. This value must be a positive integer. By the pausing condition, each cut separates a data item, and setting `<MAX-CUTS>` to a value larger than or equal to the number of data items is equivalent to setting `<MAX-CUTS>` to ∞.

`--tau <PRESPECIFIED-BUDGET>`. This flag specifies the budget. The budget τ must be a positive real number, or ∞ (the string `Inf`). By default, τ is set to infinity.

`--alpha <HYPER-PARAMETER>`. This flag is a positive real number that sets the coefficient of the empirical label proportion in the Dirichlet/multinomial prior, so that the value of the hyperparameter is α_k = `<HYPER-PARAMETER>` * n_k. The default value is 10^{-3}.

`--ntrees <NUMBER-OF-TREES>`. This flag is a positive integer that sets the number of trees to use in the random forest. By default, `<NUMBER-OF-TREES>` is set to 100. A value of 1 specifies the uRTP/wuRTP/MRTP/wMRTP priors (i.e., no random forest).

`--particles <PARTICLES>`. This flag sets the number of particles to use in the SMC approximations. The default value of `<PARTICLES>` is 100.

`--seed <SEED>`. This flag sets the random seed to `<SEED>`. The default is to use the system clock to set the random seed.

3. License

Chapter 5

Random Tessellation with Splines

5.1 Introduction

Shapes (*i.e.* boundaries of objects) within images are widely studied in biological science. This application domain includes for example classifying cells collected in a blood sample from individuals and characterizing brain diseases of patients by analyzing shapes within brain images. One basic issue that arises in estimating shapes is the choice of the representation of the shape. Curves (Su and Liu, 2014; Muller, 1997) are commonly used to provide flexible representations of a wide range of objects (e.g., : tumors, cells). In related work, shape models have been explored using smooth stochastic processes Kurtek et al. (2012). Bayesian hierarchical models and Gaussian processes have also been used Gu et al. (2012); Lin et al. (2019). Bayesian nonparametrics has identified shape modelling as an important area of application, supplementing traditional classification, regression and clustering applications Hannah and Dunson (2013); Bhattacharya and Dunson (2010). We consider modeling shapes within image with curves, in the framework of space partitioning methods.

The Mondrian process (Roy and Teh, 2008) partitions multidimensional space into hyper-rectangles with recursive axis-aligned hyperplane cutting. In the Mondrian process, hyper-rectangles are assigned to lifetimes (*i.e.*, costs) sampled from an exponential distribution, with rate given by the perimeter of the hypercube. In each iteration, the hypercube with the smallest lifetime is removed and replaced by two new partitioning hyper-rectangles with randomly sampled hyperplans. New lifetimes for all existing hyper-rectangles are then sampled. This process repeats until the total cost exceeds a prespecified budget. The Mondrian process is used as a nonparametric prior distribution in Bayesian models of relational data, and provides a Bayesian view of decision trees (Roy and Teh, 2008; Kemp et al., 2006). Developments in the Mondrian process for space partitioning achieve high predictive accuracy, with improved efficiency. These methods include online methods (Lakshminarayanan et al., 2014), and particle Gibbs for Mondrian process additive regression trees (Lakshminarayanan et al., 2015). However, the axis-aligned hyperplanes of the decision boundaries of the Mondrian process restrict its flexibility, which could lead to failure in capturing inter-dimensional dependencies among features. Bayesian nonparametric methods have been introduced to generalise the Mondrian process and allow non-axis aligned cuts, such as generating arbitrary oblique cutting lines or hyperplanes

(Fan et al., 2018, 2019, 2016; Ge et al., 2019) to partition the target space. Alternative constructions of non-axis aligned partitioning for multidimensional spaces and non-Bayesian methods include sparse linear combinations of predictors or canonical correlation as random forest generalisations (George, 1987; Tomita et al., 2015; Rainforth and Wood, 2015).

We propose a *random tessellation with splines* (RTS), an extension of the Mondrian process and *random tessellation process* (RTP) which further modifies the *random tessellation process* by replacing the non-axis aligned cutting hyperplanes with curved cuts. The RTS provides a framework for describing Bayesian nonparametric models based on partitioning two-dimensional Euclidean space with splines. The proposed nonlinear curves enable more complex data-structures to be represented, and allow arbitrary shapes to be approximated and outputted by the model. As with the *random tessellation process*, the construction of the RTS is based on the theory of stable iterated tessellations in stochastic geometry (Nagel and Weiss, 2005). The partitions induced by the RTS prior are described by a sets of compact subsets. We note that in contrast to the RTP, each compact subset could contain more than one disjoint connected components (i.e., , the blocks of the partition need not be connected).

We implement a sequential Monte Carlo (SMC) algorithm (Doucet et al., 2000) for RTS inference, which takes advantage of the hierarchical structure of the generative process of RTS. The SMC algorithm can be easily parallelized by allocating samples across multiple CPUs (cores) to decrease the computational time. Our simulation study provides some evidence that our *random tessellation with splines* is consistent, however exploring the consistency of RTS is an area of future work. We apply our proposed model to simulated yin-yang data and an HIV-1-infected human macrophage dataset to demonstrate its effectiveness. For the experiment on the macrophage dataset, we consider a task wherein the perimeter of a macrophage cell is estimated. Shapes of biological microstructures may indicate disease etyology. For example, the fractal dimension of vascularization in a cancer tumor is known to be modulated during remission and metastasis Gazit et al. (1997). This motivates the perimeter estimation, as estimation of perimeters through a continuum of resolution indicates fractal dimension.

5.2 Methods

We characterize shapes by describing their boundary through a space partitioning approach. Let $(\boldsymbol{v}_1, z_1), \ldots, (\boldsymbol{v}_n, z_n)$ denote n observed data items, where $\boldsymbol{v}_i \in \mathbb{R}^2$ are predictors and $z_i \in \{1, \ldots, K\}$ are labels. In our application, we are interested in modeling the shapes within a two-dimensional image. The predictors $\boldsymbol{v}_i$ are pixels of images, and the labels z_i are labels of shapes. For example, in our real data application, we are interested in estimating the shapes of HIV-1-infected human macrophage. Pixels falling within the cell are labelled 1 and other pixels are labelled 2.

A Bayesian nonparametric model partitions the feature space by placing a prior distribution on the partition process, and associates model parameters with the blocks of the partition. Inference is then made on the joint posterior of the parameters and the structure of the partition. In this section,

Algorithm 8 A generative Process for the RTS, as in Ge et al. (2019) but with Λ as a measure on splines

Inputs: a) Bounded domain W, b) RTS measure Λ on H, c) prespecified budget τ.
Outputs: A realisation of the Random Tessellation with Splines $(Y_t)_{0\leq t\leq\tau}$.
$\tau_0 \leftarrow 0$.
$Y_0 \leftarrow \{W\}$.
while $\tau_0 \leq \tau$ **do**
    Sample $\tau' \sim \text{Exp}\Big(\sum_{a\in Y_{\tau_0}} \Lambda([S])\Big)$.
    Set $Y_t \leftarrow Y_{\tau_0}$ for all $t \in (\tau_0, \min\{\tau, \tau_0 + \tau'\}]$.
    Set $\tau_0 \leftarrow \tau_0 + \tau'$.
    if $\tau_0 \leq \tau$ **then**
        Sample a subset S from the set Y_{τ_0} with probability proportional to $\Lambda([S])$.
        Sample a Bézier curve b from $[S]$ according to the probability measure $\Lambda(\cdot \cap [S])/\Lambda([S])$.
        $Y_{\tau_0} \leftarrow (Y_{\tau_0}/\{S\}) \cup \{S \cap b^-, S \cap b^+\}$. ($b^-$ and b^+ are the $b-$*bounded* closed half curved planes.)
    else
        return the tessellation-valued right-continuous MJP sample $(Y_t)_{0\leq t\leq\tau}$.

we develop the RTS and place a prior on partitions of $(\boldsymbol{v}_1, z_1), \ldots, (\boldsymbol{v}_n, z_n)$ induced by tessellations via splines in a two-dimensional feature space.

5.2.1 Random Tessellation with Splines

A tessellation Y in the RTS of a bounded domain $W \subset \mathbb{R}^2$ is a finite collection of compact subsets, such that the union of the subsets is all of W, and the subsets have pairwise disjoint interiors (Chiu et al., 2013). Note that each subset may contain more than one disjoint connected components, as shown in Fig 5.1. Also: throughout this work we consider the two-dimensional plane (although, these methods may be generalized to higher dimensions as a point of future work). Denote $\mathbb{C}_0$ as the set of all compact sets with finitely many connected components, $\mathbb{C}_0 = \cup_{j=1}^{J_c} C_j$, $J_c \geq 2$, $\text{interior}(C_i) \cap \text{interior}(C_j) = \varnothing$ for any $i \neq j$. Let $S_i, i = 1, \cdots, J_s$ denotes each subset in the tessellation, and each S_i could contain more than one disjoint connected components. We denote tessellations of W by $Y(W)$, $Y(W) = \cup_{i=1}^{J_s} S_i, S_i = \cup_{C_j\in S_i} C_j$. For example, in Fig 5.1, at time τ_4, $Y(W) = \cup_{i=1}^{5} S_i, S_1 = C_1, S_2 = C_2 \cup C_5, S_3 = C_4, S_4 = C_3 \cup C_6, S_5 = C_7$.

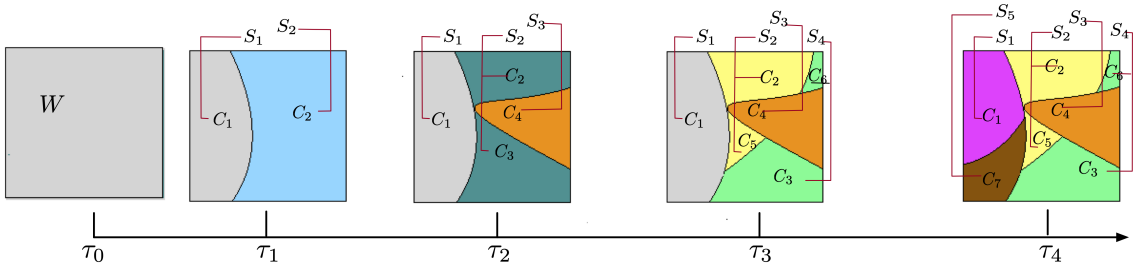


Figure 5.1: An example of a generative process from the RTS prior with domain given by a two dimensional box $(x, y) \in [-1, 1]^2$. Colours indicate subsets identity, and are randomly assigned.

The random tessellation with splines forms a partition. An element of the partition is a subset with non-empty interior formed by the intersection of finitely many closed curves. The RTS itself $Y_t(W)$ is a partition-valued Markov process defined on $[0, \tau]$, with cutting events specified by splines that intersect the tessellation's subsets. Here τ is a prespecified budget (Berger, 2012).

The RTS $Y_0(W)$ is initialized with a single subset W covering all of the observed predictors: W is the convex hull hull$\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n\}$. The Markov process for the RTS associates each subset with an exponentially distributed lifetime. The subset is replaced by two new subsets at the end of subset's lifetime t. At time t, we sample a spline that intersects the interior of the old subset to form two new subsets. We evolve the Markov process until we reach the prespecified budget τ.

As in Ge et al. (2019), we associate a measure on splines with the generative process described in *Section* 2.3. If W' is a subset of W, let $[W']$ be the set of all splines formed by the generative process described in *Section* 2.3 that intersect W'. The generative process thus implicitly defines a measure Λ on splines such that the density of the spline found through the generative process is $\Lambda(\cdot)/\Lambda([W'])$. We can sample from this density using rejection sampling.

To sample an RTS, we sample a Bézier curve (Mortenson, 1999) to cut a subset. Bézier curves are widely used in computer graphic to draw shapes, due to their flexibility in modeling complex shapes and their simple mathematical formulation. Let B be the set of Bézier curves in R^2. Every Bézier curve b is uniquely defined by a set of control points $P_0, \ldots, P_n$, where n is the order of the Bézier curve. In our numerical experiments, we let n takes values from $1, 2, 3$ randomly ($n = 1$ for linear, 2 for quadratic, 3 for cubic).

5.2.2 Coordinate rotation

We discuss the problem of determining which side of a Bézier curve a point lies on. This is required in order to partition points in the target, given the Bézier curve cuts. After proposing a Bézier curve in a reference coordinate system, we may rotate it and translate it along a normal in order to produce a proposal for the cut. However, after rotation the curve may no longer be a function of the x-axis (i.e., , the mapping from x to the rotated and translated spline is not injective), which makes it difficult to identify which side of the curve target points lie on (we cannot just compare their y-levels). This is shown in Fig.5.2 (a). In order to use the Bézier curve to split targets after rotation by an arbitrary angle, we rotate the

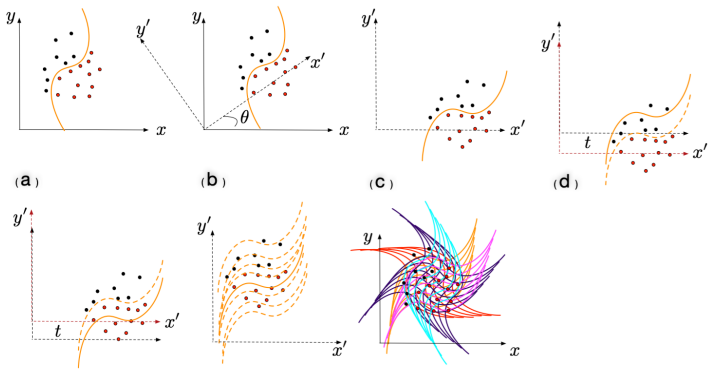


Figure 5.2: An example of coordinate rotation. (a) A rotated Bézier curve (with angle θ), which is not injective in the original coordinate system. (b-c) A new coordinate system and a Bézier curve proposed so as to be injective. The new coordinate is obtained by rotating the original coordinate by angle θ. (d-e)Translating a Bézier curve in the new coordinate system. (f) A subset of a Bézier curve splits the targets from different positioning via moving up and down. (g) A subset of a Bézier curve splits the target from different angles and positions.

coordinate system rather than the curve: We project the targets to a new coordinate system and the Bézier curve proposal can split the targets by comparing the y-levels in the new coordinate system. This rotation is shown in Fig.5.2 (b) - (c). The translation of the Bézier curve is displayed in Fig.5.2 (d) - (e). Fig.5.2 (f) and (g) show subsets of Bézier curves splitting targets according to an arbitrary angle.

5.2.3 Generative process for the cutting spline

Now we turn to the description of a generative process for the cutting spline. Without loss of generality, we can assume the shape being cut is a square. Our cutting procedure should define a cut given by a single random spline, and incorporate the coordinate rotation scheme. The generative process is as follows:

1. Project the shape to a new coordinate system by rotating the coordinate with angle θ as in Fig.5.2 (a) - (c), and moving it along a normal vector by t as in Fig.5.2 (e) - (f). Denote the coordinates before projection by (x, y), and after projection by (x', y'). This indicates the following transformations:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ t \end{bmatrix}, \qquad \theta \sim \text{Unif}(0, 2\pi), \qquad t \sim \text{Unif}(-L_t, L_t). \tag{1}$$

2. Get the range of the segment in the new coordinate system on both x and y-axes, denote them by $[x'_{\min}, x'_{\max}]$, $[y'_{\min}, y'_{\max}]$. Expand the ranges $L_{x'}, L_{y'}$ times separately. Denote the new ranges as $[x^*_{\min}, x^*_{\max}]$, $[y^*_{\min}, y^*_{\max}]$, where:

$$\begin{bmatrix} x^*_{\min} \\ x^*_{\max} \end{bmatrix} = \begin{bmatrix} x'_{\min} - L_{x'} \times (x'_{\max} - x'_{\min}) \\ x'_{\max} + L_{x'} \times (x'_{\max} - x'_{\min}) \end{bmatrix}, \begin{bmatrix} y^*_{\min} \\ y^*_{\max} \end{bmatrix} = \begin{bmatrix} y'_{\min} - L_{y'} \times (y'_{\max} - y'_{\min}) \\ y'_{\max} + L_{y'} \times (y'_{\max} - y'_{\min}) \end{bmatrix}. \tag{2}$$

3. Sample the control points for the spline in the new coordinate system via following sampling scheme:

   (a) Sample the number of control points, n_c, from values $\{2, 3, 4\}$ with equal probability.

   (b) Sample the control points themselves:

$$\text{Sample } \begin{bmatrix} x^c_1 \\ y^c_1 \end{bmatrix}, \cdots, \begin{bmatrix} x^c_{n_c} \\ y^c_{n_c} \end{bmatrix}, \text{ such that:} \tag{3}$$

$$x^c_1 = x^*_{\min}, x^c_{n_c} = x^*_{\max}, x^c_j \sim \text{Uniform}(x^*_{\min}, x^*_{\max}) \forall\, j \notin \{1, n_c\}. \tag{4}$$

$$\text{Sample } y_j^c \sim \text{Uniform}(y_{\min}^*, y_{\max}^*). \text{ Set } y_{n_c}^c = y_1^c. \tag{5}$$

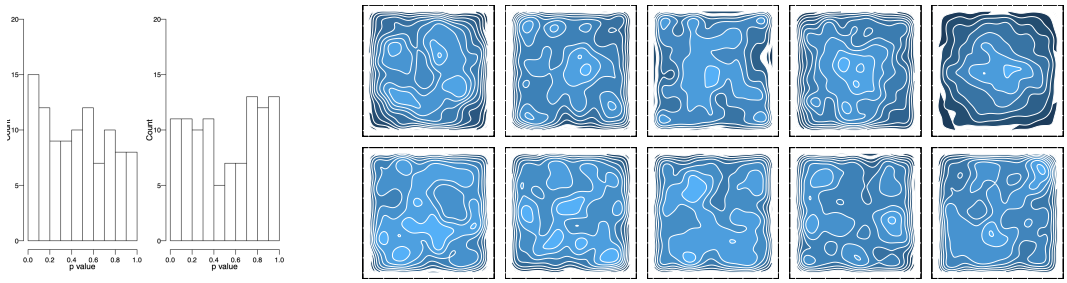
(c) The spline is then determined by the control points, according to Mortenson (1999).

4. Divide the segment into two new segments, according to the spline.

5. Transform the new segments and the the sampled spline back to the original coordinate system.

Note that in Equation 4 we assume that $x_1^c, ..., x_{n_c}^c$ are arranged in ascending order. If we choose L_t, $L_{x'}$,$L_{y'}$ appropriately, the proposed splines would be approximately uniformly distributed over the target as shown in *Section* 5.2.4.

5.2.4 Evidence of projectivity for the RTS



(a) (b)

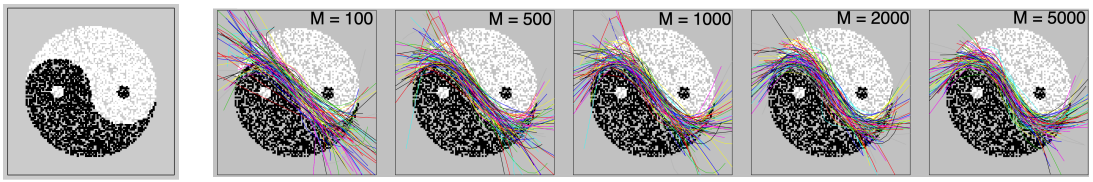
Figure 5.3: (a) Frequency of p-values of bivariate uniformity tests on 100 experiments. The *left* panel displays the p-values of data sampled from the first cuts, the *right* panel presents the p-values of data sampled from a bivariate uniform distribution. (b) The *upper* panels are bivariate density plots of 5 randomly selected experiments, in which data are sampled from the proposed first cuts on a unit square; the *lower* panels show density of 5 random draws of 5000 points sampled from a uniform distribution on a unit square.

Projectivity (or self-consistency) is required in order for inference on Bayesian nonparametric objects to be well defined. For all bounded domains $W' \subseteq W$, if $Y_t(W')$ is equal in distribution to $Y_t(W) \cap W'$, then projectivity holds. We conduct a simulation study to explore the projectivity of the RTS. The experiment is repeated 100 times with different random seeds. In each replicate, we generate 5000 random Béizer curves that intersect with the unit square $[0, 1]^2$, with $L_t = 9\sqrt{2}, L_{x'} = 10, L_{y'} = 2$. We randomly sample one point from each Béizer curve, lying inside the square. The total number of sampled points is 5000 in each replicate. Under the assumption of projectivity, the sampled points are distributed according to a bivariate uniform random variable. We then conduct a bivariate uniformity test (Yang and Modarres, 2017) on the sampled points for each experiment. The null hypothesis H_0 is that the data are drawn from a bivariate uniform distribution on a square. The frequency of the p-values is displayed in Fig. 5.3 (*a*) *left* panel, in which 93 out of the 100 p-values are greater than 0.05. We also provide the p-values of the bivariate uniformity test on 100 randomly

sampled same sized datasets from a uniformly distribution on $[0,1]^2$ as a benchmark in Fig. 5.3 (*a*) *right* panel, and 95% of the p-values are greater than 0.05. In Fig.5.3 (*b*) *upper* panels, we provide the bivariate density plots of 5 randomly drawn data sets from the 100 replicates. The density plots show that the distribution of data points are not concentrated on a certain small area, but on many small areas or the distribution is relatively flat over the square, which has a similar pattern as bivariate uniform distributions (e.g. Fig.5.3 (*b*) *lower* panels). This indicates the sampled points may follow a bivariate uniform distribution, and provides some evidence towards projectivity.

5.2.5 Sequential Monte Carlo for RTS inference

We use techniques from relational modeling for categorical data in order to specify a likelihood model for the RTS. This specifies the same likelihood that was used in the random tessellation process Ge et al. (2019). Let $\boldsymbol{Y}_t(\boldsymbol{W})$ be an RTS on the domain $W = \text{hull}\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n\}$. Our objective is to infer the normalized posterior distribution of $\boldsymbol{Y}_t(\boldsymbol{W})$, denoted $\pi(\boldsymbol{Y}_t(\boldsymbol{W}))$. We develop an SMC algorithm to infer $\pi(\boldsymbol{Y}_t(\boldsymbol{W}))$, by taking advantage of the hierarchical structure of the RTS prior. Our algorithm iterates between three steps until a total budget τ is reached: resampling particles, propagation of particles and weighting of particles. We obtain a list of weighted particles to approximate the RTS posterior $\sum_{m=1}^{M} \varpi_m \delta_{\boldsymbol{Y}_{\tau,m}(\boldsymbol{W})}$ at time τ. Here m is the particle index, M is the total number of particles and ϖ_m are the particle weights. Two methods are used to decrease the cost and complexity for computation. Firstly, we approximate $\Lambda([S])$ with the measure $\Lambda([\cdot])$ applied to the smallest closed circle containing S in the computation of the lifetimes of subsets. Secondly, we use a pausing condition so that no cutting splines are proposed for subsets in which the labels of all predictors are same. We refer readers to Ge et al. (2019) for more details of these two methods and the frame work of the SMC algorithm. In addition, we parallel the propagation step and the weighting step in the SMC algorithm by allocating samples across multiple CPUs (cores) to decrease the computational time.



(a) (b)

Figure 5.4: (*a*) A view of the yin-yang dataset, with black indicting label 1 and white indicting label 2 . (*b*) Best first cuts among different number of particles. Each experiment is repeated 100 times.

5.3 Experiments

In *Section* 5.2.5, we conduct a simulation study to evaluate the performance of our SMC approach with respect to different number of cores and particles. In *Section* 5.3.2 we evaluate our method via an HIV-1-infected human macrophage image data sets, and compare our method with modern machine learning approaches. We also explore the value of outputing shapes by computing the

perimeter of cells. Throughout our experiments, we set the likelihood hyperparameters for the RTS to $\alpha_k = n_k/1000$, $n_k = \sum_i \delta(z_i = k)$, and use $\delta(\cdot)$ as is an indicator function with $\delta(z_i = k) = 1$ if $z_i = k$ and $\delta(z_i = k) = 0$ otherwise.

5.3.1 Simulations on the yin-yang

We simulate an yin-yang dataset. First, we uniformly sample 10000 points in the square $[-1, 1]^2$, points falling outside of the circle $x^2+y^2 = 1$ are removed. Points satisfying $(x-0.5)^2 + y^2 < 0.1^2$ or $x > 0, y < 0, (x-0.5)^2 + y^2 > 0.25$ or $x < 0, y < 0, (x+0.5)^2+(y+0.5)^2 > 0.1^2$ or $x < 0, y > 0, (x+0.5)^2 + y^2 < 0.25$ are labeled 1 and the rest are labeled 2. Fig. 5.4a provides a visualization of the yin-yang dataset, wherein points are colored black or white based on their label. We first conduct an experiment to evaluate the performance of SMC approach with respect to different number of particles. Each experiment is repeated 100 times, and we allocate 60% of the data items at random to a training set, the rest to the testing set. Fig. 5.4b displays the best first cuts with different number of particles. The first cut can generally capture the division of the yin-yang image when we use $M = 5000$ particles in SMC, which indicates that a larger number of particles can improve the performance of the SMC algorithm. We note that due to the recursive nature of tree-based algorithms such as the RTS, successful first cuts suggest good general performance of the algorithm.

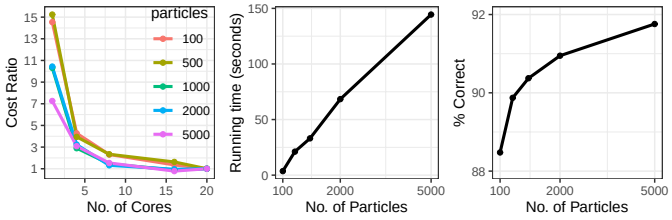


Figure 5.5: *(Left)* Cost-ratio of SMC versus number of cores with different number of particles settings. The baseline is the running time with 20 cores. *(Middle)* Running time (seconds) of SMC versus different numbers of particles with 16 cores. *(Right)* % correct of parallel SMC with different numbers of particles. All experiments were run on Intel E5-2683 v4 Broadwell 2.1Ghz machines.

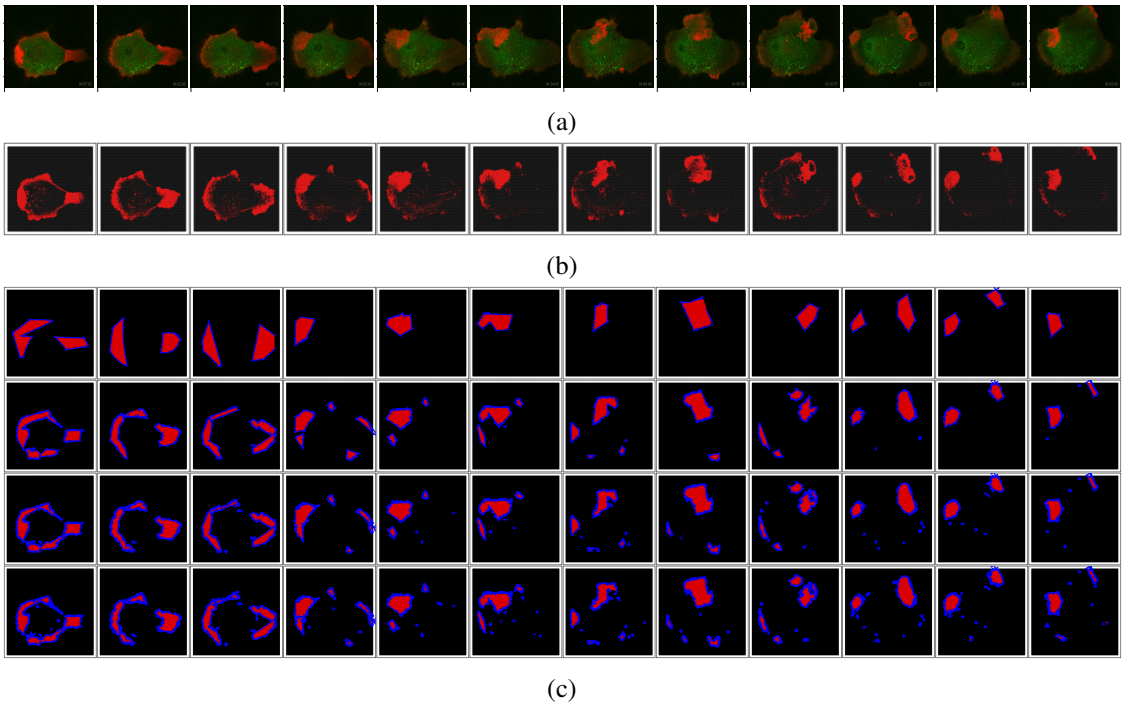
We conduct another experiment to evaluate the computational cost of our SMC approach. As indicated in Fig. 5.5, the running time of SMC is a linear function of the number of particles. A larger number of particles increase the computational cost. But we can parallelize the propagation and weighting steps to decrease the cost by allocating samples across different cores. The computational cost decreases more than 10 times if we use 20 cores. This shows one advantage of using our SMC approach over traditional Bayesian inference methods such as Markov chain Monte Carlo.

5.3.2 Experiment on an HIV-1–infected human macrophage

We evaluate the performance of the RTS on a series of images of Human Immunodeficiency Virus Type I (HIV-1) infected human macrophage. We also display the results provided by some popular image segmentation approaches, such as the Simple Linear Iterative Clustering (SLIC) Superpixels (Achanta et al., 2010, 2012), zero version of the SLIC (SLICO) (Ren and Malik, 2003; Lucchi et al., 2010), affinity propagation clustering based on the SLIC superpixels (SLICAP) (Frey and Dueck, 2007; Zhou, 2015) and supervised classification algorithms methods, such as the K-nearest neighbor

classifier (KNN), decision trees (DT), random forest (RF) (with 100 trees), support vector machines (SVM).

The images are extracted from a real-time video showing HIV-1–infected human macrophage sensing its environment (Gaudin). Images are taken from stills every second from the video, with 12 images are acquired (see Fig.5.6a). All images are converted to binary images and each binary image contains 600×460 pixels. We reduce the size of each image to 20% of its original size (*i.e.* 120×92 pixels), and these images are assumed to be the ground truth (see Fig.5.6b). Our goal is to capture the smooth shape of HIV-1–infected human macrophage for each image. In this experiment, we use 2000 particles, 16 cores, and set $\tau = \infty$ (i.e., , running the RTS until the pausing conditions are met for all partitions). The rest of the parameters are set to the default values described in the beginning of this section.



(a)

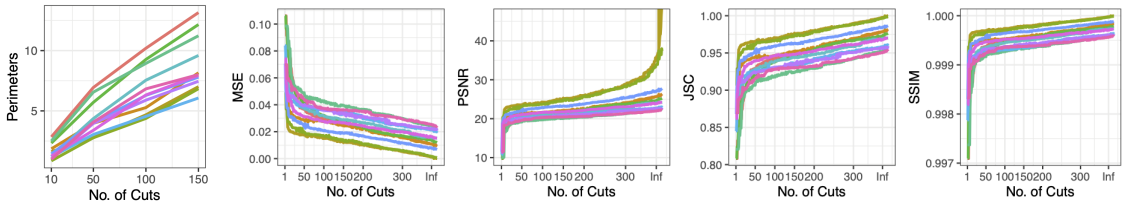
(b)

(c)

Figure 5.6: (*a*) Original Images. (*b*) Ground truth. (*c*) RTS produced image shapes (with shape boundaries in blue) with different number of cuts. Rows from top to bottom are the shapes produced by the RTS with 10, 50, 100, 150 cuts.

5.4 Results

Macrophage infection is a major aspect of HIV-1 Merrill and Chen (1991); Cunningham et al. (1997); Hrecka et al. (2011); Koppensteiner et al. (2012). Measures of HIV-1–infected human macrophages, such as the perimeters of infected cells may reveal aspects of HIV-1 etiology Castellano et al. (2017). To acquire these shapes within an image, we combine the curves that lie at the boundaries between subsets in the tessellation, and then mark the curve segments as interior boundaries if the K nearest pixels on both sides of the curve have the same label, and then remove the interior boundaries.



(a) (b)

Figure 5.7: (*a*) Estimated perimeters of shapes as a function of number of cuts for images 1 to 12. Each line represents one image.(*b*) Performance measures MSE, PSNR, JSC and SSIM as a function of the number of cuts for hiv dataset. For MSE, lower is better and for PSNR, SSIM and JSC higher is better.

This leaves only curves that delineate the boundary of the object. We sample points from the curves evenly and use the K-nearest neighbours algorithm to obtain $K = 10$ pixels for each point on the curve. We then extract properties of the resulting shape: In this experiment we focus on the perimeter. Fig. 5.6c displays the shapes of the macrophage captured by the RTS with different numbers of cuts, and Table 5.1 displays the corresponding estimated perimeters. The 12 extracted images in the Table are labelled by $I1$, $I2$, ... In Fig. 5.6c, overplotting (in which multiple splines indicate the same border) arises on the boundaries. The degree of overplotting increases linearly with the number of the cuts: Fig. 5.7 (*a*). This suggests that the true perimeter of the shape is proportional to $\tilde{p}/\#\text{cuts}$, where $\tilde{p}$ is the perimeter of the shape found by the RTS.

Table 5.1: Estimated perimeters of the shapes of the 12 images (I1 to I12) by the RTS with different numbers of cuts.

#cuts	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10	I11	I12
10	2.84	2.33	2.50	1.22	1.18	1.44	1.02	1.25	0.98	1.85	1.62	0.86
50	6.94	5.71	6.56	4.36	3.04	4.02	3.96	3.39	4.07	3.97	2.77	2.77
100	10.20	9.27	8.94	7.55	4.55	6.28	5.89	6.32	6.83	5.27	4.67	4.37
150	13.16	12.16	11.23	9.60	6.07	7.80	7.46	7.93	8.01	8.16	7.01	6.84

Table 5.2: Mean of quantitative measures PSNR, MSE, JSC, SSIM on images over different methods.

Methods	SVM	RF	DT	KNN	SLIC	SLICO	SLICAP	RTS
PSNR	11.9100	17.8337	15.2747	15.0990	17.2681	17.2713	17.2681	27.3121
MSE	0.0670	0.0172	0.0309	0.0320	0.0197	0.0196	0.0197	0.0132
JSC	0.8751	0.9662	0.9402	0.9381	0.9614	0.9615	0.9614	0.9741
SSIM	0.9983	0.9997	0.9995	0.9994	0.9996	0.9996	0.9996	0.9998

We compare the ground truth and predicted images with different number of cuts via both visualization (see Fig. 5.8 in Appendix A) and quantitative measures (Fig. 5.7b). Fig. 5.7b displays the mean squared error (*MSE*), peak signal-to-noise ratio (*PSNR*), Jaccard similarity coefficient (JSC) and structural similarity (SSIM) of RTS as a function of number of cuts. Each line denotes

a quantitative measure for one image. The *MSE*s decrease as the budget τ increase, and becomes stable when number of cuts exceed 25. The *PSNR*s, JSCs and SSIM increase rapidly for the first few cuts and become stable after 25 cuts. Table 5.2 shows the average of different quantitative measures on the images across the methods. Fig. 5.8(b) displays the predicted images versus number of cuts, with ground truth (Fig. 5.8(a)) , showing that a large number of cuts can better capture the shape of the macrophage. Comparing Fig. 5.8(b) to Fig. 5.8(c-d), our proposed RTS captures the shapes of the images comparable or better to other methods. Table 5.2 and Fig. 5.8 indicate that our proposed RTS with $\tau = \infty$ performs comparable to other methods in terms of the quantitative measures and visualization. The SVM in Table 5.2 uses radial basis kernels, the result of the SVM with other kernels in is given in Appendix A.

5.5 Conclusion

In this chapter, we present a novel Bayesian nonparametric method for space partitioning. Our method partitions a domain with curves to capture complex images shapes. We develop an inference algorithm that can be parallelized, and provide evidence towards consistency. Our real data application shows that the RTS has satisfactory performance in a task in which the smooth shape of HIV-1–infected human macrophage is captured, provided that enough computing resources are allocated (in terms of the number of particles and the computational budget τ). One future direction of this work is to extend the RTS to capture shapes of multidimensional objects with cutting manifolds.

Broader Impact

Our work involves a Bayesian nonparametric model for shapes. Theoretically, we consider the problem of generalizing a decision tree to use decision boundaries that are drawn from a parameterized set of curves. We illustrate a potential biomedical application: determination of the characteristics of the shapes of cells. Our aim with that application is to advance medicine by providing a method for determining the perimeter of cells or microstructures which may be correlated with severity of disease. Our work towards this biomedical application is in early stages and is currently a proof-of-concept. Positive impact in medicine in our intended direction would require much additional work involving acquisition of datasets, and collaboration with pathologists.

Bayesian nonparametric methods are general, and can be applied to any problem. For example, Bayesian nonparametric versions of latent dirichlet allocation (LDA) have been used extensively by Facebook and sometimes in possibly unhelpful gender discourse Wang et al. (2013). In addition, general advances in image processing and image analysis (our work is also a contribution to this) in machine learning have led to a cat-and-mouse game of detection and generation in image manipulation (deep fakes), and erosion of privacy through technologies such as automatic name-plate recognition. The primary application we consider in this chapter is biomedical, and is in the public interest. However, much work is still required for its potential to become realized to that end.

5.6 Appendix A

5.6.1 Results of different methods on an HIV-1–infected human macrophage

In this section, we provide the results of different methods on a series of images of HIV-1 infected human macrophage. Fig.5.8 gives the comparison of the ground truth and predicted images of different methods, include the RTS with different number of cuts, DT, KNN, SLIC, SLICO, SLICAP and SVM with different kernels. Table 5.3 presents the mean of quantitative measures of PSNR, MSE, JSC and SSIM of the SVM with different kernels on the 12 images. These metrics are described in the main text. And the mean of these quantitative measures of other methods is given in the Table 5.2 of the main text. Figure 5.8 provides the reconstruction of the shape of the macrophage, according to various methods, as described in the main text.

Table 5.3: Mean of quantitative measures PSNR, MSE, JSC, SSIM on images over the SVM with different kernels.

Methods	Radial	Polynomial	Linear	Sigmoid
PSNR	11.9100	11.1974	11.1974	8.3061
MSE	0.0670	0.0779	0.0779	0.1516
JSC	0.8751	0.8560	0.8560	0.7383
SSIM	0.9983	0.9978	0.9978	0.9974

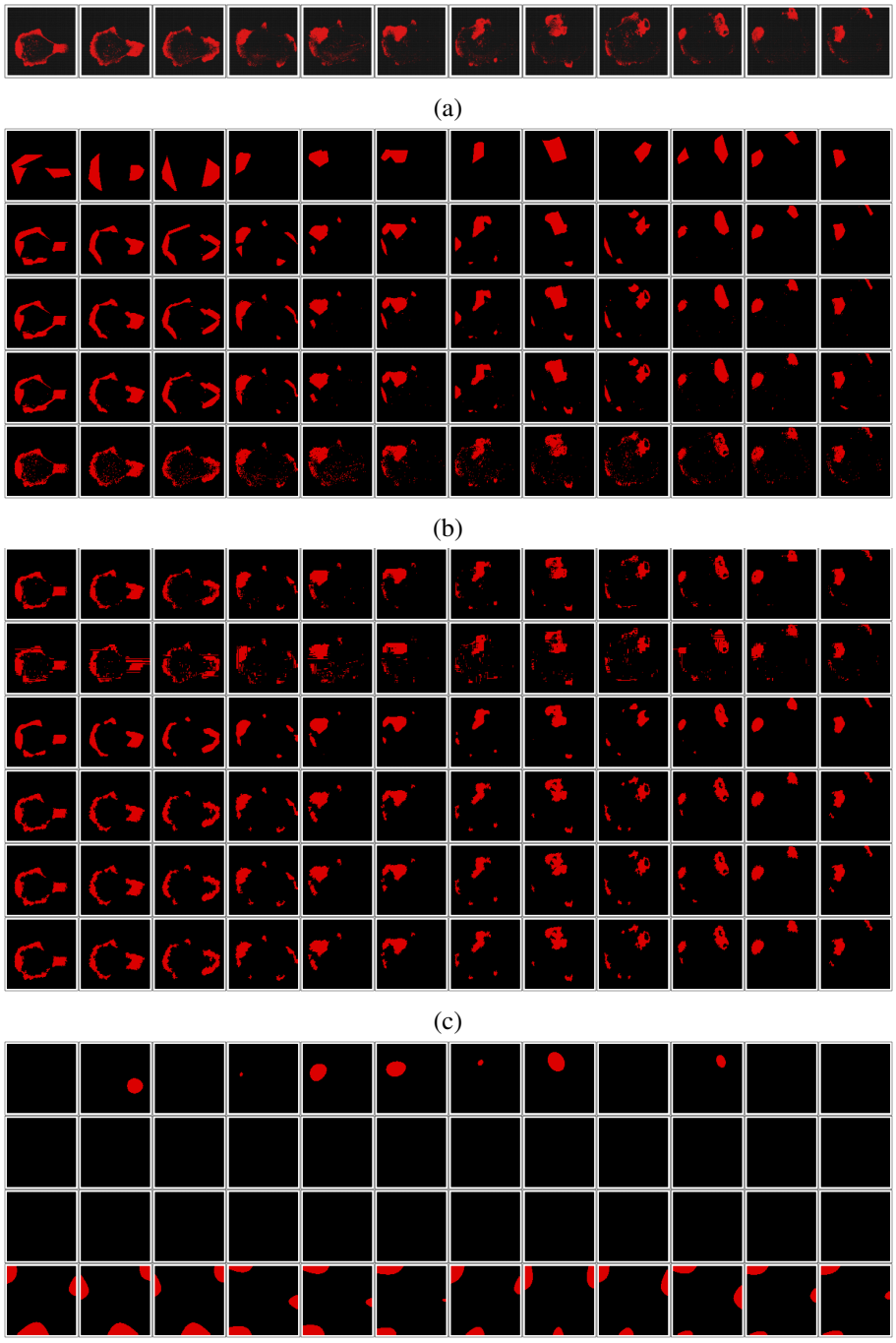


Figure 5.8: (*a*) Ground truth. (*b*) RTS predictions with 10, 50, 100, 150, ∞ cuts from top to bottom. (*c*) From top to bottom, predictions of the RF (with 100 trees), DT, KNN, SLIC, SLICO, SLICAP. (*d*) Predicted images of the SVM with different kernels. From top to bottom, predictions of the SVM with radial basis function, polynomial, linear and sigmoid kernels.

Chapter 6

Online Bayesian learning for mixtures of spatial spline regressions with mixed-effects

6.1 Introduction

Classification and clustering of random objects has been received substantial attention in functional data analysis. James and Hastie (2001) proposed the functional linear discriminant analysis, which can perform the classification of curves, by extending the classical method of linear discriminant analysis (Izenman, 2013) to functional data. Motivated by radar-range problems, Hall et al. (2001) developed the classification of Gaussian process regression to do real-time discrimination in the context of signal analysis based on principal coordinates analysis. James and Sugar (2003) generalized the clustering of functional data based on mixtures of linear mixed models, and their method is particularly well suited for sparsely sampled data. Chamroukhi et al. (2010) proposed the clustering of curves based on piecewise polynomial regression. These authors combined the piecewise polynomial regression with a discrete hidden regression model to accommodate abrupt or smooth changes in curves.

Although not as abundant as univariate functions, bivariate functional data analysis has also been well-developed as a statistical tool in surface smoothing and estimation. Matheron (1963) introduced a geostatistical procedure called "kriging", which predicts the value of a panel at any arbitrary location via a weighted average of available samples. Duchon (1977) proposed the use of thin-plate splines as an interpolation method for surfaces and a penalty term was introduced to control the smoothness of the fitted surface. Malfait and Ramsay (2003) proposed a spatial spline regression (SSR) model to deal with functional time series. Wood et al. (2008) showed that conventional smoothing methods may not work well for complex domains, due to poor performance at the boundary of these domains. These authors discuss a technique known as soap film smoothing which can be applied for smoothing over difficult subregions of $\mathbb{R}^2$. The aforementioned work has a main focus on surface smoothing and estimation for a single population rather than clustering of surfaces from several populations.

Nguyen et al. (2016) proposed a new application of bivariate functions to do clustering and classification for surfaces by extending the clustering techniques of James and Sugar (2003) to mixtures of spatial spline regression with mixed-effects model. We will refer to mixture of spatial spline regression with mixed-effects as "MSSR_m" in what follows. Nguyen et al. (2016) fitted the MSSR_m model using the Expectation-Maximization (EM) algorithm (Dempster et al., 1977). In the E-step the expectation of log-likelihood function is computed over the latent variables conditioned on the observations and the current parameter estimates. In the M-step the expected log-likelihood obtained in E-step is maximized over the model parameters. The EM algorithm is efficient in finding a local mode of likelihood function but it is very sensitive to the initial values and it only provides point estimates.

An alternative approach to implement inference in the latent variable model is Markov chain Monte Carlo (MCMC) (Jasra et al., 2005; Diebolt and Robert, 1994). Besides point estimates, MCMC provides uncertainty quantification for the parameters of interest. Chamroukhi (2015); Chamroukhi and Nguyen (2019) consider Bayesian approaches to the problem of static estimation for spatial spline regression. However, there are several constraints for an MCMC algorithm to be implemented for inference in an online problem. First of all, the storage of a "big data" set may cause memory issues in practice. Second, the MCMC has to be rerun to update parameters when a new observation arrives, which is inefficient for dynamic problems. Finally, the marginal likelihood is expensive and challenging to compute in an MCMC algorithm.

Sequential Monte Carlo (SMC) (Doucet et al., 2001; Liu and Chen, 1998; Doucet et al., 2000), aka Particle Filtering, provides a sequential approximate of the posterior distribution using sampled particles (samples), and is often used to learn the distribution of latent variables. SMC has been demonstrated as a gold standard for dynamic problems, which leads to interest in implementing this method to many batch problems as an alternative to traditional Bayesian computation methods (e.g. MCMC). The sequential scheme in SMC makes online inference in latent variable models possible. Some sophisticated work has been done on mixture models using SMC (Fearnhead and Meligkotsidou, 2007; Carvalho et al., 2010b; MacEachern et al., 1999). When a new observation arrives, within this framework we only need to update the uncertainty immediately using this new observation and the low dimensional sufficient statistics we have stored, which means every observation is only used once. This is especially suitable for large-scale stream type data. In addition, another advantage of our proposed SMC algorithm over MCMC is that the marginal likelihood calculation is very convenient, which can be used for model selection. The posterior cluster allocation is also obtained from the algorithm.

In this chapter, we build a Bayesian hierarchical model based on the MSSR_m as described in Section 6.3.1, in which we take the cluster labels as latent variables. The latent variables of the model are discrete and evolve with time. We apply the SMC scheme to learn the distribution of these latent variables and static parameters of the model. As described in Algorithm 11, we take the Resample-Propagate strategy to reduce the approximation errors since both the resample and propagate step will be informed by the new observation (Carvalho et al., 2010a,b; Pitt and Shephard,

1999). In addition, we include MCMC moves within the SMC algorithm to mutate particles so as to prevent the progressive degeneration (Fearnhead, 2002; Gilks and Berzuini, 2001). Moreover, with the adoption of model sufficient statistics, instead of requiring storage of an entire data set, only sufficient statistics are required to summarize the data in the MCMC move, and this scheme makes the SMC algorithm computationally more efficient.

In this chapter, we have three main contributions. First, we derive the full MCMC (Gibbs) sampling scheme for the MSSR_m model. Second, we propose an online SMC algorithm for image clustering based on the MSSR_m model, which allows us to conduct model inference for sequential images efficiently. Our proposed method is related to the work in Carvalho et al. (2010b). Their method is applied to relatively simple mixture models. In contrast, our spatial spline mixture has a more complicated model structure, high dimensional model parameters and a large number of random effects. Third, we numerically assess the performance of our algorithm with different number of latent states (clusters) and observations.

The remainder of this chapter is organized as follows. We introduce the mixture of spatial spline regression model in Section 6.2. In Section 6.3, we derive the Gibbs sampling algorithm for the mixture of spatial spline regression and propose an online inference scheme for this model. In Section 6.4, we introduce the model selection criteria for the proposed online SMC algorithm. In Section 6.5, we use a numerical study to assess and compare the performance of MCMC and the proposed online SMC algorithm. In Section 6.6, we apply the methods to hand writing recognition data and brain image magnetoencephalography (MEG) data. We list all notations in the 6.8.1.

6.2 Mixture of spatial spline regression model

The spatial spline regression (SSR) dates back to Malfait and Ramsay (2003); Sangalli et al. (2013). Malfait and Ramsay (2003) first applied the SSR model to functional time series. Sangalli et al. (2013) refined the SSR model and showed that it can be extended to areal data with three dimensions, including volumes and surfaces that are embedded in three-dimensional spaces. Nguyen et al. (2016) implemented this methodology to surface clustering and classification by extending SSR model to MSSR_m model. In this section, we will describe the MSSR_m Model.

6.2.1 Mixture of spatial spline regression model with mixed-effects

Let t be an index for time, at time t, $t = 1, \ldots, T$, denote $\boldsymbol{y}_t$, an $m_t \times 1$ vector, as the observation at time t, where m_t is the length of observation $\boldsymbol{y}_t$. Suppose observations $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_T$ can be grouped into K clusters. Let $\boldsymbol{S}_t$ be the spatial covariates matrix at time t. The matrix $\boldsymbol{S}_t$ has dimension $m_t \times d$ and is calculated from the tent shaped piecewise linear Nodal basis functions (NBFs) (Malfait and Ramsay, 2003), which will be introduced in Section 6.2.2, and d is fixed and refers to the number of basis functions. Let $^\intercal$ be a symbol denoting the transpose of a vector or matrix. Let $\boldsymbol{\pi} = (\pi_1, \pi_2, \ldots, \pi_K)^\intercal$ be the cluster allocation probability for observations and let $\boldsymbol{\beta}_k$, a $d \times 1$ vector, be the model fixed effects for cluster k. We use $\boldsymbol{b}_{tk}$, a $d \times 1$ vector, to denote the random effect coefficients of cluster k for

$\boldsymbol{y}_t$. We let $\boldsymbol{e}_{tk}$, a $d \times 1$ vector, represent the random error of cluster k for observation t. The MSSR_m model for the observation at time t can be expressed as

$$\boldsymbol{y}_t = \sum_{k=1}^{K} \pi_k(\boldsymbol{S}_t\boldsymbol{\beta}_k + \boldsymbol{S}_t\boldsymbol{b}_{tk} + \boldsymbol{e}_{tk}), \tag{1}$$

where $\boldsymbol{b}_{tk} \sim MVN(0, \xi_k^2\boldsymbol{I}_d)$ and $\boldsymbol{e}_{tk} \sim MVN(0, \sigma_k^2\boldsymbol{I}_{m_t})$. Here $MVN(\cdot,\cdot)$ denotes the multivariate normal distribution, and $\boldsymbol{I}_l$ denotes an identity matrix with dimension $l \times l$. Denote $\boldsymbol{\pi} = (\{\pi_k\}_{k=1}^K)^\intercal$, $\boldsymbol{\beta} = (\{\boldsymbol{\beta}_k\}_{k=1}^K)^\intercal$, $\boldsymbol{\sigma}^2 = (\{\sigma_k^2\}_{k=1}^K)^\intercal$ and $\boldsymbol{\xi}^2 = (\{\xi_k^2\}_{k=1}^K)^\intercal$. Our interest is to make Bayesian inference for $\boldsymbol{\theta} = (\boldsymbol{\pi}^\intercal, \boldsymbol{\beta}^\intercal, (\boldsymbol{\sigma}^2)^\intercal, (\boldsymbol{\xi}^2)^\intercal)^\intercal$.

6.2.2 Linear nodal basis function

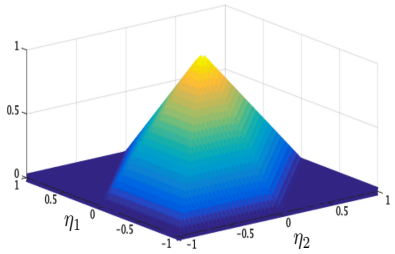
In this section, we focus on the nodal basis functions (NBFs) (Malfait and Ramsay, 2003). While B-splines are often used to approximate the univariate functions, NBFS can be used in a similar way to approximate surfaces. As argued by Nguyen et al. (2016), the linear NBFs (Malfait and Ramsay, 2003) are useful for problems involving clustering and classification.

The linear NBF is a "tent shaped" piecewise linear function with shape parameter $\boldsymbol{\delta} = (\delta_1, \delta_2)^\intercal$, center parameter $\mathbf{c} = (c_1, c_2)^\intercal$ and coordinates x_1, x_2 on a rectangular domain, where δ_1 is a positive real number representing the horizontal shape parameter, and δ_2 is a positive real number representing the vertical shape parameter. For any coordinates $\boldsymbol{\eta} = (\eta_1, \eta_2)^\intercal$ on a rectangular domain $R = [\eta_1^-, \eta_1^+] \times [\eta_2^-, \eta_2^+]$, let $\eta_1' = (\eta_1 - c_1)/\delta_1$, $\eta_2' = (\eta_2 - c_2)/\delta_2$, the exact form of linear NBF of $\boldsymbol{\eta}$ is defined in Eq.(2) (Malfait and Ramsay, 2003; Sangalli et al., 2013; Nguyen et al., 2016),

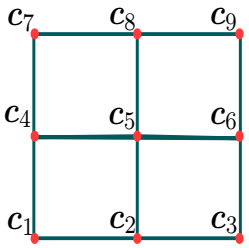
$$s(\boldsymbol{\eta}; \mathbf{c}, \boldsymbol{\delta}) = \begin{cases} 1 + \eta_2' & \text{if } \boldsymbol{\eta}' \in \{(\eta_1', \eta_2') : -1 \le \eta_1' \le 0,\ -1 \le \eta_2' \le \eta_1'\}, \\ 1 + \eta_1' & \text{if } \boldsymbol{\eta}' \in \{(\eta_1', \eta_2') : -1 \le \eta_1' \le 0,\ \eta_1' \le \eta_2' \le 0\}, \\ 1 + \eta_1' - \eta_2' & \text{if } \boldsymbol{\eta}' \in \{(\eta_1', \eta_2') : -1 \le \eta_1' \le 0,\ 0 \le \eta_2' \le \eta_1' + 1\}, \\ 1 - \eta_1' + \eta_2' & \text{if } \boldsymbol{\eta}' \in \{(\eta_1', \eta_2') : 0 \le \eta_1' \le 1,\ \eta_1' - 1 \le \eta_2' \le 0\}, \\ 1 - \eta_1' & \text{if } \boldsymbol{\eta}' \in \{(\eta_1', \eta_2') : 0 \le \eta_1' \le 1,\ 0 \le \eta_2' \le \eta_1'\}, \\ 1 - \eta_2' & \text{if } \boldsymbol{\eta}' \in \{(\eta_1', \eta_2') : 0 \le \eta_1' \le 1,\ \eta_1' \le \eta_2' \le 1\}, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Figure 6.1a is an example of a NBF function defined on a rectangular domain $R = [-1, 1] \times [-1, 1]$, with $\boldsymbol{\delta} = (1, 1)^\intercal$ and $\mathbf{c} = (0, 0)^\intercal$.

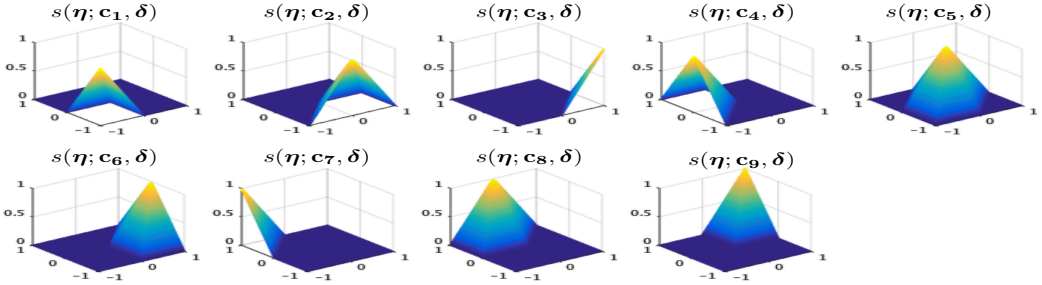
Similarly to B-splines, the number of nodal basis functions used to approximate surfaces has to be specified. Suppose we use $d = d_1 \times d_2$ basis functions to approximate the surfaces over a fixed rectangular domain. The rectangular domain is divided into $(d_1 - 1) \times (d_2 - 1)$ small grids evenly, $d_1 - 1$ grids in each row and $d_2 - 1$ grids in each column. Nodes on the grids are centers of these nodal basis, and the horizontal length and vertical height on small grids are $\delta_1 = (\eta_1^+ - \eta_1^-)/(d_1 - 1)$,



(a) $s(\boldsymbol{\eta}; \mathbf{c}, \boldsymbol{\delta})$, with $\mathbf{c} = (0,0)^\intercal$ and $\boldsymbol{\delta} = (1,1)^\intercal$.



(b) centers of 3×3 NBFs on a fixed domain.



(c) 3×3 NBFs on fixed rectangular domain $R = [-1,1] \times [-1,1]$.

Figure 6.1: An example of Nodal Basis Functions.

$\delta_2 = (\eta_2^+ - \eta_2^-)(d_2 - 1)$, respectively. The covariates matrix $\boldsymbol{S}_t$ for $\boldsymbol{y}_t$, $t = 1, \ldots, T$, is defined as

$$\boldsymbol{S}_t = \begin{pmatrix} s(\boldsymbol{\eta}_{t,1}; \boldsymbol{c}_1, \boldsymbol{\delta}) & \ldots & s(\boldsymbol{\eta}_{t,1}; \boldsymbol{c}_d, \boldsymbol{\delta}) \\ s(\boldsymbol{\eta}_{t,2}; \boldsymbol{c}_1, \boldsymbol{\delta}) & \ldots & s(\boldsymbol{\eta}_{t,2}; \boldsymbol{c}_d, \boldsymbol{\delta}) \\ \vdots & \ddots & \vdots \\ s(\boldsymbol{\eta}_{t,m_t}; \boldsymbol{c}_1, \boldsymbol{\delta}) & \ldots & s(\boldsymbol{\eta}_{t,m_t}; \boldsymbol{c}_d, \boldsymbol{\delta}) \end{pmatrix}, \tag{3}$$

where $\boldsymbol{\eta}_{t,1}, \ldots, \boldsymbol{\eta}_{t,m_t}$ are coordinates of $\boldsymbol{y}_t$, $\boldsymbol{\delta} = (\delta_1, \delta_2)^\intercal = ((\eta_1^+ - \eta_1^-)/(d_1 - 1), (\eta_2^+ - \eta_2^-)/(d_2 - 1))^\intercal$, and $\boldsymbol{c}_l = (c_{l1}, c_{l2})^\intercal$, $l = 1, \ldots, d$ are the centers and can be obtained by setting $\boldsymbol{c}_1 = (\eta_1^-, \eta_2^-)^\intercal$, $\boldsymbol{c}_2 = (\eta_1^- + \delta_1, \eta_2^-)^\intercal$, …, $\boldsymbol{c}_{d-1} = (\eta_1^+, \eta_2^- + (d_2 - 1)\delta_2)^\intercal$, $\boldsymbol{c}_d = (\eta_1^+, \eta_2^+)^\intercal$. For example, given a fixed rectangular domain $R = [-1,1] \times [-1,1]$, suppose we set the number of NBFs $d = 3 \times 3$, therefore $\delta_1 = \delta_2 = 1$, and the corresponding 9 NBFs centers are $\mathbf{c}_1 = (-1,-1)^\intercal$, $\mathbf{c}_2 = (-1,0)^\intercal$, $\mathbf{c}_3 = (-1,1)^\intercal$, $\mathbf{c}_4 = (0,-1)^\intercal$, $\mathbf{c}_5 = (0,0)^\intercal$, $\mathbf{c}_6 = (0,1)^\intercal$, $\mathbf{c}_7 = (1,-1)^\intercal$, $\mathbf{c}_8 = (1,0)^\intercal$ and $\mathbf{c}_9 = (1,1)^\intercal$. These centers and their corresponding NBFs are shown in Figure 6.1b and 6.1c, respectively.

6.3 Model inference

In this section, we introduce two methodologies to make inference on the mixture of spatial spline regression model. Before introducing the inference methods, we discuss the Bayesian framework for this MSSR_m model. We first incorporate the auxiliary variable z_t ($t = 1, \ldots, T$) in the mixture model to indicate the cluster label of observation $\boldsymbol{y}_t$. Denote $\boldsymbol{Y} = (\{\boldsymbol{y}_t^\intercal\}_{t=1}^T)^\intercal$, $\boldsymbol{Z} = (\{z_t\}_{t=1}^T)^\intercal$, $\boldsymbol{b} = (\{\boldsymbol{b}_t^\intercal\}_{t=1}^T)^\intercal$, $\boldsymbol{b}_t = \{\boldsymbol{b}_{t1}^\intercal, \ldots, \boldsymbol{b}_{tK}^\intercal\}^\intercal$. Recall that $\boldsymbol{\theta} = (\boldsymbol{\pi}^\intercal, \boldsymbol{\beta}^\intercal, (\sigma^2)^\intercal, (\boldsymbol{\xi}^2)^\intercal)^\intercal$, conditional on z_t, $\boldsymbol{b}_t$ and $\boldsymbol{\theta}$, we have

$$f(\boldsymbol{y}_t|z_t, \boldsymbol{b}_t, \boldsymbol{\theta}) = \prod_{k=1}^{K} \left\{\phi(\boldsymbol{y}_t; \boldsymbol{S}_t\boldsymbol{\beta}_k + \boldsymbol{S}_t\boldsymbol{b}_{tk}, \sigma_k^2\boldsymbol{I}_{m_t})\right\}^{\mathbf{1}_k(z_t)}, \tag{4}$$

where $\phi(\cdot; \boldsymbol{S}_t\boldsymbol{\beta}_k + \boldsymbol{S}_t\boldsymbol{b}_{tk}, \sigma_k^2\boldsymbol{I}_{m_t})$ denotes the density of a multivariate normal distribution with mean $\boldsymbol{S}_t\boldsymbol{\beta}_k + \boldsymbol{S}_t\boldsymbol{b}_{tk}$, and covariance $\sigma_k^2\boldsymbol{I}_{m_t}$. $\mathbf{1}(\cdot)$ is a indicator function: $\mathbf{1}_k(z_t) = 1$ if $z_t = k$, otherwise $\mathbf{1}_k(z_t) = 0$.

The joint density (likelihood) of $\boldsymbol{Y}$ conditional on $\boldsymbol{Z}, \boldsymbol{b}, \boldsymbol{\theta}$ can be written as

$$f(\boldsymbol{Y}|\boldsymbol{Z}, \boldsymbol{b}, \boldsymbol{\pi}, \boldsymbol{\beta}, \boldsymbol{\xi}^2, \boldsymbol{\sigma}^2) = \prod_{t=1}^{T}\prod_{k=1}^{K} \left\{\phi(\boldsymbol{y}_t; \boldsymbol{S}_t\boldsymbol{\beta}_k + \boldsymbol{S}_t\boldsymbol{b}_{tk}, \sigma_k^2\boldsymbol{I}_{m_t})\right\}^{\mathbf{1}_k(z_t)}. \tag{5}$$

Taking integration of Eq.(4) over $\boldsymbol{b}_t$, we have

$$f(\boldsymbol{y}_t|z_t, \boldsymbol{\theta}) = \sum_{k=1}^{K} \mathbf{1}_k(z_t)\phi(\boldsymbol{y}_t; \boldsymbol{S}_t\boldsymbol{\beta}_k, \xi_k^2\boldsymbol{S}_t\boldsymbol{S}_t^\top + \sigma_k^2\boldsymbol{I}_{m_t}). \tag{6}$$

Taking integration of Eq.(4) over z_t and $\boldsymbol{b}_t$, we have

$$f(\boldsymbol{y}_t|\boldsymbol{\theta}) = \sum_{k=1}^{K} \pi_k\phi(\boldsymbol{y}_t; \boldsymbol{S}_t\boldsymbol{\beta}_k, \xi_k^2\boldsymbol{S}_t\boldsymbol{S}_t^\top + \sigma_k^2\boldsymbol{I}_{m_t}). \tag{7}$$

6.3.1 Bayesian framework

In this section, we build a hierarchical MSSR$_m$ model under the Bayesian framework. Let a Dirichlet distribution with hyperparameters $(\alpha_1, \ldots, \alpha_K)$ be the prior for the allocation parameter $\boldsymbol{\pi}$, a multinomial distribution with parameter $\boldsymbol{\pi}$ be the prior for the label z_t, $t = 1, \ldots, T$, a multivariate normal distribution with hyperparameters $\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0$ be the prior of the fixed-effects coefficients. We use an inverse-gamma distribution with hyperparameters a_0, b_0 (or g_0, h_0) as the prior for the variance parameter ξ_k^2 (or σ_k^2) for $k = 1, \ldots, K$ and use a multivariate normal distribution with parameters ξ_k^2 to model the random-effects coefficients $\boldsymbol{b}_{tk}$ for $k = 1, \ldots, K$, $t = 1, \ldots, T$. The hierarchical MSSR$_\mathrm{m}$ model can be expressed as follows:

$$\begin{aligned}
\boldsymbol{y}_t|z_t = k, \boldsymbol{\beta}_k, \boldsymbol{b}_{tk}, \sigma_k^2 &\sim MVN(\boldsymbol{S}_t\boldsymbol{\beta}_k + \boldsymbol{S}_t\boldsymbol{b}_{tk}, \sigma_k^2\boldsymbol{I}_{m_t}), && (8)\\
\boldsymbol{\pi} &\sim Dir(\alpha_1, \ldots, \alpha_K), && (9)\\
z_t &\sim Mult(\pi_1, \ldots, \pi_K),\ t = 1, \ldots, T, && (10)\\
\boldsymbol{\beta}_k &\sim MVN(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0),\ k = 1, \ldots, K, && (11)\\
\boldsymbol{b}_{tk}|\xi_k^2 &\sim MVN(0_d, \xi_k^2\boldsymbol{I}_d),\ k = 1, \ldots, K,\ t = 1, \ldots, T, && (12)\\
\xi_k^2 &\sim IG(a_0, b_0),\ k = 1, \ldots, K, && (13)\\
\sigma_k^2 &\sim IG(g_0, h_0),\ k = 1, \ldots, K. && (14)
\end{aligned}$$

where $a_0, b_0, g_0, h_0, \alpha_1, \ldots, \alpha_K, \boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0$ are hyperparameters. Therefore, the normalized posterior distribution for $(\boldsymbol{\theta}^\intercal, \boldsymbol{b}^\intercal, \boldsymbol{Z}^\intercal)^\intercal = (\boldsymbol{\pi}^\intercal, \boldsymbol{\beta}^\intercal, (\boldsymbol{\sigma}^2)^\intercal, (\boldsymbol{\xi}^2)^\intercal, \boldsymbol{b}^\intercal, \boldsymbol{Z}^\intercal)^\intercal$ can be written as

$$p(\boldsymbol{\theta}, \boldsymbol{b}, \boldsymbol{Z}|\boldsymbol{Y}) = \frac{f(\boldsymbol{\pi}) \prod_{k=1}^{K} f(\boldsymbol{\beta}_k) f(\xi_k^2) f(\sigma_k^2) \prod_{t=1}^{T} \prod_{k=1}^{K} f(\boldsymbol{b}_{tk}|\xi_k^2) \left\{\phi(\boldsymbol{y}_t|\boldsymbol{S}_t\boldsymbol{\beta}_k + \boldsymbol{S}_t\boldsymbol{b}_{tk}, \sigma_k^2 \boldsymbol{I}_{m_t})\pi_k\right\}^{\mathbf{1}_k(z_t)}}{f(\boldsymbol{Y})},$$

where $f(\boldsymbol{Y})$ refers to the marginal likelihood and can be evaluated by

$$\begin{aligned} f(\boldsymbol{Y}) \quad = \quad & \int \ldots \int f(\boldsymbol{\pi}) \prod_{k=1}^{K} f(\boldsymbol{\beta}_k) f(\xi_k^2) f(\sigma_k^2) \\ & \times \prod_{t=1}^{T} \prod_{k=1}^{K} f(\boldsymbol{b}_{tk}|\xi_k^2) \left\{\phi(\boldsymbol{y}_t|\boldsymbol{S}_t\boldsymbol{\beta}_k + \boldsymbol{S}_t\boldsymbol{b}_{tk}, \sigma_k^2 \boldsymbol{I}_{m_t})\pi_k\right\}^{\mathbf{1}_k(z_t)} d\boldsymbol{\theta}\, d\boldsymbol{b}\, d\boldsymbol{Z}. \end{aligned} \tag{15}$$

However, the integral in Eq.(15) is intractable. We propose two methods to estimate the normalized posterior distribution for $\boldsymbol{\theta}$, which will be described in the next two sections.

6.3.2 Markov chain Monte Carlo

Markov chain Monte Carlo is the most commonly used approach for the implementation of Bayesian inference. The basic idea is to construct an ergodic irreducible Markov chain which admits the normalized posterior as its stationary and limiting distribution. The algorithm is run sufficiently long for a burn-in period so that subsequent draws after this period are approximate draws from the posterior. The availability of the conditional posterior distribution for all parameters of interest allows us to estimate the mixture of spatial spline regression in the Gibbs sampling framework (see Algorithm 9). The full conditional distributions is derived in Section 6.9.1.

Algorithm 9 Markov chain Monte Carlo (MCMC) algorithm - Gibbs sampler

Input: data $\boldsymbol{y}_{1:T}$, initial parameters $\{\boldsymbol{\theta}^{(0)\intercal}, \boldsymbol{b}^{(0)\intercal}\}^\intercal$, where $\boldsymbol{\theta}^{(0)\intercal} = (\boldsymbol{\pi}^{(0)\intercal}, \boldsymbol{\beta}^{(0)\intercal}, \boldsymbol{\sigma}^{2(0)\intercal}, \boldsymbol{\xi}^{2(0)\intercal})^\intercal$.
Output: $\{\boldsymbol{\theta}^{(i)\intercal}, \boldsymbol{b}^{(i)\intercal}\}_{1\le i\le N^*}^\intercal$, where $\boldsymbol{\theta}^{(i)\intercal} = (\boldsymbol{\pi}^{(i)\intercal}, \boldsymbol{\beta}^{(i)\intercal}, \boldsymbol{\sigma}^{2(i)\intercal}, \boldsymbol{\xi}^{2(i)\intercal})^\intercal$, N^* is total number of iterations.
for $i = 1$ **to** N^* **do**
    for $t = 1$ **to** T **do**
        Sample $z_t^{(i)} \sim p(\cdot|\boldsymbol{y}_t, \boldsymbol{\pi}^{(i-1)}, \boldsymbol{\beta}^{(i-1)}, \boldsymbol{\sigma}^{2(i-1)}, \boldsymbol{\xi}^{2(i-1)}, \boldsymbol{b}^{(i-1)})$ according to Eq.(21).
    Sample $\boldsymbol{\pi}^{(i)} \sim p(\cdot|\boldsymbol{Y}, \boldsymbol{Z}^{(i)}, \boldsymbol{\beta}^{(i-1)}, \boldsymbol{\sigma}^{2(i-1)}, \boldsymbol{\xi}^{2(i-1)}, \boldsymbol{b}^{(i-1)})$ according to Eq.(22).
    for $k = 1$ **to** K **do**
        Sample $\boldsymbol{\beta}_k^{(i)} \sim p(\cdot|\boldsymbol{Y}, \boldsymbol{Z}^{(i)}, \boldsymbol{\pi}^{(i)}, \sigma_k^{2(i-1)}, \xi_k^{2(i-1)}, \{\boldsymbol{b}_{tk}^{(i-1)}\}_{t=1}^T)$ according to Eq.(23).
        for $t = 1$ **to** T **do**
            Sample $\boldsymbol{b}_{tk}^{(i)} \sim p(\cdot|\boldsymbol{Y}, \boldsymbol{Z}^{(i)}, \boldsymbol{\pi}^{(i)}, \boldsymbol{\beta}_k^{(i)}, \sigma_k^{2(i-1)}, \xi_k^{2(i-1)})$ according to Eq.(24).
        Sample $\sigma_k^{2(i)} \sim p(\cdot|\boldsymbol{Y}, \boldsymbol{Z}^{(i)}, \boldsymbol{\pi}^{(i)}, \boldsymbol{\beta}_k^{(i)}, \xi_k^{2(i-1)}, \{\boldsymbol{b}_{tk}^{(i)}\}_{t=1}^T)$ according to Eq.(25).
        Sample $\xi_k^{2(i)} \sim p(\cdot|\boldsymbol{Y}, \boldsymbol{Z}^{(i)}, \boldsymbol{\pi}^{(i)}, \boldsymbol{\beta}_k^{(i)}, \sigma_k^{2(i)}, \{\boldsymbol{b}_{tk}^{(i)}\}_{t=1}^T)$ according to Eq.(26).

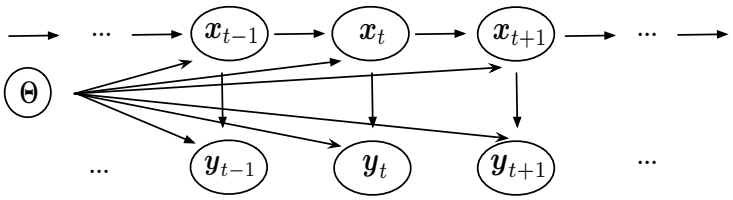


Figure 6.2: Graphical representation of a simple state space model.

6.3.3 Online sequential Monte Carlo with Gibbs moves

Sequential Monte Carlo methods (Doucet et al., 2001; Liu and Chen, 1998) were developed to analyze dynamical models. The most popular application is in state space models. In a state space model, which is graphically displayed in Figure 6.2, we let $\boldsymbol{x}_t$ denote the latent variable at time t, and assume it is specified by $f_{\boldsymbol{\theta}}(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})$, where $\boldsymbol{\theta}$ refers to the model static parameter. Let $\boldsymbol{y}_t$ denote the observation at time t. We assume that the observation $\boldsymbol{y}_t$ is independent conditional on $\boldsymbol{x}_t$, and it is specified by $f_{\boldsymbol{\theta}}(\boldsymbol{y}_t|\boldsymbol{x}_t)$. For simplicity, we will adopt the notation $\boldsymbol{a}_{1:t}$ to be an abbreviation of $\{\boldsymbol{a}_1, \ldots, \boldsymbol{a}_t\}$. The target distribution is $p_{\boldsymbol{\theta}}(\boldsymbol{x}_{1:T}|\boldsymbol{y}_{1:T}) \propto \prod_{t=1}^{T} f_{\boldsymbol{\theta}}(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) f_{\boldsymbol{\theta}}(\boldsymbol{y}_t|\boldsymbol{x}_t)$.

To approximate $p_{\boldsymbol{\theta}}(\boldsymbol{x}_{1:T}|\boldsymbol{y}_{1:T})$ using a standard SMC algorithm described in Algorithm 10, we introduce a sequence of intermediate target distributions $p_{\boldsymbol{\theta}}(\boldsymbol{x}_{1:t}|\boldsymbol{y}_{1:t})$ ($t = 1, 2, \ldots, T$). Each intermediate target distribution is approximated by a set of weighted samples, also called particles in the SMC literature. More specifically, at time t, $p_{\boldsymbol{\theta}}(\boldsymbol{x}_{1:t}|\boldsymbol{y}_{1:t})$ is approximated by $\{\boldsymbol{x}_{1:t}^{(i)}, W_t^{(i)}\}_{1\leq i\leq N}$. In our notation, superscripts and subscripts respectively refer to the particle and time indices.

The standard SMC algorithm iterates between the following three steps to approximate the intermediate targets: resampling, propagating and weighting. At each iteration t, we first conduct a resampling step to prune particles at iteration $t-1$ with small weights. The path degeneracy issue is well known in SMC literature (Lin et al., 2013; Doucet et al., 2006). The earlier approximation of intermediate target distributions may collapse as t increases. The purpose of resampling step is to alleviate the path degeneracy issue. A commonly used resampling algorithm is multinomial resampling. Equally weighted particles are produced after multinomial resampling. Then we propose new particles from a proposal distribution. Finally, we compute the weights for each proposed particle.

In Algorithm 10, $q_{t,\boldsymbol{\theta}}(\cdot)$ denotes the proposal distribution for $\boldsymbol{x}_t$, $w_t^{(i)}$ refers to the unnormalized weight and we use $W_t^{(i)}$ to represent the normalized version. If we choose the proposal distribution for $\boldsymbol{x}_t^{(i)}$ to be the prior $f_{\boldsymbol{\theta}}(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}^{(A_t^i)})$, we can obtain a simplified weight update form of $w_t^{(i)} = f_{\boldsymbol{\theta}}(\boldsymbol{y}_t|\boldsymbol{x}_t^{(i)})$. However, $f_{\boldsymbol{\theta}}(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}^{(A_t^i)})$ does not take advantage of any information carried by the observations. With this choice of simple proposal distribution, the performance of SMC algorithm will be inefficient if the observations are informative. A more efficient importance proposal distribution is the "partial posterior" distribution (Pitt and Shephard, 1999; Carvalho et al., 2010b,a) (also known as "optimal proposal distribution" in literature), $q_{t,\boldsymbol{\theta}}(\boldsymbol{x}_t) = p_{\boldsymbol{\theta}}(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}^{(A_t^i)}, \boldsymbol{y}_t) = f_{\boldsymbol{\theta}}(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}^{(A_t^i)}) \cdot f_{\boldsymbol{\theta}}(\boldsymbol{y}_t|\boldsymbol{x}_t)/f_{\boldsymbol{\theta}}(\boldsymbol{y}_t|\boldsymbol{x}_{t-1}^{(A_t^i)})$, in

Algorithm 10 Standard Sequential Monte Carlo for a simple State Space Model

Input: data $\boldsymbol{y}_{1:T}$, parameter $\boldsymbol{\theta}$.
Output: $\{\boldsymbol{x}_{1:T}^{(i)}, W_T^{(i)}\}_{1\le i\le N}$.
for $i = 1$ **to** N **do**
    Draw $\boldsymbol{x}_1^{(i)} \sim q_{1,\boldsymbol{\theta}}(\cdot|\boldsymbol{y}_1)$.
    Update weights $W_1^{(i)} = w_1^{(i)} / \sum_{i=1}^N w_1^{(i)}$, where

$$w_1^{(i)} = \frac{f_{\boldsymbol{\theta}}(\boldsymbol{y}_1|\boldsymbol{x}_1^{(i)}) f_{\boldsymbol{\theta}}(\boldsymbol{x}_1^{(i)})}{q_{1,\boldsymbol{\theta}}(\boldsymbol{x}_1^{(i)}|\boldsymbol{y}_1)}. \tag{16}$$

for $t = 2$ **to** T **do**
    for $i = 1$ **to** N **do**
        Resample the ancestor index $A_t^i \sim Mult(\{W_{t-1,\boldsymbol{\theta}}^{(j)}\}_{1\le j\le N})$.
        Sample $\boldsymbol{x}_t^{(i)} \sim q_{t,\boldsymbol{\theta}}(\cdot|\boldsymbol{x}_{t-1}^{(A_t^i)}, \boldsymbol{y}_t)$.
        Update weights $W_t^{(i)} = w_t^{(i)} / \sum_{i=1}^N w_t^{(i)}$, where

$$w_t^{(i)} = \frac{f_{\boldsymbol{\theta}}(\boldsymbol{y}_t|x_t^{(i)}) f_{\boldsymbol{\theta}}(\boldsymbol{x}_t^{(i)}|\boldsymbol{x}_{t-1}^{(A_t^i)})}{q_{t,\boldsymbol{\theta}}(\boldsymbol{x}_t^{(i)}|\boldsymbol{x}_{t-1}^{(A_t^i)}, \boldsymbol{y}_t)}. \tag{17}$$

which the numerator is the same as the numerator in Eq.(16) and Eq.(17) and therefore it can be canceled. In this case, the unnormalized weight takes the form $w_t^{(i)} = f_{\boldsymbol{\theta}}(\boldsymbol{y}_t|\boldsymbol{x}_{t-1}^{(A_t^i)})$.

Now we extend the above standard SMC scheme to the MSSR_m model to conduct online inference. All the notations related to the MSSR_m model in this section are consistent with the notations used before. Recall that z_t is the cluster label (latent variable) in the model and $\boldsymbol{\theta}$ is a vector of unknown static parameters, and $\boldsymbol{b}_t = \{\boldsymbol{b}_{t,1}, \ldots, \boldsymbol{b}_{t,K}\}$, where $\boldsymbol{b}_{t,k}$ is the random effect coefficients of cluster k for $\boldsymbol{y}_t$. Let $\boldsymbol{x}_t = (z_t, \boldsymbol{b}_t)$. The hierarchical MMSR_m model specified in Section 6.3.1 is also a state space model, with notations listed in Table 6.1. We choose the "partial posterior" as proposals for the latent variable, i.e. $q_{t,\boldsymbol{\theta}}(\boldsymbol{x}_t) = p_{\boldsymbol{\theta}}(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \boldsymbol{y}_t) = p_{\boldsymbol{\theta}}(\boldsymbol{x}_t|\boldsymbol{y}_t) = p_{\boldsymbol{\theta}}(z_t, \boldsymbol{b}_t|\boldsymbol{y}_t) = p(z_t|\boldsymbol{y}_t, \boldsymbol{\theta}) p(\boldsymbol{b}_t|z_t, \boldsymbol{y}_t, \boldsymbol{\theta})$, which is a product of $p(z_t|\boldsymbol{\theta}, \boldsymbol{y}_t)$, partial posterior of z_t and $p(\boldsymbol{b}_t|z_t, \boldsymbol{y}_t, \boldsymbol{\theta})$, full conditional distribution of $\boldsymbol{b}_t$. Using $p(z_t|\boldsymbol{\theta}, \boldsymbol{y}_t) \propto f(z_t|\boldsymbol{\theta}) f(\boldsymbol{y}_t|z_t, \boldsymbol{\theta})$, $p(z_t|\boldsymbol{\theta}, \boldsymbol{y}_t)$ can be expressed as

$$p(z_t|\boldsymbol{\theta}, \boldsymbol{y}_t) \sim Mult(1; \tau_{t1}^*, \ldots, \tau_{tK}^*), \tag{18}$$

where $\tau_{tk}^* \propto \phi(\boldsymbol{y}_t; \boldsymbol{S}_t \boldsymbol{\beta}_{t,k}, \xi_{t,k}^2 \boldsymbol{S}_t \boldsymbol{S}_t^T + \sigma_{t,k}^2 \boldsymbol{I}_{m_t}) \pi_{t,k}$ and $\sum_{k=1}^K \tau_{tk}^* = 1$.

Our objective is to sequentially approximate $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:T}) \propto f_{\boldsymbol{\theta}}(\boldsymbol{y}_{1:T}) f(\boldsymbol{\theta})$, where $f(\boldsymbol{\theta})$ is the prior distribution of $\boldsymbol{\theta}$: $f(\boldsymbol{\theta}) = f(\boldsymbol{\pi}) \prod_{k=1}^K f(\boldsymbol{\beta}_k) f(\xi_k^2) f(\sigma_k^2)$. Direct estimation of the posterior distribution of $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:T})$ is complicated. In order to approximate $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:T})$, we combine the latent variable $\boldsymbol{x}_{1:T}$ with the model static parameters $\boldsymbol{\theta}$ to conduct model inference. We aim to estimate the posterior distribution $p(\boldsymbol{x}_{1:T}, \boldsymbol{\theta}|\boldsymbol{y}_{1:T}) = p(z_{1:T}, \boldsymbol{b}_{1:T}, \boldsymbol{\theta}|\boldsymbol{y}_{1:T})$ in the SMC framework.

Table 6.1: Notations of a general state space model and a hierarchical MMSR_m model.

Notations	Simple State Space Model	Hierarchical MSSR_m model
Observation	$\boldsymbol{y}_t$	$\boldsymbol{y}_t$
Latent variable	$\boldsymbol{x}_t$	$\boldsymbol{x}_t = (z_t, \boldsymbol{b}_t^\top)^\top$, $\boldsymbol{b}_t = \{\boldsymbol{b}_{t1}^\top, \ldots, \boldsymbol{b}_{tK}^\top\}^\top$
Static Parameter	$\boldsymbol{\theta}$ (known)	$\boldsymbol{\theta} = (\boldsymbol{\pi}^\top, \boldsymbol{\beta}^\top, (\boldsymbol{\sigma}^2)^\top, (\boldsymbol{\xi}^2)^\top)^\top$ (unknown)
Prior	$f_{\boldsymbol{\theta}}(\boldsymbol{x}_t \vert \boldsymbol{x}_{t-1})$	$f_{\boldsymbol{\theta}}(\boldsymbol{x}_t \vert \boldsymbol{x}_{t-1}) = f_{\boldsymbol{\theta}}(\boldsymbol{x}_t) = f(z_t \vert \boldsymbol{\pi}) \prod_{k=1}^K f(\boldsymbol{b}_{tk} \vert \boldsymbol{\xi}^2)$
Proposal	$p_{\boldsymbol{\theta}}(\boldsymbol{x}_t \vert \boldsymbol{x}_{t-1}, \boldsymbol{y}_t)$	$p_{\boldsymbol{\theta}}(\boldsymbol{x}_t \vert \boldsymbol{x}_{t-1}, \boldsymbol{y}_t) = p_{\boldsymbol{\theta}}(\boldsymbol{x}_t \vert \boldsymbol{y}_t) = p_{\boldsymbol{\theta}}(z_t, \boldsymbol{b}_t \vert \boldsymbol{y}_t)$ $= p(z_t \vert \boldsymbol{y}_t, \boldsymbol{\theta}) p(\boldsymbol{b}_t \vert z_t, \boldsymbol{y}_t, \boldsymbol{\theta})$ $= p(z_t \vert \boldsymbol{y}_t, \boldsymbol{\theta}) \prod_{k=1}^K p(\boldsymbol{b}_{tk} \vert z_t, \boldsymbol{y}_t, \boldsymbol{\theta})$
Weight w_t	$f_{\boldsymbol{\theta}}(\boldsymbol{y}_t \vert \boldsymbol{x}_{t-1})$	$f_{\boldsymbol{\theta}}(\boldsymbol{y}_t \vert \boldsymbol{x}_{t-1}) = f(\boldsymbol{y}_t \vert \boldsymbol{x}_{t-1}, \boldsymbol{\theta}) = f(\boldsymbol{y}_t \vert \boldsymbol{\theta})$

Conditional on ($\{\boldsymbol{y}_i^\top\}_{1:t}, z_{1:t}, \{\boldsymbol{b}_i^\top\}_{1:t}$), for $t = 1, \ldots, T$, sufficient statistics $\boldsymbol{s_t}$ for the parameters $\boldsymbol{\theta}$ in MSSR_m is given in **Proposition 1**.

Proposition 1 The sufficient statistics $\boldsymbol{s}_t$ for the parameters $\boldsymbol{\theta}$ in MSSR_m model given $(\boldsymbol{y}_{1:t}, z_{1:t}, \boldsymbol{b}_{1:t})$ for $t = 1, \ldots, T$ can be written as

$$\boldsymbol{s}_t = \left\{ \sum_{t'=1}^{t} \boldsymbol{b}_{t'k}^\top \boldsymbol{b}_{t'k}, \sum_{t'=1}^{t} \mathbf{1}_k(z_{t'}), \sum_{t'=1}^{t} \mathbf{1}_k(z_{t'}) \boldsymbol{S}_{t'}^\top \boldsymbol{y}_{t'}^*, \sum_{t'=1}^{t} \mathbf{1}_k(z_{t'}) \boldsymbol{y}_{t'}^{*\top} \boldsymbol{y}_{t'}^*, \sum_{t'=1}^{t} \mathbf{1}_k(z_{t'}) \boldsymbol{S}_{t'}^\top \boldsymbol{S}_{t'} \right\}_{k=1}^{K},$$

where $\boldsymbol{y}_{t'}^* = \boldsymbol{y}_{t'} - \boldsymbol{S}_{t'} \boldsymbol{b}_{t'k}$ for $t' = 1, \ldots, t$.

Since $\boldsymbol{s}_t$ is a function of $\boldsymbol{s}_{t-1}$ and $\boldsymbol{y}_t, z_t, \boldsymbol{b}_t$, we denote it as $\boldsymbol{s}_t = T(\boldsymbol{s}_{t-1}, \boldsymbol{y}_t, z_t, \boldsymbol{b}_t)$. Section 6.9.2 provides the proof of existence of sufficient statistics for the model static parameters.

Similar to the general particle learning (i.e. SMC) procedures of state space models in Carvalho et al. (2010a), when model static parameter $\boldsymbol{\theta}$ is unknown and its sufficient statistics exists, the online SMC learning of the hierarchical MSSRm model iterates between the following 4 steps at time t: *Resample* (ancestor index), *Propagate latent variable*, *Propagate sufficient statistics*, update model static parameters $\boldsymbol{\theta}$ using one *Gibbs Move*:

Step 1. Resample Sample ancestor index A_t^i, $i = 1, \ldots, N$, from a multinomial distribution with event probabilities $\{W_{t-1}^{(j)}\}_{1 \le j \le N}$, $W_{t-1}^j \propto f(\boldsymbol{y}_t \vert \boldsymbol{\theta}_{t-1}^{(j)})$ according to Eq.(7). We introduce another set of notations $\{\boldsymbol{s}_{t-1}^{(A_t^i)}, \boldsymbol{\theta}_{t-1}^{(A_t^i)}\}_{1 \le i \le N}$ to represent particles after resampling.

Step 2. Propagate latent variable Sample ${z_t}^{(i)}$ from $p({z_t}^{(i)} \vert \boldsymbol{\theta}_{t-1}^{(A_t^i)}, \boldsymbol{y}_t)$ according to Eq.(18). Then sample $\boldsymbol{b}_t^i$ according to Eq.(24) with $(\boldsymbol{\theta}_t^{(A_t^i)}, z_t^{(i)})$.

Step 3. Propagate sufficient statistics Update sufficient statistics $\boldsymbol{s}_t^{(i)}$ according to **Proposition 1**, by letting $\boldsymbol{s}_t^{(i)} = T(\boldsymbol{s}_{t-1}^{(A_t^i)}, \boldsymbol{y}_t, z_t^{(i)}, \boldsymbol{b}_t^{(i)})$

Step 4. Gibbs move Update $\boldsymbol{\theta}_t^{(i)}$ with one Gibbs move. Details are shown in Algorithm 11.

Algorithm 11 provides a detailed description for our proposed online SMC algorithm. Once a new observation, for example a new image, arrives, we iterate between the above "Resample-Propagate-Propagate" steps with one Gibbs Move to update the parameters. After obtaining the particles $\{z_{1:T}^{(i)}, \boldsymbol{b}_{1:T}^{(i)}, \boldsymbol{\theta}_T^{(i)}\}_{i=1}^N$, we get the approximated marginal posterior density $\hat{p}(\boldsymbol{\theta}|\boldsymbol{y}_{1:T}) = \frac{1}{N}\sum_{i=1}^N \mathbf{1}_{\boldsymbol{\theta}_T^{(i)}}(\boldsymbol{\theta})$ by dropping particles $z_{1:T}^{(i)}$ and $\boldsymbol{b}_{1:T}^{(i)}$, where $\mathbf{1}_{\boldsymbol{\theta}_T^{(i)}}(\boldsymbol{\theta}) = 1$ if $\boldsymbol{\theta} = \boldsymbol{\theta}_T^{(i)}$, otherwise $\mathbf{1}_{\boldsymbol{\theta}_T^{(i)}}(\boldsymbol{\theta}) = 0$.

As specified in Algorithm 11, we do not run MCMC within SMC to update the model static parameters $\boldsymbol{\theta}_t^{(i)}$ until $t = n.min$. The purpose is to make sure our algorithm is numerically well behaved as MCMC may be degenerate for models with a very small number of observations. In our numerical experiments we set $n.min$ to a small number (e.g. a value falls between 20 and 100), and we observe this is sufficient. As the size of first batch of input for the proposed online SMC algorithm reaches $n.min$, we start to update the static parameters when new observations arrive.

As argued by Fearnhead (2002) and Gilks and Berzuini (2001), including MCMC moves within SMC to mutate particles can alleviate the progressive degeneration. With the existence of model sufficient statistics, this online SMC learning algorithm will be computationally efficient. Instead of using all of the data in the MCMC move, only sufficient statistics are required to summarize the data.

Recall that K is the total number of clusters, and N^* is the total number of iterations in the MCMC algorithm, as described in Algorithm 9. The MCMC algorithm would take $O(tKN^*)$-time to update model parameters when a new observation $\boldsymbol{y}_t$ arrives, because we have to re-run the algorithm in order to use all data information $\boldsymbol{y}_{1:t}$. The cost of MCMC algorithm is a function of the total number of observations t, total number of MCMC iterations N^* and total number of clusters K. In contrast, the computation of the proposed online SMC algorithm (Algorithm 11) at time t takes $O(NK)$-time, which does not depend on the number of observations t with the adoption of sufficient statistics in the algorithm. The cost only depends on the total number of clusters K and total number of particles N. Consequently, the proposed online SMC algorithm is computationally more efficient than the MCMC algorithm, especially when t is large.

6.4 Model selection

Model selection is an important task in the Bayesian MSSR_m model framework as in many scenarios we are not able to know the optimum model. The goal is to compute the marginal likelihood $p(\boldsymbol{y}_{1:T})$. Numerous methods have been proposed to estimate the marginal likelihood (Friel and Pettitt, 2008; Gelman and Meng, 1998; Watanabe, 2010; Gelman et al., 2014). These algorithms require substantial additional effort in both computation and implementation to compute the marginal likelihood, which is not accessible to stream type data.

One advantage of the standard sequential Monte Carlo method is that it can provide an unbiased marginal likelihood estimator as a by-product of the algorithm. This marginal likelihood estimator admits a concise form, which is the product of average unnormalized weights. In the following proposition, we show that the marginal likelihood estimator provided by our proposed algorithm is unbiased, with specific conditions. The proof of this proposition is presented in Section 6.10.1.

Algorithm 11 Online learning for mixtures spatial spline regression model

Input: data $\boldsymbol{y}_{1:T}$, number of clusters K, initial value $\{\boldsymbol{\theta}_0\}$.
Output: $\{\boldsymbol{\theta}_{1:T}^{(i)\intercal}, \boldsymbol{s}_{1:T}^{(i)\intercal}, W_T^{(i)\intercal}\}_{1\leq i\leq N}^{\intercal}$.
if $t = 1$ **then**
    for $i = 1$ **to** N **do**
        Draw $z_1^{(i)} \sim p(\cdot|\boldsymbol{y}_1, \boldsymbol{\theta}_0)$ according to Eq.(18).
        for $k = 1$ **to** K **do**
            Draw $\boldsymbol{b}_{1k}^{(i)} \sim p(\cdot|\boldsymbol{y}_1, \boldsymbol{\theta}_0, z_1^{(i)})$ according to Eq.(24).
        Update the model sufficient statistics $\boldsymbol{s}_1^{(i)}$ as described in **Proposition 1**.
if $t > 1$ **then**
    for $i = 1$ **to** N **do**
        Compute weights $\{W_{t-1}^{(j)}\}_{j=1}^{N}$, $W_{t-1}^{(j)} \propto f(\boldsymbol{y}_t|\boldsymbol{\theta}_{t-1}^{(j)})$ according Eq.(7).
        Sample the ancestor index of particle i at time t, $A_t^i \sim Mult(\{W_{t-1}^{(j)}\}_{1\leq j\leq N})$.
        Sample $z_t^{(i)} \sim p_t(\cdot|\boldsymbol{y}_t, \boldsymbol{\theta}_{t-1}^{(A_t^i)})$ according to Eq.(18).
        for $k = 1$ **to** K **do**
            Sample $\boldsymbol{b}_{tk}^{(i)} \sim p(\cdot|\boldsymbol{y}_t, \boldsymbol{\theta}_{t-1}^{(A_t^i)}, z_t^{(i)})$ according to Eq.(24).
        Update the model sufficient statistics $\boldsymbol{s}_t^{(i)}$ conditional on $\boldsymbol{s}_{t-1}^{(A_t^i)}$ and $\boldsymbol{y}_t, \boldsymbol{b}_{tk}^{(i)}, z_t^{(i)}$ via **Proposition 1**.
        if $t > n.min$ **then**
            Sample $\boldsymbol{\pi}_t^{(i)} \sim p(\cdot|\boldsymbol{y}_{1:t}, z_{1:t}^{(i)}, \boldsymbol{\beta}_{t-1}^{(A_t^i)}, \boldsymbol{\sigma}^{2(A_t^i)}_{t-1}, \boldsymbol{\xi}^{2(A_t^i)}_{t-1}, \boldsymbol{b}_t^{(i)}, \boldsymbol{s}_t^{(i)})$ according to Eq.(22).
            for $k = 1$ **to** K **do**
                Sample $\boldsymbol{\beta}_{t,k}^{(i)} \sim p(\cdot|\boldsymbol{y}_{1:t}, z_{1:t}^{(i)}, \boldsymbol{\pi}_t^{(i)}, \sigma^{2(A_t^i)}_{t-1,k}, \xi^{2(A_t^i)}_{t-1,k}, \boldsymbol{b}_{t,k}^{(i)}, \boldsymbol{s}_t^i)$ according to Eq.(23).
                Sample $\sigma_{t,k}^{2\ (i)} \sim p(\cdot|\boldsymbol{y}_{1:t}, z_{1:t}^{(i)}, \boldsymbol{\pi}_t^{(i)}, \boldsymbol{\beta}_{t,k}^{(i)}, \xi^{2(A_t^i)}_{t-1,k}, \boldsymbol{b}_{t,k}^{(i)}, \boldsymbol{s}_t^i)$ according to Eq.(25).
                Sample $\xi_{t,k}^{2\ (i)} \sim p(\cdot|\boldsymbol{y}_{1:t}, z_{1:t}^{(i)}, \boldsymbol{\pi}_t^{(i)}, \boldsymbol{\beta}_{t,k}^{(i)}, \sigma^{2(i)}_{t,k}, \boldsymbol{b}_{t,k}^{(i)}, \boldsymbol{s}_t^i)$ according to Eq.(26).
        else
            Set $\boldsymbol{\theta}_t^{(i)} = \boldsymbol{\theta}_{t-1}^{(A_t^i)}$.

Proposition 2. If we update $\boldsymbol{\theta}_t^{(i)}$ with multiple Gibbs moves until convergence achieved, such that $\boldsymbol{\theta}_t^{(i)} \sim p(\boldsymbol{\theta}|\boldsymbol{x}_{1:t}, \boldsymbol{y}_{1:t})$. The product of average unnormalized weights $\prod_{t=1}^{T} \frac{1}{N} \sum_{i=1}^{N} p(\boldsymbol{y}_t|\boldsymbol{\theta}_{t-1}^{(i)})$ is an unbiased estimator of the marginal likelihood $p(\boldsymbol{y}_{1:T})$,

$$E\left(\prod_{t=1}^{T} \frac{1}{N} \sum_{i=1}^{N} p(\boldsymbol{y}_t|\boldsymbol{\theta}_{t-1}^{(i)})\right) = p(\boldsymbol{y}_{1:T}). \tag{19}$$

Even though practically we only use one Gibbs move to update $\boldsymbol{\theta}_t^{(i)}$, the marginal likelihood estimator is generally not unbiased as the condition displayed in Proposition 2 is broken. In our real data analysis, we compare our model selection criteria with Watanabe-Akaike information criteria (WAIC) (Watanabe, 2010; Gelman et al., 2014) and demonstrate it works in practice.

6.5 Simulation study

We evaluate the proposed algorithms using simulation studies. Assume each surface cluster k, $k = 1, \cdots, 6$, its common features $f_k(\eta_1, \eta_2)$ over a rectangular domain $[-1, 1] \times [-1, 1]$ admit the following functional forms

$$\begin{aligned}
f_1(\eta_1, \eta_2) &= \frac{\eta_1^3 + \eta_2^3 + 3}{\sqrt{1 + \eta_1^2 + \eta_2^2}}, & f_2(\eta_1, \eta_2) &= \frac{\eta_1^2 + \eta_2^2 + 1}{\sqrt{4 + \eta_1^2 + \eta_2/4}}, \\
f_3(\eta_1, \eta_2) &= 1 - \sin(\eta_1^2 + 1) + \frac{\cos(1 + \eta_2^2)}{2}, & f_4(\eta_1, \eta_2) &= \sin(\eta_1 \eta_2), \\
f_5(\eta_1, \eta_2) &= \cos(\eta_1 + \eta_2) + \sin(\eta_1^2) + \cos(\eta_2^2), & f_6(\eta_1, \eta_2) &= \eta_1 + \eta_2.
\end{aligned} \tag{20}$$

A surface $\boldsymbol{y}_t$ belonging to cluster k is simulated through

$$\boldsymbol{y}_t = (f_k(\boldsymbol{\eta}_{t,1}), \ldots, f_k(\boldsymbol{\eta}_{t,m_t}))^\intercal + \boldsymbol{S}_t \boldsymbol{b}_{tk} + \boldsymbol{e}_{tk},$$

where $(f_k(\boldsymbol{\eta}_{t,1}), \ldots, f_k(\boldsymbol{\eta}_{t,m_t}))^\intercal$ is the common feature for surfaces in cluster k - the mean surface for cluster k, and $\boldsymbol{\eta}_{t,1}, \ldots, \boldsymbol{\eta}_{t,m_t}$ are distributed uniformly over the domain, each surface contains $m_t = 12 \times 12$ points. $\boldsymbol{S}_t$ is the basis covariates function defined in Eq.(3). The observed surface is then $\boldsymbol{y}_t = (y_{t,1}, \ldots, y_{t,m_t})$. The random effects and the errors of the model are simulated from $\boldsymbol{b}_{tk} \sim MVN(0, \xi_k^2 \boldsymbol{I}_d)$ and $e_{tk} \sim MVN(0, \sigma_k^2 \boldsymbol{I}_{m_t})$. The number of nodal basis functions we used in the simulation study is $d = 6 \times 6$. The 6 panels in the first row of Figure 6.3 display the mean surfaces simulated by functions $(f_k(\boldsymbol{\eta}_{t,1}), \ldots, f_k(\boldsymbol{\eta}_{1,m_t}))^\intercal$ in Eq.(20).

6.5.1 Surface fitting

In this section, we use a relatively simple experiment to illustrate the MSSR_m model fitting using the MCMC algorithm. We use Eq.(20) to generate the common feature for surfaces in each cluster. A data set comprising 600 images in total (100 for each cluster) is simulated. We set $\sigma_k^2 = 0.1$ and

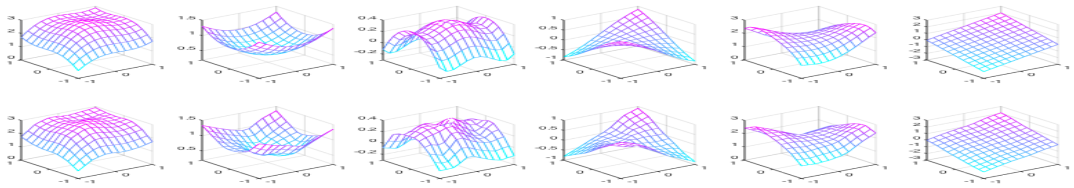


Figure 6.3: True surfaces versus fitted surfaces: the top row displays the true surfaces of simulated by functions $f_1, \ldots, f_6$ uniformly over the rectangular domain $[-1, 1] \times [-1, 1]$, and the bottom row is the corresponding MCMC fitted mean surfaces with $d = 6 \times 6$.

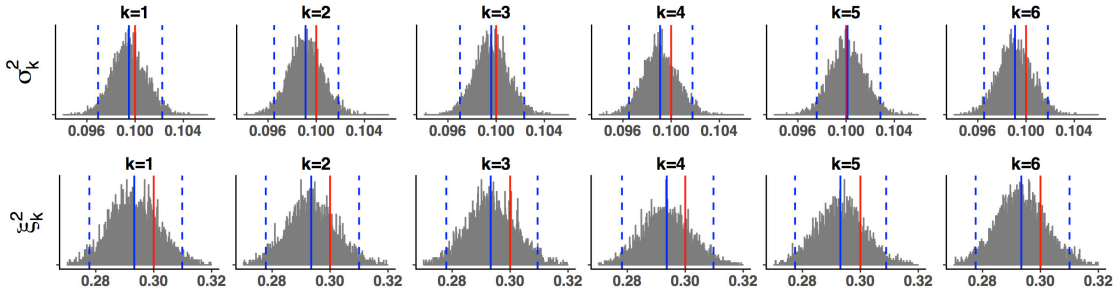


Figure 6.4: MCMC estimates of σ_k^2 and ξ_k^2, $(k = 1, \ldots, 6)$. The dashed blue lines represent 95% equal tailed credible interval, the solid blue lines indicate the mean value of the estimates and the solid red lines imply the true values of the estimates.

$\xi_k^2 = 0.3$ for $k = 1, \ldots, 6$. The total number of MCMC iterations is set to $5,000$, and we discard the first $1,000$ iterations as burn-in. Figure 6.3 displays a comparison between the true surfaces we simulated (upper panels) and the estimated mean surfaces provided by running the MCMC algorithm (bottom panels). As indicated in Figure 6.3, the fitted mean surfaces, $\boldsymbol{S}_t\boldsymbol{\beta}_k$, recover the common features of the true surfaces. In Figure 6.4, we show the histogram of the posterior distributions of σ_k^2 and ξ_k^2 obtained from running MCMC. The red line represents the true value of parameter and the blue line represents the mean value of estimated posterior. The dashed blue lines are the 2.5% and 97.5% quantiles of the posterior distribution. Figure 6.4 indicates that all of the true values of the parameters can be covered by the associated 95% credible intervals. The estimated posterior means of both σ_k^2 and ξ_k^2 are very close to the corresponding true values.

6.5.2 Comparison of online SMC with MCMC

In this section, we simulate two data sets to evaluate the performance of online SMC and MCMC in terms of sequential image clustering. In both data sets, we simulate images in an online fashion where the observation arrives one by one, and the data keep accumulating until the total number of observations reaches $T = 10,000$. The number of particles we set in the online SMC algorithm is $N = 1000$. The number of MCMC iterations is set to 5000 to guarantee the convergence of the algorithm and we burn-in the first 20% of the chain.

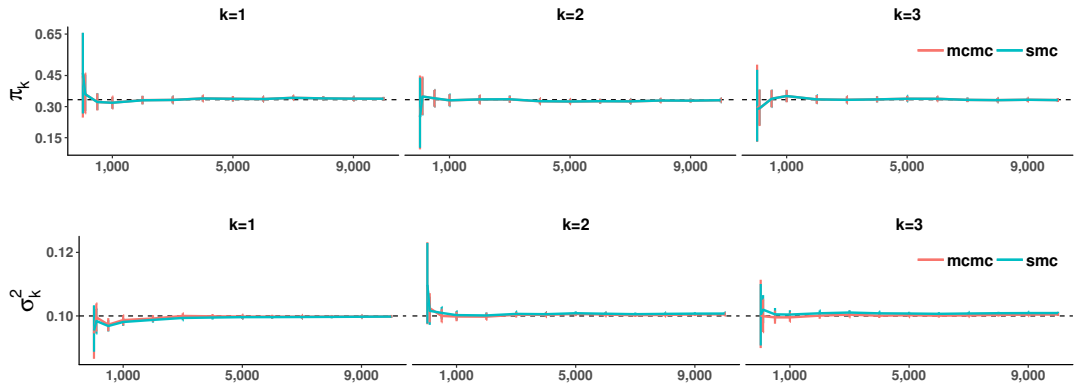


Figure 6.5: Estimated parameters π_k, σ_k^2 with 95% equal tailed credible interval of online SMC algorithm versus MCMC algorithm when $K = 3$ and the number of observations increases from $t = 20$ to $10,000$.

In the first experiment, we simulate a data set with a relatively small number of clusters, $K = 3$. The common features of images are generated by f_1, f_2 and f_3 of Eq.(20). The allocation probability is set to $\pi = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$. We assume the variance of random effects and error are $\sigma_k^2 = 0.1$ and $\xi_k^2 = 0.3$ for $k = 1, 2, 3$. In online SMC algorithm, we do not update the model parameters until we have received a small batch of images, here we set $n.min = 20$, a relative small number. The initial value of parameters are obtained by running MCMC on the first 20 images. After that, we update the model parameters once when we obtain a new image. The MCMC algorithm serves as a baseline for comparison, and is updated with $T = n_{min}, 100, 500, 1000, 2000, \ldots, 10000$ images. Figure 6.5 displays the comparison of π_k, σ_k^2 as a function of t provided by online SMC and MCMC. The horizontal lines represent the posterior mean provided by the different methods. The vertical lines represent the 95% credible intervals for online SMC and MCMC. We only display the credible intervals for online SMC every 500 images for the purpose of making a comparison with the MCMC algorithm. As indicated in Figure 6.5, both algorithms achieve similar performance in terms of inference on the parameters π_k and σ_k^2, for $k = 1, 2, \ldots, K$. With the increment of observations, the posterior mean of both MCMC and online SMC gets closer to the true value, and the credible interval tends to get narrower, as expected.

In our second experiment, we simulate from the MSSR_m model with more latent states by setting $K = 6$. The common features are simulated from the six functions listed in Eq.(20). The allocation probability π_k, random effects ξ_k^2 and variance of the error σ_k^2 are set to $\pi = (\frac{1}{6}, \frac{1}{6}, \frac{1}{6}, \frac{1}{6}, \frac{1}{6}, \frac{1}{6})$, $\sigma_k^2 = 0.1$ and $\xi_k^2 = 0.3$ for $k = 1, \ldots, 6$. Our online SMC algorithm does not update the parameters until we have $n.min = 40$ observations. Figure 6.6 displays the posterior mean and 95% credible interval for π_k and σ_k^2 as a function of time t. Both online SMC and MCMC can achieve good performance in terms of estimating π_k and σ_k^2. With the increment of the number of images, the posterior means of π_k and σ_k^2 tend to converge to the corresponding true value.

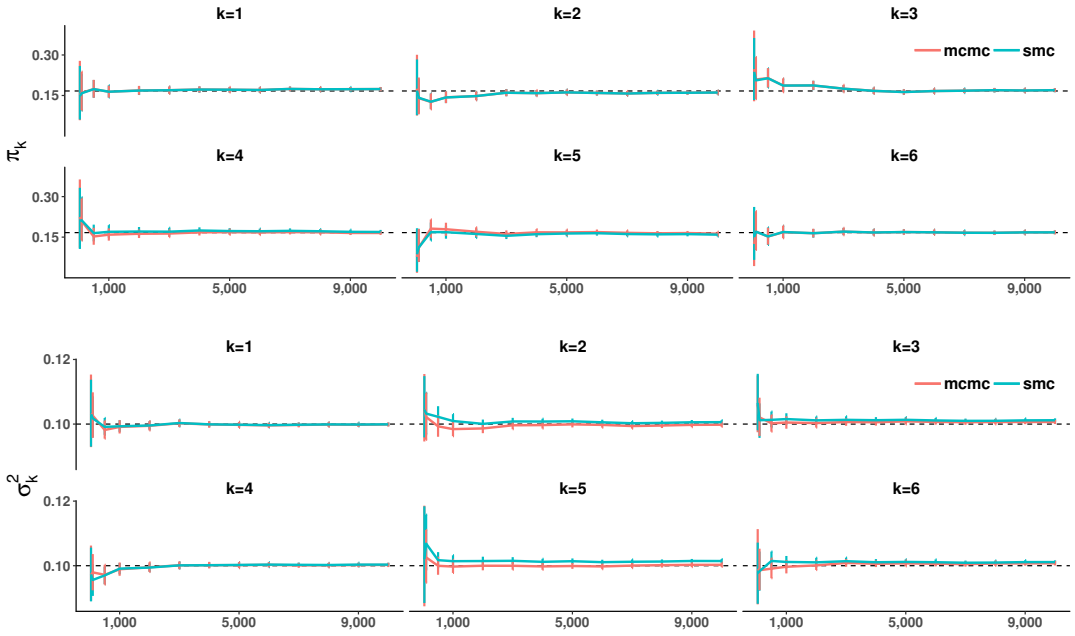


Figure 6.6: Estimated parameters π_k, σ_k^2 with 95% equal tailed credible interval of online SMC algorithm versus MCMC algorithm when $K = 6$ and the number of observations increases from $t = 40$ to $10,000$.

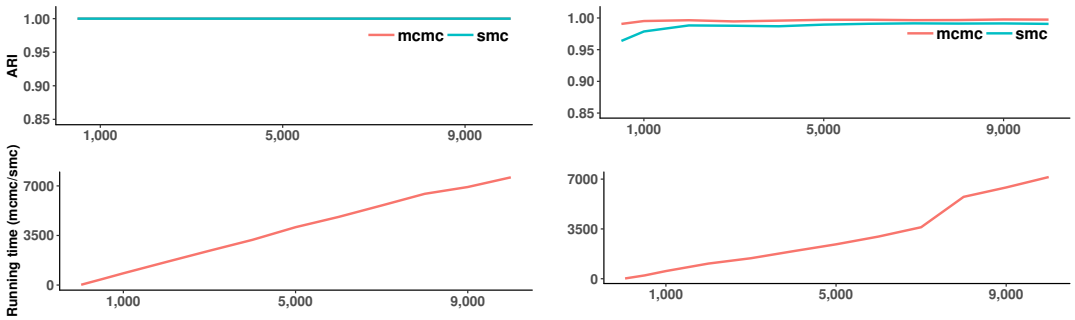


Figure 6.7: Estimated ARI of online SMC algorithm versus MCMC algorithm when $K = 3$ (top left panel) and $K = 6$ (top right panel) and running time ratio of MCMC algorithm over online SMC algorithm when $K = 3$ and (bottom right panel) when $K = 6$ (bottom right panel).

In both experiments, we also apply the Adjusted Rand Index (ARI) (Hubert and Arabie, 1985) to measure the performance of clustering. As indicated in Figure.6.7, both the online SMC algorithm and the MCMC algorithm can achieve quite a good performance in terms of ARI. However, the running time ratio of MCMC over online SMC increases almost linearly as time evolves, which indicates that online SMC is more scalable to stream type data. Both experiments indicate that the online SMC algorithm is at least several orders of magnitude faster than the MCMC algorithm in terms of computational cost when t is large ($> 2,000$). Instead of a re-run of the MCMC method, our proposed online SMC algorithm only needs to update the sufficient statistics in **Proposition 1** in order to achieve model updating, which leads to efficient computation relative to that of MCMC for sequential image data. With the increment of observations, both algorithms can achieve good performance in terms of parameter estimation and image clustering.

6.6 Real data analysis

In this section, we apply our proposed online SMC algorithm to two real data sets: one handwritten image data and one brain imaging dataset where brain activity is recorded using MEG. In real applications, one challenge in clustering is the lack of information for the number of clusters. This generally requires some model selection technique to choose the optimum model. As we have alluded in the previous section, the computation of the marginal likelihood $p(y_{1:T})$ in Bayesian statistics is challenge, but online SMC can provide an unbiased estimator of the marginal likelihood as a by-product of the algorithm, which is a big advantage of SMC over MCMC. In this real data analysis, we investigate the performance of marginal likelihood of online SMC, and treat Watanabe-Akaike information criteria (WAIC) (Watanabe, 2010; Gelman et al., 2014) as a baseline for comparison. The implementation of WAIC for our model can be found in Section 6.12.1.

6.6.1 Handwritten number images

The first real application we consider is the analysis of handwritten number images. We apply our proposed online SMC algorithm to a subset of the ZIP code data set used in Friedman et al. (2009). Every image is of one Hindu-Arabic handwritten number and consists of 16×16 grid of pixels, i.e. each image contains 256 observations and we use $5,000$ images in total. The images distribution is shown in Table 6.2

Table 6.2: Distribution of Hindu-Arabic handwritten numbers in the sample.

Number	0	1	2	3	4	5	6	7	8	9	Total
Frequency	904	723	500	385	413	344	443	419	408	461	5000

We set $K = 8, 10, 12$, and for each K, let $d = 6 \times 6,\ 8 \times 8$ separately. Hence, we have 6 MSSR_{m} models in total. We mimic a scenario where the images arrive one by one. For the online SMC algorithm, we do not update the model parameters until we have $n.min = 100$ images. After $t = 100$,

we update the parameters once one image arrives. The number of particles we use is $N = 1000$. For each pair of K and d, we try several sets of initial values.

The WAIC and $\log(p(\boldsymbol{y}_{1:T}))$ provided by online SMC is shown in Table 6.3. It shows model with a larger number of basis functions has higher marginalized likelihood and smaller WAIC value, which indicates a larger number of basis function is more preferable. And the performance of the MSSR_m model gets better as K increases as shown in Table 6.3. Both WAIC and $\log(p(\boldsymbol{y}_{1:T}))$ indicate the MSSR_m model with $K = 12$ and $d = 8 \times 8$ is the optimal model for those considered. However, WAIC is computationally much more expensive, as it takes an extra $O(TN)$-time to compute. We display the common features of the image digits provided by MSSR_m model with $K = 12$ in Figure 6.8. The MSSR_m model is able to recover all the common features of the 10 digits as well as multiple sub-groups for 0 (the $2^{nd}, 7^{th}$ clusters), 2 (the $4^{th}, 9^{th}, 11^{th}$ clusters), 5 ($5^{th}, 10^{th}$ clusters). The 1^{st} cluster indicates that handwritten digit 9 shares some common features with handwritten digit 7, the 6^{th} cluster indicates that handwritten digit 8 shares some common features with 3 and the 9^{th} indicates that some of the handwritten digit 8 share same common features as some handwritten digits 2. The common features of models with $K = 8$ and $K = 10$ are displayed in Section 6.11.1.

Table 6.3: Comparison of WAIC and logarithm of marginal likelihood $\log(p(\boldsymbol{y}_{1:T}))$ for MSSR_m model with $K = 8, 10, 12$ and $d = 6 \times 6, 8 \times 8$.

Criteria	d	No. of Clusters		
		K=8	K=10	K=12
$\log(p(\boldsymbol{y}_{1:T}))$	6×6	-1452774	-1443364	-1442439
	8×8	-1294886	-1277154	-1274572
WAIC	6×6	2901789	2883780	2881520
	8×8	2582337	2548418	2541894

6.6.2 Brain images

In this subsection we apply the online SMC algorithm to a subset of images collecting during a neuroimaging study examining the neural response to different natural stimuli. The data are collected using 204 MEG sensors located around the scalp where each sensor measures a time series representing the magnetic field at its location. The magnetic field at a given location is an indirect measurement of the electric neural activity within the brainNathoo et al. (2013, 2014) . The sensor data at a given time point are projected onto a 2D grid and represented through a 2D image and each recording is made at 200Hz for 1 second resulting in 200 images made in each recording. The study involves over 700 such recordings, where each is associated with one of the five visual stimuli (1. Artificial: screen savers showing animated shapes or text; 2. Nature: clips from nature documentaries, showing natural scenery like mountains or oceans; 3. Football: clips taken from (European) football matches of Spanish La Liga; 4. Mr. Bean: clips from the episode Mind the Baby, Mr. Bean of the Mr. Bean television series; 5. Chaplin: clips from the Modern Times feature film, starring Charlie Chaplin).

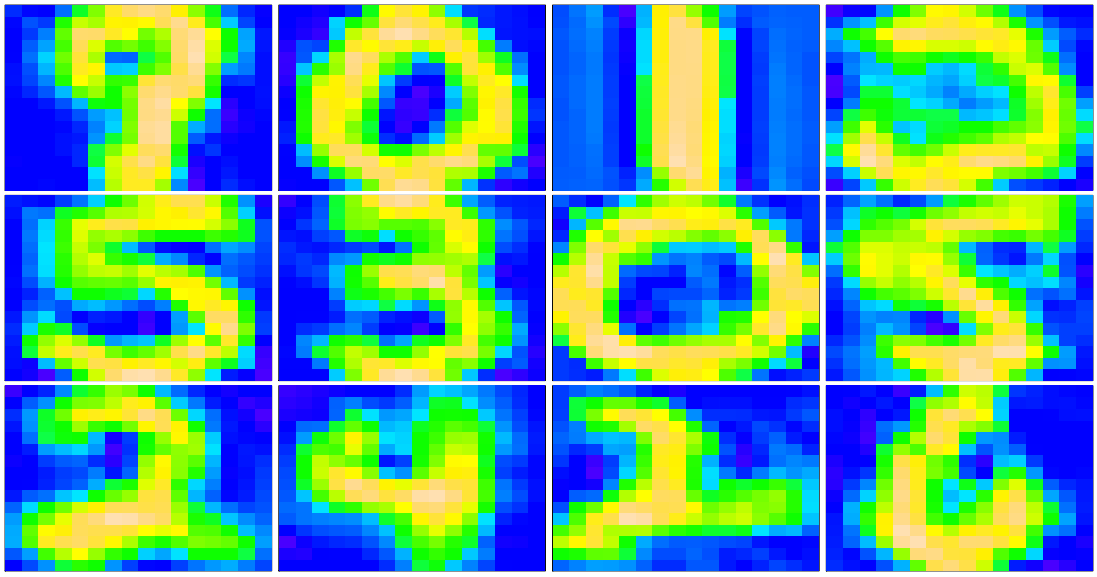


Figure 6.8: Online SMC estimated common features from MSSR_{m} model of handwritten number images by cluster when $K = 12,\ d = 8 \times 8$. Images are labeled as the 1^{st} cluster to the 12^{th} cluster in left-to-right, top-to-bottom order.

The neuroimaging data set contains a time series of 200 images for each of 727 recordings. Each time series (sample) is associated with one of the aforementioned five stimuli, that is, what the subject was watching when the time series of images was recorded. Figure 6.9 shows 2D images of 3 randomly drawn recordings against the same stimulus at the beginning ($t = 1, 2, 3$), middle ($t = 101, 102, 103$) and end ($t = 198, 199, 200$) of the 1 second recording period. Figure 6.9 demonstrates the variability in brain activity across different recordings for the same stimulus and thus shows that the type of stimulus associated with a given recording does not clealy distinguish the images from each other, at least by eye. We apply our classifier to decode the images in order to reveal the common stimuli via clustering. We focus our analysis of this application on model selection.

In this application, we use $5,000$ images, the corresponding stimuli associated with each image is distributed as shown in Table 6.4. Each image originally consists of 512×512 pixels, the high dimensionality of these images makes the implementation challenging. Hence, we begin by removing redundant zeros around the boundaries for each image and compressing the images to a more coarse level, 14×18 pixels to save computational cost.

Table 6.4: Distribution of stimuli in the sample data.

Stimulus	1	2	3	4	5
Frequency	1183	1003	706	887	1221

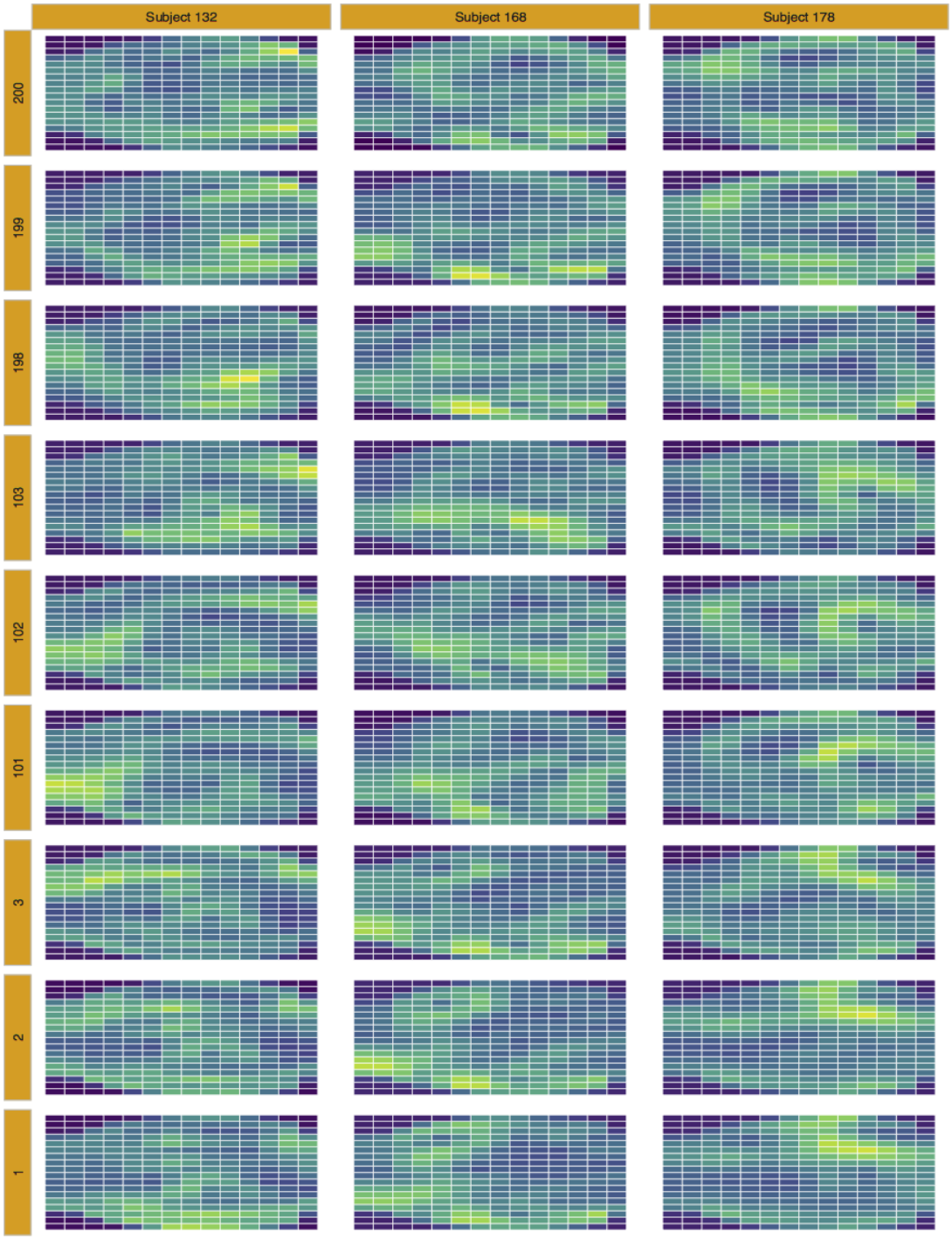


Figure 6.9: Brain Images of 3 different recordings against same stimulus at the beginning ($t = 1, 2, 3$), middle ($t = 101, 102, 103$) and end ($t = 198, 199, 200$) of the experimental period.

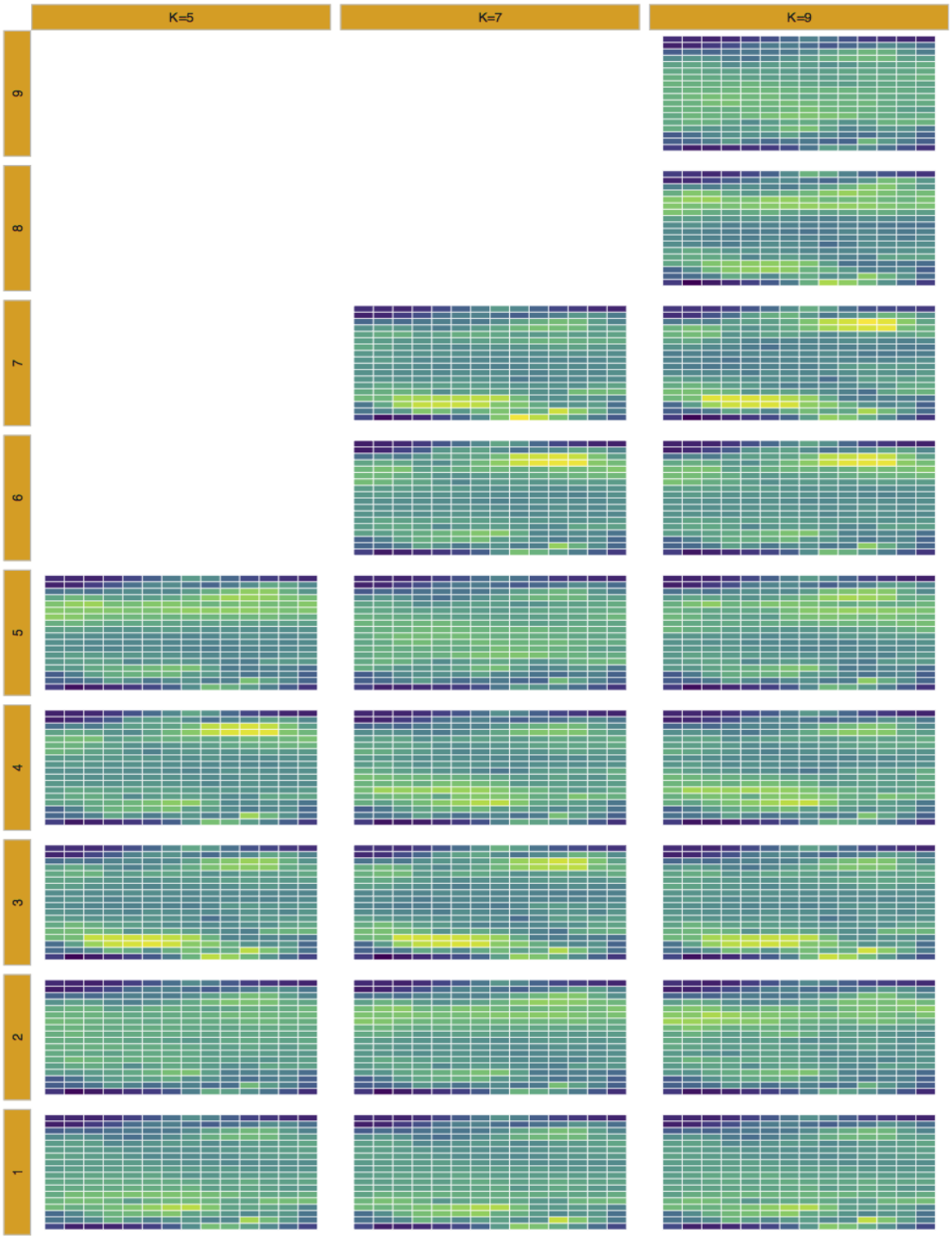


Figure 6.10: Online SMC estimated common features from MSSR_{m} model for Brain Images when $K = 5, 7, 9$ and $d = 8 \times 8$.

We set $K = 5, 7, 9$, and $d = 6 \times 6$, 8×8 for each K. We assume that the images are received one by one or batch by batch. We do not update the model parameters until we have $n.min = 100$ images. After $t = 100$, we update the parameters once one image arrives. The number of particles

we use is $N = 1000$. For each pair of K and d, we try several sets of initial values. The WAIC and $\log(p(\boldsymbol{y}_{1:T}))$ provided by online SMC is shown in Table 6.5. Both $\log(p(\boldsymbol{y}_{1:T}))$ and WAIC indicate the model performs better when more basis functions are used. And the performance of MSSR_m model gets better as K increases as shown in Table 6.5. Both WAIC and online SMC indicate the MSSR_m model with $K = 9$ and $d = 8 \times 8$ is the optimum.

Table 6.5: WAIC and $\log(p(\boldsymbol{y}_{1:T}))$ for MSSR_m models with $K = 5, 7, 9$ and $d = 6 \times 6, 8 \times 8$.

Criteria	d	No. of Clusters		
		$K = 5$	$K = 7$	$K = 9$
$\log(p(\boldsymbol{y}_{1:T}))$	6×6	-1549603	-1545583	-1543856
	8×8	-1466522	-1460035	-1456731
WAIC	6×6	3098131	3089754	3086072
	8×8	2931129	2917420	2909831

As indicated in Figure 6.10, compared to $K = 5$, when $K = 7, 9$, the MSSR_m model is able to capture more common features of several subtle subgroups. As we mentioned in Section 6.3.3, the computational cost of updating parameters in the online SMC algorithm is $O(NK)$. Thus, the larger is K, the higher is the computational cost. To balance the computational cost and the performance of estimation, in this application of brain images we suggest to take 7 as the appropriate K for the MSSR_m model, since its estimates result in more subtle discrimination than that of when $K = 5$ and its computational cost is lower than that of when $K = 9$.

6.7 Discussion

In this chapter, we derive an MCMC algorithm under the Bayesian framework as an alternative approach to do model inference for the MSSR_m models. Moreover, we propose an online SMC algorithm to deal with the stream type of image data efficiently via adoption of sufficient statistics and augment variables. When new data arrive, our proposed online SMC algorithm achieves parameter updating in a constant time, which is more adaptable to large sequential image data. In contrast, the required computation time for the MCMC algorithm is a linear function of the total number of observations at time t since it always has to be re-run when new data arrive. Our simulation studies demonstrated that the proposed online SMC algorithm is more efficient than the MCMC algorithm in terms of computing time, while both algorithms can achieve good performance from the perspective of model inference.

Model selection is an important but challenging task in Bayesian statistics. We show that the marginal likelihood estimator provided by our proposed algorithm is unbiased, and it serves as a by-product of the algorithm. We compare this estimator with existing model selection criteria, WAIC, and showed that the same models are selected via $\log(p(\boldsymbol{y}_{1:T}))$ and WAIC. WAIC requires extra cost to compute and the computational complexity increases linearly with artificial time T.

In the posterior distribution of MSSR$_m$ models, the likelihood function is identical for all $K!$ permutation of labels. This may induce the label switching and complicate the inference, which makes it impossible to justify the convergence of MCMC (Stephens, 2000). Many approaches have been developed to address the label switching issue (Stephens, 2000; Grün and Leisch, 2009; Geweke, 2007; Diebolt and Robert, 1994; Puolamäki and Kaski, 2009) of mixture models. Our developed online algorithm has much more common with importance sampling (IS) than with MCMC and the validation of the proposed algorithm is based on IS principles, hence the approximation to the target is valid at each iteration and does not require convergence (Marin et al., 2005). Our numerical experiments demonstrate our proposed online algorithm provides an appropriate discrete approximation to the distributions of interest.

6.8 Appendix A

6.8.1 List of notations

Table 6.6: List of notations.

Notation	Description
$^\intercal$	transpose symbol of a vector or matrix.
T	total number of observations.
N^*	total number of Gibbs sampling iterations.
N	total number of particles in online SMC algorithm.
K	number of clusters.
d	number of Nodal basis functions.
k	index for cluster k, for $k = 1, \ldots, K$.
t	time index, takes value from $1, \ldots, T$.
m_t	length of observation $\boldsymbol{y}_t$, for $t = 1, \ldots, T$.
$\boldsymbol{y}_t$	a $m_t \times 1$ vector, observation at time t, for $t = 1, \ldots, T$.
$\boldsymbol{\eta}$	$\boldsymbol{\eta} = (\eta_1, \eta_2)$, arbitrary coordinates.
$\boldsymbol{\eta}_{t,1}, \ldots, \boldsymbol{\eta}_{t,m_t}$	coordinates for $\boldsymbol{y}_t$, for $t = 1, \ldots, T$.
$\boldsymbol{c}$	$\boldsymbol{c} = (c_1, c_2)$, center parameter for a Nodal basis function.
$\boldsymbol{c}_l$	$\boldsymbol{c}_l = (c_{l1}, c_{l2})$, center parameter for l^{th} Nodal basis function, $l = 1, \ldots, d$.
$\boldsymbol{\delta}$	$\boldsymbol{\delta} = (\delta_1, \delta_2)$, shape parameter for a Nodal basis function.
$s(\cdot)$	Nodal basis function.
$\boldsymbol{S}_t$	a $m_t \times d$ matrix, spatial coordinates matrix of observation, $\boldsymbol{y}_t$, for $t = 1, \ldots, T$.
$\boldsymbol{\beta}_k$	a $d \times 1$ vector, fixed effects for cluster k, for $k = 1, \ldots, K$.
$\boldsymbol{b}_{tk}$	a $d \times 1$ vector, random effects of cluster k for $\boldsymbol{y}_t$, for $k = 1, \ldots, K$, $t = 1, \ldots, T$.
$\boldsymbol{e}_{tk}$	a $d \times 1$ vector, random error of cluster k for $\boldsymbol{y}_t$, for $k = 1, \ldots, K$, $t = 1, \ldots, T$.
$\boldsymbol{\sigma}_k^2, \boldsymbol{\xi}_k^2$	variance parameter for cluster k, $k = 1, \ldots, K$.
$\boldsymbol{\pi}$	$\boldsymbol{\pi} = (\pi_1, \ldots, \pi_K)$, cluster allocation probability for observations.
z_t	cluster label of $\boldsymbol{y}_t$ for $t = 1, \ldots, T$.
$\boldsymbol{Y}, \boldsymbol{Z}, \boldsymbol{b}, \boldsymbol{b}_t$	$\boldsymbol{Y} = \{\boldsymbol{y}_t\}_{t=1}^T$, $\boldsymbol{Z} = \{z_t\}_{t=1}^T$, $\boldsymbol{b} = \{\boldsymbol{b}_t\}_{t=1}^T$, $\boldsymbol{b}_t = \{\boldsymbol{b}_{tk}\}_{k=1}^K$ for $k = 1, \ldots, K$.
$\boldsymbol{\theta}$	$\boldsymbol{\theta} = \{\boldsymbol{\pi}, \boldsymbol{\beta}, \boldsymbol{\sigma}^2, \boldsymbol{\xi}^2\}$.
$MVN, Dir, Mult, IG$	abbreviations for multivariate normal distribution, Dirichlet distribution, multinomial distribution and inverse-gamma distribution, respectively.
$\phi(\cdot)$	density function of a multivariate normal distribution,
$f(\cdot), f_{\boldsymbol{\theta}}(\cdot), p(\cdot), p_{\boldsymbol{\theta}}(\cdot)$	density functions, usually $f(\cdot)$(or $f_{\boldsymbol{\theta}}(\cdot)$) for a prior, $p(\cdot)$(or $p_{\boldsymbol{\theta}}(\cdot)$) for a posterior, and $p(z_t = k)$ refers to probability of $z_t = k$ for $k = 1, \ldots, K$, $t = 1, \ldots, T$.
a_0, b_0, g_0, h_0	hyper parameters.
$\{\alpha_k\}_{k=1}^K, \boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0$	hyper parameters.
$\mathbf{1}(\cdot)$	indicator function.
$\boldsymbol{b}_{1:t}, z_{1:t}, \boldsymbol{x}_{1:t}, \boldsymbol{y}_{1:t}$	abbreviations of $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_t$,$z_1, \ldots, z_t$, $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_t$, $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_t$, for $t = 1, \ldots, T$.
$w_t^{(i)}, W_t^{(i)}$	unnormalized and normalized weight, respectively, for $i = 1, \ldots, N$, $t = 1, \ldots, T$.
$A_t^{(i)}$	ancestor index for particle i at time t for $i = 1, \ldots, N$, $t = 1, \ldots, T$.
$q_{t,\boldsymbol{\theta}}(\cdot)$	proposal distribution for $\boldsymbol{x}_t$ for $t = 1, \ldots, T$.
$\boldsymbol{s_t}$	sufficient statistics for the MSSR_m model given ($\boldsymbol{y}_{1:t}$, $z_{1:t}$, $\boldsymbol{b}_{1:t}$) for $t = 1, \ldots, T$.
$T(\cdot)$	function of ($\boldsymbol{s_{t-1}}, \boldsymbol{y}_t, z_t, \boldsymbol{b}_t$), and takes $\boldsymbol{s_t}$ as return, for $t = 2, \ldots, T$.
$f_k(\cdot)$	function used to simulate surfaces from cluster k, $k = 1, \ldots, K$.

6.9 Appendix B

6.9.1 Derivations for the Gibbs Sampling Algorithm.

In this section we derived the full conditional distributions for z_t, $\boldsymbol{\pi}$, $\boldsymbol{\beta}_k$, $\boldsymbol{b}_{tk}$, $\boldsymbol{\sigma}_k^2$ and $\boldsymbol{\xi}_k^2$. Given the hierarchical priors of the Bayesian mixture of spatial spline regression with mixed effects model described in section 6.3.1, the full joint posterior distribution of $\boldsymbol{\pi}, \boldsymbol{\beta}, \boldsymbol{b}, \boldsymbol{\sigma}^2, \boldsymbol{\xi}^2, \boldsymbol{Z}$ can be expressed up to a marginal likelihood as

$$
\begin{aligned}
&p(\boldsymbol{\pi}, \boldsymbol{\beta}, \boldsymbol{b}, \boldsymbol{\sigma}^2, \boldsymbol{\xi}^2, \boldsymbol{Z}|\boldsymbol{Y}) \propto f(\boldsymbol{\pi}) f(\boldsymbol{\beta}^2) f(\boldsymbol{\xi}^2) f(\boldsymbol{\sigma}^2) f(\boldsymbol{Y}, \boldsymbol{Z}|\boldsymbol{\beta}, \boldsymbol{b}, \boldsymbol{\xi}^2, \boldsymbol{\sigma}^2) \\
&\propto f(\boldsymbol{\pi}) \times \prod_{k=1}^{K} f(\boldsymbol{\beta}_k) \times \prod_{k=1}^{K} f(\xi_k^2) \times \prod_{t=1}^{T} \prod_{k=1}^{K} f(\boldsymbol{b}_{tk}|\xi_k^2) \times \prod_{k=1}^{K} f(\sigma_k^2) \\
&\quad \times \prod_{t=1}^{T} \prod_{k=1}^{K} \left\{ \phi(\boldsymbol{y}_t|\boldsymbol{S}_t\boldsymbol{\beta}_k + \boldsymbol{S}_t\boldsymbol{b}_{tk}, \sigma_k^2 \boldsymbol{I}_{m_t}) p(z_t = k) \right\}^{\mathbf{1}_k(z_t)} \\
&\propto f(\boldsymbol{\pi}) \times \prod_{k=1}^{K} f(\boldsymbol{\beta}_k) f(\xi_k^2) f(\sigma_k^2) \times \prod_{t=1}^{T} \prod_{k=1}^{K} f(\boldsymbol{b}_{tk}|\xi_k^2) \\
&\quad \times \prod_{t=1}^{T} \prod_{k=1}^{K} \left\{ \phi(\boldsymbol{y}_t|\boldsymbol{S}_t\boldsymbol{\beta}_k + \boldsymbol{S}_t\boldsymbol{b}_{tk}, \sigma_k^2 \boldsymbol{I}_{m_t}) p(z_t = k) \right\}^{\mathbf{1}_k(z_t)} \\
&\propto f(\boldsymbol{\pi}) \times \prod_{k=1}^{K} f(\boldsymbol{\beta}_k) f(\xi_k^2) f(\sigma_k^2) \times \prod_{t=1}^{T} \prod_{k=1}^{K} (\xi_k^2)^{-\frac{d}{2}} \exp\{-\frac{1}{2\xi_k^2} \boldsymbol{b}_{tk}^\intercal \boldsymbol{b}_{tk}\} \\
&\quad \times \prod_{t=1}^{T} \prod_{k=1}^{K} \left\{ (\sigma_k^2)^{-\frac{d}{2}} \exp\left[-\frac{1}{2\sigma_k^2} (\boldsymbol{y}_t - \boldsymbol{S}_t\boldsymbol{\beta}_k - \boldsymbol{S}_t\boldsymbol{b}_{tk})^\intercal (\boldsymbol{y}_t - \boldsymbol{S}_t\boldsymbol{\beta}_k - \boldsymbol{S}_t\boldsymbol{b}_{tk}) \right] \pi_k \right\}^{\mathbf{1}_k(z_t)} \\
&\propto f(\boldsymbol{\pi}) \times \prod_{k=1}^{K} f(\boldsymbol{\beta}_k) f(\xi_k^2) f(\sigma_k^2) (\xi_k^2)^{-\frac{dT}{2}} \times \prod_{k=1}^{K} \exp\{-\frac{1}{2\xi_k^2} \sum_{t=1}^{T} \boldsymbol{b}_{tk}^\intercal \boldsymbol{b}_{tk}\} \\
&\quad \times \prod_{k=1}^{K} (\sigma_k^2)^{-\frac{d}{2} \sum_{t=1}^{T} \mathbf{1}_k(z_t)} (\pi_k)^{\sum_{t=1}^{T} \mathbf{1}_k(z_t)} \exp\{-\frac{1}{2\sigma_k^2} \sum_{t=1}^{T} \mathbf{1}_k(z_t) (\boldsymbol{y}_t - \boldsymbol{S}_t\boldsymbol{\beta}_k - \boldsymbol{S}_t\boldsymbol{b}_{tk})^\intercal (\boldsymbol{y}_t - \boldsymbol{S}_t\boldsymbol{\beta}_k - \boldsymbol{S}_t\boldsymbol{b}_{tk})\} \\
&\propto f(\boldsymbol{\pi}) \times \prod_{k=1}^{K} f(\boldsymbol{\beta}_k) f(\xi_k^2) f(\sigma_k^2) (\xi_k^2)^{-\frac{dT}{2}} \times \prod_{k=1}^{K} \exp\{-\frac{1}{2\xi_k^2} \sum_{t=1}^{T} \boldsymbol{b}_{tk}^\intercal \boldsymbol{b}_{tk}\} (\sigma_k^2)^{-\frac{d}{2} \sum_{t=1}^{T} \mathbf{1}_k(z_t)} (\pi_k)^{\sum_{t=1}^{T} \mathbf{1}_k(z_t)} \\
&\quad \times \prod_{k=1}^{K} \exp\left\{ -\frac{1}{2\sigma_k^2} \sum_{t=1}^{T} \mathbf{1}_k(z_t) \left((\boldsymbol{y}_t - \boldsymbol{S}_t\boldsymbol{b}_{tk})^\intercal (\boldsymbol{y}_t - \boldsymbol{S}_t\boldsymbol{b}_{tk}) - 2(\boldsymbol{y}_t - \boldsymbol{S}_t\boldsymbol{b}_{tk})^\intercal \boldsymbol{S}_t\boldsymbol{\beta}_k + \boldsymbol{\beta}_k^\intercal \boldsymbol{S}_t^\intercal \boldsymbol{S}_t\boldsymbol{\beta}_k \right) \right\} \\
&\propto f(\boldsymbol{\pi}) \times \prod_{k=1}^{K} f(\boldsymbol{\beta}_k) f(\xi_k^2) f(\sigma_k^2) (\xi_k^2)^{-\frac{dT}{2}} \\
&\quad \times \prod_{k=1}^{K} \exp\{-\frac{1}{2\xi_k^2} \sum_{t=1}^{T} \boldsymbol{b}_{tk}^\intercal \boldsymbol{b}_{tk}\} \exp\{-\log(\sigma_k^2) \frac{d}{2} \sum_{t=1}^{T} \mathbf{1}_k(z_t)\} \exp\{\log(\pi_k) \sum_{t=1}^{T} \mathbf{1}_k(z_t)\} \\
&\quad \times \prod_{k=1}^{K} \exp\left\{ -\frac{1}{2\sigma_k^2} \sum_{t=1}^{T} \mathbf{1}_k(z_t) \left((\boldsymbol{y}_t - \boldsymbol{S}_t\boldsymbol{b}_{tk})^\intercal (\boldsymbol{y}_t - \boldsymbol{S}_t\boldsymbol{b}_{tk}) - 2\boldsymbol{\beta}_k \boldsymbol{S}_t^\intercal (\boldsymbol{y}_t - \boldsymbol{S}_t\boldsymbol{b}_{tk}) + \boldsymbol{\beta}_k^\intercal \boldsymbol{S}_t^\intercal \boldsymbol{S}_t\boldsymbol{\beta}_k \right) \right\}.
\end{aligned}
$$

Full conditional distribution of z_t

$$p(z_t|\boldsymbol{Y},\boldsymbol{\pi},\boldsymbol{\beta},\boldsymbol{b},\boldsymbol{\sigma}^2,\boldsymbol{\xi}^2) \quad \propto \prod_{k=1}^{K}\{\phi(\boldsymbol{y}_t;\boldsymbol{S}_t\boldsymbol{\beta}_k + S_i\boldsymbol{b}_{tk},\sigma_k^2\boldsymbol{I}_{m_t})p(z_t=k)\}^{\mathbf{1}_k(z_t)}.$$

Therefore,

$$z_t = k|\boldsymbol{Y},\boldsymbol{\pi},\boldsymbol{\beta},\boldsymbol{b},\boldsymbol{\sigma}^2,\boldsymbol{\xi}^2 \sim Mult(1;\tau_{t1},\ldots,\tau_{tK}), \tag{21}$$

where $\tau_{tk} = \frac{\phi(\boldsymbol{y}_t;\boldsymbol{S}_t\boldsymbol{\beta}_k+\boldsymbol{S}_t\boldsymbol{b}_{tk},\sigma_k^2\boldsymbol{I}_{m_t})\pi_k}{\sum_{j=1}^{K}\phi(\boldsymbol{y}_t;\boldsymbol{S}_t\boldsymbol{\beta}_j+\boldsymbol{S}_t\boldsymbol{b}_{tj},\sigma_j^2\boldsymbol{I}_{m_t})\pi_j},$ for $k=1,\ldots,K,\ t=1,\ldots,T.$

Full conditional distribution of $\boldsymbol{\pi}$

$$\begin{aligned}
p(\boldsymbol{\pi}|\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{\beta},\boldsymbol{b},\boldsymbol{\sigma}^2,\boldsymbol{\xi}^2) &\propto p(\boldsymbol{\pi})\prod_{t=1}^{T}\prod_{k=1}^{K}p(z_t=k)^{\mathbf{1}_k(z_t)}\\
&\propto \frac{1}{B(\alpha_1,\ldots,\alpha_K)}\prod_{k=1}^{K}\pi_k^{\alpha_k-1}\prod_{t=1}^{T}\prod_{k=1}^{K}\pi_k^{\mathbf{1}_k(z_t)}\\
&\propto (\prod_{k=1}^{K}\pi_k^{\alpha_k-1})\prod_{k=1}^{K}\pi_k^{n_{T,k}}\\
&\propto \prod_{k=1}^{K}\pi_k^{(\alpha_k+n_{T,k})-1}.
\end{aligned}$$

Therefore,

$$\boldsymbol{\pi}|\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{\beta},\boldsymbol{b},\boldsymbol{\sigma}^2,\boldsymbol{\xi}^2 \sim Dir(\alpha_1+n_{T,1},\ldots,\alpha_K+n_{T,K}), \tag{22}$$

where $n_{T,k} = \sum_{t=1}^{T}\mathbf{1}_k(z_t),$ for $k=1,\ldots,K.$

Full conditional distribution of $\boldsymbol{\beta_k}$

$$
\begin{aligned}
p(\boldsymbol{\beta}_k|\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{\pi},\boldsymbol{b},\boldsymbol{\sigma}^2,\boldsymbol{\xi}^2) &\propto f(\boldsymbol{\beta}_k|\boldsymbol{\mu}_0,\boldsymbol{\Sigma}_0)\prod_{t=1}^{T}\left\{\phi(\boldsymbol{y}_t|\boldsymbol{S}_t\boldsymbol{\beta}_k+S_t\boldsymbol{b}_{tk},\sigma_k^2\boldsymbol{I}_{m_t})\right\}^{\mathbf{1}_k(z_t)}\\
&\propto \left\{(2\pi)^{-\frac{d}{2}}|\boldsymbol{\Sigma}_0|^{-\frac{1}{2}}\exp\{-\frac{1}{2}(\boldsymbol{\beta}_k-\boldsymbol{\mu}_0)^\intercal\boldsymbol{\Sigma}_0^{-1}(\boldsymbol{\beta}_k-\boldsymbol{\mu}_0)\}\right\}\\
&\quad\times\prod_{t=1}^{T}\left\{(2\pi)^{-\frac{m_t}{2}}|\boldsymbol{I}_{m_t}\sigma_k^2|^{-\frac{1}{2}}\exp\left[-\frac{1}{2}(\boldsymbol{y}_t-\boldsymbol{S}_t\boldsymbol{\beta}_k-\boldsymbol{S}_t\boldsymbol{b}_{tk})^\intercal(\boldsymbol{I}_{m_t}\sigma_k^2)^{-1}(\boldsymbol{y}_t-\boldsymbol{S}_t\boldsymbol{\beta}_k-\boldsymbol{S}_t\boldsymbol{b}_{tk})\right]\right\}^{\mathbf{1}_k(z_t)}\\
&\propto \left\{\exp\{-\frac{1}{2}(\boldsymbol{\beta}_k-\boldsymbol{\mu}_0)^\intercal\boldsymbol{\Sigma}_0^{-1}(\boldsymbol{\beta}_k-\boldsymbol{\mu}_0)\}\right\}\\
&\quad\times\prod_{t=1}^{T}\left\{\exp\{-\frac{1}{2}(\boldsymbol{y}_t-\boldsymbol{S}_t\boldsymbol{\beta}_k-\boldsymbol{S}_t\boldsymbol{b}_{tk})^\intercal(\sigma_k^2)^{-1}(\boldsymbol{y}_t-\boldsymbol{S}_t\boldsymbol{\beta}_k-\boldsymbol{S}_t\boldsymbol{b}_{tk})\}\right\}^{\mathbf{1}_k(z_t)}\\
&\propto \exp\left\{-\frac{1}{2}\left(\beta_k^\intercal{\boldsymbol{\Sigma}_0}^{-1}\boldsymbol{\beta}_k-2\boldsymbol{\beta}_k^\intercal{\boldsymbol{\Sigma}_0}^{-1}\boldsymbol{\mu}_0+\boldsymbol{\mu}_0\boldsymbol{\Sigma}_0^{-1}\boldsymbol{\mu}_0\right)\right\}\\
&\quad\times\prod_{t=1}^{T}\exp\left\{-\frac{1}{2}\left(\boldsymbol{\beta}_k^\intercal\frac{\boldsymbol{S}_t^\intercal\boldsymbol{S}_t}{\sigma_k^2}\boldsymbol{\beta}_k-2\boldsymbol{\beta}_k^\intercal\frac{\boldsymbol{S}_t^\intercal(\boldsymbol{y}_t-\boldsymbol{S}_t\boldsymbol{b}_{tk})}{\sigma_k^2}+\frac{(\boldsymbol{y}_t-\boldsymbol{S}_t\boldsymbol{b}_{tk})^\intercal(\boldsymbol{y}_t-\boldsymbol{S}_t\boldsymbol{b}_{tk})}{\sigma_k^2}\right)\right\}^{\mathbf{1}_k(z_t)}\\
&\propto \exp\left\{-\frac{1}{2}\left(\boldsymbol{\beta}_k^\intercal{\boldsymbol{\Sigma}_0}^{-1}\boldsymbol{\beta}_k-2\boldsymbol{\beta}_k^\intercal{\boldsymbol{\Sigma}_0}^{-1}\boldsymbol{\mu}_0\right)\right\}\\
&\quad\times\prod_{t=1}^{T}\exp\left\{-\frac{1}{2}\left(\boldsymbol{\beta}_k^\intercal\frac{\boldsymbol{S}_t^\intercal\boldsymbol{S}_t}{\sigma_k^2}\boldsymbol{\beta}_k-2\boldsymbol{\beta}_k^\intercal\frac{\boldsymbol{S}_t^\intercal(\boldsymbol{y}_t-\boldsymbol{S}_t\boldsymbol{b}_{tk})}{\sigma_k^2}\right)\right\}^{\mathbf{1}_k(z_t)}\\
&\propto \exp\left\{-\frac{1}{2}\left(\boldsymbol{\beta}_k^\intercal{\boldsymbol{\Sigma}_0}^{-1}\boldsymbol{\beta}_k-2\boldsymbol{\beta}_k^\intercal{\boldsymbol{\Sigma}_0}^{-1}\boldsymbol{\mu}_0\right)\right\}\\
&\quad\times\exp\left\{-\frac{1}{2}\left(\boldsymbol{\beta}_k^\intercal\sum_{t=1}^{T}\mathbf{1}_k(z_t)\frac{\boldsymbol{S}_t^\intercal\boldsymbol{S}_t}{\sigma_k^2}\boldsymbol{\beta}_k-2\boldsymbol{\beta}_k^\intercal\sum_{t=1}^{T}\mathbf{1}_k(z_t)\frac{\boldsymbol{S}_t^\intercal(\boldsymbol{y}_t-\boldsymbol{S}_t\boldsymbol{b}_{tk})}{\sigma_k^2}\right)\right\}\\
&\propto \exp\left\{-\frac{1}{2}\left[\boldsymbol{\beta}_k^\intercal\left(\boldsymbol{\Sigma}_0^{-1}+\sum_{t=1}^{T}\mathbf{1}_k(z_t)\frac{\boldsymbol{S}_t^\intercal\boldsymbol{S}_t}{\sigma_k^2}\right)\boldsymbol{\beta}_k-2\boldsymbol{\beta}_k^\intercal\left(\boldsymbol{\Sigma}_0^{-1}\boldsymbol{\mu}_0+\sum_{t=1}^{T}\mathbf{1}_k(z_t)\frac{\boldsymbol{S}_t^\intercal(\boldsymbol{y}_t-\boldsymbol{S}_t\boldsymbol{b}_{tk})}{\sigma_k^2}\right)\right]\right\}.
\end{aligned}
$$

Therefore,

$$\boldsymbol{\beta}_k|\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{\pi},\boldsymbol{b},\boldsymbol{\sigma}^2,\boldsymbol{\xi}^2\sim MVN(\boldsymbol{\mu}_{\boldsymbol{\beta}_k},\boldsymbol{\Sigma}_{\boldsymbol{\beta}_k}),\tag{23}$$

where ${\boldsymbol{\Sigma}_{\boldsymbol{\beta}_k}}^{-1}=\boldsymbol{\Sigma}_0^{-1}+\sum_{t=1}^{T}\mathbf{1}_k(z_t)\frac{\boldsymbol{S}_t^\intercal\boldsymbol{S}_t}{\sigma_k^2}$, $\boldsymbol{\mu}_{\boldsymbol{\beta}_k}=\boldsymbol{\Sigma}_{\boldsymbol{\beta}_k}\left(\boldsymbol{\Sigma}_0^{-1}\boldsymbol{\mu}_0+\sum_{t=1}^{T}\mathbf{1}_k(z_t)\frac{\boldsymbol{S}_t^\intercal(\boldsymbol{y}_t-\boldsymbol{S}_t\boldsymbol{b}_{tk})}{\sigma_k^2}\right)$, for $k=1,\ldots,K$.

Full conditional distribution of $\boldsymbol{b_{tk}}$

$$
\begin{aligned}
&p(\boldsymbol{b}_{tk}|\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{\pi},\boldsymbol{\beta},\boldsymbol{\sigma}^2,\boldsymbol{\xi}^2) \propto f(\boldsymbol{b}_{tk}|0_d,\xi_k^2\boldsymbol{I}_d)\phi(\boldsymbol{y}_t;\boldsymbol{S}_t\boldsymbol{\beta}_k+\boldsymbol{S}_t\boldsymbol{b}_{tk},\sigma_k^2\boldsymbol{I}_{m_t})^{\mathbf{1}_k(z_t)}\\
&\propto (2\pi)^{-\frac{d}{2}}|\xi_k^2\boldsymbol{I}_d|^{-\frac{1}{2}}\exp\{-\frac{1}{2}\boldsymbol{b}_{tk}^\intercal\frac{1}{\xi_k^2}\boldsymbol{I}_d\boldsymbol{b}_{tk}\}\\
&\times\left\{(2\pi)^{-\frac{m_t}{2}}|\boldsymbol{I}_{m_t}\sigma_k^2|^{-\frac{1}{2}}\exp\{-\frac{1}{2}(\boldsymbol{y}_t-\boldsymbol{S}_t\boldsymbol{\beta}_k-\boldsymbol{S}_t\boldsymbol{b}_{tk})^\intercal(\boldsymbol{I}_{m_t}\sigma_k^2)^{-1}(\boldsymbol{y}_t-\boldsymbol{S}_t\boldsymbol{\beta}_k-\boldsymbol{S}_t\boldsymbol{b}_{tk})\}\right\}^{\mathbf{1}_k(z_t)}\\
&\propto \exp\{-\frac{1}{2}\boldsymbol{b}_{tk}^\intercal\frac{1}{\xi_k^2}\boldsymbol{I}_d\boldsymbol{b}_{tk}\}\\
&\times\left\{\exp\{-\frac{1}{2}(\boldsymbol{y}_t-\boldsymbol{S}_t\boldsymbol{\beta}_k-\boldsymbol{S}_t\boldsymbol{b}_{tk})^\intercal(\boldsymbol{I}_{m_t}\sigma_k^2)^{-1}(\boldsymbol{y}_t-\boldsymbol{S}_t\boldsymbol{\beta}_k-\boldsymbol{S}_t\boldsymbol{b}_{tk})\}\right\}^{\mathbf{1}_k(z_t)}\\
&\propto \exp\{-\frac{1}{2}\boldsymbol{b}_{tk}^\intercal\frac{1}{\xi_k^2}\boldsymbol{I}_d\boldsymbol{b}_{tk}\}\\
&\times\exp\left\{-\frac{1}{2}\left(\boldsymbol{b}_{tk}^\intercal\frac{\boldsymbol{S}_t^\intercal\boldsymbol{S}_t}{\sigma_k^2}\boldsymbol{b}_{tk}-2\boldsymbol{b}_{tk}^\intercal\frac{\boldsymbol{S}_t^\intercal(\boldsymbol{y}_t-\boldsymbol{S}_t\boldsymbol{\beta}_k)}{\sigma_k^2}+\frac{(\boldsymbol{y}_t-\boldsymbol{S}_t\boldsymbol{\beta}_k)^\intercal(\boldsymbol{y}_t-\boldsymbol{S}_t\boldsymbol{\beta}_k)}{\sigma_k^2}\right)\right\}^{\mathbf{1}_k(z_t)}\\
&\propto \exp\{-\frac{1}{2}\boldsymbol{b}_{tk}^\intercal\frac{1}{\xi_k^2}\boldsymbol{I}_d\boldsymbol{b}_{tk}\}\\
&\times\exp\left\{-\frac{1}{2}\left(\boldsymbol{b}_{tk}^\intercal\frac{\boldsymbol{S}_t^\intercal\boldsymbol{S}_t}{\sigma_k^2}\boldsymbol{b}_{tk}-2\boldsymbol{b}_{tk}^\intercal\frac{\boldsymbol{S}_t^\intercal(\boldsymbol{y}_t-\boldsymbol{S}_t\boldsymbol{\beta}_k)}{\sigma_k^2}\right)\right\}^{\mathbf{1}_k(z_t)}\\
&\propto \exp\left\{-\frac{1}{2}\left[\boldsymbol{b}_{tk}^\intercal\left(\frac{1}{\xi_k^2}\boldsymbol{I}_d+\mathbf{1}_k(z_t)\frac{\boldsymbol{S}_t^\intercal\boldsymbol{S}_t}{\sigma_k^2}\right)\boldsymbol{b}_{tk}-2\boldsymbol{b}_{tk}^\intercal\mathbf{1}_k(z_t)\frac{\boldsymbol{S}_t^\intercal(\boldsymbol{y}_t-\boldsymbol{S}_t\boldsymbol{\beta}_k)}{\sigma_k^2}\right]\right\}.
\end{aligned}
$$

Therefore,

$$\boldsymbol{b}_{tk}|\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{\pi},\boldsymbol{\beta},\boldsymbol{\sigma}^2,\boldsymbol{\xi}^2 \sim MVN(\boldsymbol{\mu}_{\boldsymbol{b}_{tk}},\boldsymbol{\Sigma}_{\boldsymbol{b}_{tk}}), \tag{24}$$

where ${\boldsymbol{\Sigma}_{\boldsymbol{b}_{tk}}}^{-1} = \frac{1}{\xi_k^2}\boldsymbol{I}_d + \mathbf{1}_k(z_t)\frac{\boldsymbol{S}_t^\intercal\boldsymbol{S}_t}{\sigma_k^2}$, $\boldsymbol{\mu}_{\boldsymbol{b}_{tk}} = \mathbf{1}_k(z_t)\boldsymbol{\Sigma}_{\boldsymbol{b}_{tk}}\frac{\boldsymbol{S}_t^\intercal(\boldsymbol{y}_t-\boldsymbol{S}_t\boldsymbol{\beta}_k)}{\sigma_k^2}$, for $k = 1,\ldots,K$.

Full conditional distribution of σ_k^2

$$
\begin{aligned}
&p(\sigma_k^2|\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{\pi},\boldsymbol{\beta},\boldsymbol{b},\boldsymbol{\xi}^2) \propto f(\sigma_k^2) f(\boldsymbol{Y}|\boldsymbol{Z},\boldsymbol{\beta}_k,\boldsymbol{b}_{tk},\sigma_k^2,\xi_k^2,\pi_k)\\
&\propto \frac{{h_0}^{g_0}}{\Gamma(g_0)}(\sigma_k^2)^{-g_0-1}\exp\left\{-\frac{h_0}{\sigma_k^2}\right\}\\
&\times\prod_{t=1}^{T}\left\{(2\pi)^{-\frac{m_t}{2}}|\boldsymbol{I}_{m_t}\sigma_k^2|^{-\frac{1}{2}}\exp\left[-\frac{1}{2}(\boldsymbol{y}_t-\boldsymbol{S}_t\boldsymbol{\beta}_k-\boldsymbol{S}_t\boldsymbol{b}_{tk})^\intercal(\boldsymbol{I}_{m_t}\sigma_k^2)^{-1}(\boldsymbol{y}_t-\boldsymbol{S}_t\boldsymbol{\beta}_k-\boldsymbol{S}_t\boldsymbol{b}_{tk})\right]\right\}^{\mathbf{1}_k(z_t)}\\
&\propto (\sigma_k^2)^{-g_0-1}\exp\left\{-\frac{h_0}{\sigma_k^2}\right\}\\
&\times\prod_{t=1}^{T}\left\{\sigma_k^{2-\frac{m_t}{2}}\exp\left[-\frac{1}{2}(\boldsymbol{y}_t-\boldsymbol{S}_t\boldsymbol{\beta}_k-\boldsymbol{S}_t\boldsymbol{b}_{tk})^\intercal\frac{1}{\sigma_k^2}(\boldsymbol{y}_t-\boldsymbol{S}_t\boldsymbol{\beta}_k-\boldsymbol{S}_t\boldsymbol{b}_{tk})\right]\right\}^{\mathbf{1}_k(z_t)}\\
&\propto (\sigma_k^2)^{-g_0-1}\exp\left\{-\frac{h_0}{\sigma_k^2}\right\}\\
&\times\left\{\sigma_k^{2-\frac{m_t}{2}\sum_{t=1}^{T}\mathbf{1}_k(z_t)}\exp\left[-\frac{\sum_{t=1}^{T}\mathbf{1}_k(z_t)(\boldsymbol{y}_t-\boldsymbol{S}_t\boldsymbol{\beta}_k-\boldsymbol{S}_t\boldsymbol{b}_{tk})^\intercal(\boldsymbol{y}_t-\boldsymbol{S}_t\boldsymbol{\beta}_k-\boldsymbol{S}_t\boldsymbol{b}_{tk})}{2\sigma_k^2}\right]\right\}\\
&\propto (\sigma_k^2)^{-\left\{g_0+\frac{m_t}{2}\sum_{t=1}^{T}\mathbf{1}_k(z_t)\right\}-1}\\
&\times\exp\left\{-\frac{h_0+\frac{\sum_{t=1}^{T}\mathbf{1}_k(z_t)(\boldsymbol{y}_t-\boldsymbol{S}_t\boldsymbol{\beta}_k-\boldsymbol{S}_t\boldsymbol{b}_{tk})^\intercal(\boldsymbol{y}_t-\boldsymbol{S}_t\boldsymbol{\beta}_k-\boldsymbol{S}_t\boldsymbol{b}_{tk})}{2}}{\sigma_k^2}\right\}.
\end{aligned}
$$

Therefore,

$$\boldsymbol{\sigma}_k^2|\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{\pi},\boldsymbol{b},\boldsymbol{\beta},\boldsymbol{\xi}^2 \sim IG(g_0^*,h_0^*), \tag{25}$$

where $g_0^* = g_0 + \frac{n_{T,k}}{2}m_t$, $h_0^* = h_0 + \frac{\sum_{t=1}^{T}\mathbf{1}_k(z_t)(\boldsymbol{y}_t-\boldsymbol{S}_t\boldsymbol{\beta}_k-\boldsymbol{S}_t\boldsymbol{b}_{tk})^\intercal(\boldsymbol{y}_t-\boldsymbol{S}_t\boldsymbol{\beta}_k-\boldsymbol{S}_t\boldsymbol{b}_{tk})}{2}$, $n_{T,k} = \sum_{t=1}^{T}\mathbf{1}_k(z_t)$, for $k = 1,\ldots,K$.

Full conditional distribution of ξ_k^2

$$
\begin{aligned}
p(\xi_k^2|\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{\pi},\boldsymbol{\beta},\boldsymbol{b},\boldsymbol{\sigma}^2) &\propto f(\xi_k^2)\prod_{t=1}^{T} f(\boldsymbol{b}_{tk}|\xi_k^2)\\
&\propto \frac{\boldsymbol{b}_0{}^{a_0}}{\Gamma(a_0)}(\xi_k^2)^{-a_0-1}\exp\left\{-\frac{b_0}{\xi_k^2}\right\}\\
&\times\prod_{t=1}^{T}(2\pi)^{-\frac{d}{2}}|\xi_k^2\boldsymbol{I}_d|^{-\frac{1}{2}}\exp\{-\frac{1}{2}\boldsymbol{b}_{tk}^{\intercal}\frac{1}{\xi_k^2}\boldsymbol{I}_d\boldsymbol{b}_{tk}\}\\
&\propto (\xi_k^2)^{-a_0-1}\exp\left\{-\frac{b_0}{\xi_k^2}\right\}\\
&\times\prod_{t=1}^{T}(\xi_k^2)^{-\frac{d}{2}}\exp\{-\frac{1}{2}\boldsymbol{b}_{tk}^{\intercal}\frac{1}{\xi_k^2}\boldsymbol{b}_{tk}\}\\
&\propto (\xi_k^2)^{-(a_0+\frac{nd}{2})-1}\exp\left\{-\frac{b_0+\frac{1}{2}\sum_{t=1}^{T}\boldsymbol{b}_{tk}^{\intercal}\boldsymbol{b}_{tk}}{\xi_k^2}\right\}.
\end{aligned}
$$

Therefore,

$$
\xi_k^2|\boldsymbol{Y},\boldsymbol{Z},\boldsymbol{\pi},\boldsymbol{b},\boldsymbol{\beta},\boldsymbol{\sigma}^2 \sim IG(a_0^*,b_0^*), \tag{26}
$$

where $a_0^* = a_0 + \frac{Td}{2}$, $b_0^* = b_0 + \frac{1}{2}\sum_{t=1}^{T}\boldsymbol{b}_{tk}^{\intercal}\boldsymbol{b}_{tk}$, for $k = 1,\ldots,K$.

6.9.2 Proof of Proposition 1

As indicated in Section 6.9.1, the joint posterior distribution of $\boldsymbol{\pi},\boldsymbol{\beta},\boldsymbol{\sigma}^2$, conditional on $z_{1:t},\boldsymbol{b}_{1:t},\boldsymbol{y}_{1:t}$ can be expressed up to a marginal likelihood as

$$
\begin{aligned}
&p(\boldsymbol{\pi},\boldsymbol{\beta},\boldsymbol{\sigma}^2,\boldsymbol{\xi}^2|z_{1:t},\boldsymbol{b}_{1:t},\boldsymbol{y}_{1:t}) \propto f(\boldsymbol{\pi})f(\boldsymbol{\beta}^2)f(\boldsymbol{\xi}^2)f(\boldsymbol{\sigma}^2)f(\boldsymbol{y}_{1:t},z_{1:t}|\boldsymbol{\beta},\boldsymbol{b}_{1:t},\boldsymbol{\xi}^2,\boldsymbol{\sigma}^2)\\
&\propto f(\boldsymbol{\pi})\times\prod_{k=1}^{K} f(\boldsymbol{\beta}_k)f(\xi_k^2)f(\sigma_k^2)(\xi_k^2)^{-\frac{dt}{2}}\\
&\times\prod_{k=1}^{K}\exp\{-\frac{1}{2\xi_k^2}\sum_{t=1}^{t}\boldsymbol{b}_{t'k}^{\intercal}\boldsymbol{b}_{t'k}\}\exp\{-\log(\sigma_k^2)\frac{d}{2}\sum_{t=1}^{t}\mathbf{1}_k(z_{t'})\}\exp\{\log(\pi_k)\sum_{t'=1}^{t}\mathbf{1}_k(z_{t'})\}\\
&\times\prod_{k=1}^{K}\exp\left\{-\frac{1}{2\sigma_k^2}\sum_{t'=1}^{t}\mathbf{1}_k(z_{t'})\left((\boldsymbol{y}_{t'}-\boldsymbol{S}_{t'}\boldsymbol{b}_{t'k})^{\intercal}(\boldsymbol{y}_{t'}-\boldsymbol{S}_{t'}\boldsymbol{b}_{t'k})-2\boldsymbol{\beta}_k\boldsymbol{S}_{t'}^{\intercal}(\boldsymbol{y}_{t'}-\boldsymbol{S}_{t'}\boldsymbol{b}_{t'k})+\boldsymbol{\beta}_k^{\intercal}\boldsymbol{S}_{t'}^{\intercal}\boldsymbol{S}_{t'}\boldsymbol{\beta}_k\right)\right\}
\end{aligned}
$$

$$
\begin{aligned}
&\propto f(\boldsymbol{\pi}) \times \prod_{k=1}^{K} f(\boldsymbol{\beta}_k) f(\xi_k^2) f(\sigma_k^2) (\xi_k^2)^{-\frac{dt}{2}} \\
&\times \prod_{k=1}^{K} \exp\{-\frac{1}{2\xi_k^2} \sum_{t'=1}^{t} \boldsymbol{b}_{t'k}^{\intercal} \boldsymbol{b}_{t'k}\} \exp\{-\log(\sigma_k^2) \frac{d}{2} \sum_{t'=1}^{t} \mathbf{1}_k(z_{t'})\} \exp\{\log(\pi_k) \sum_{t'=1}^{t} \mathbf{1}_k(z_{t'})\} \\
&\times \prod_{k=1}^{K} \exp\left\{-\frac{1}{2\sigma_k^2}\left[\sum_{t'=1}^{t} \mathbf{1}_k(z_{t'})(\boldsymbol{y}_{t'} - \boldsymbol{S}_{t'}\boldsymbol{b}_{t'k})^{\intercal}(\boldsymbol{y}_{t'} - \boldsymbol{S}_{t'}\boldsymbol{b}_{t'k}) - 2\boldsymbol{\beta}_k \sum_{t'=1}^{t} \mathbf{1}_k(z_{t'}) \boldsymbol{S}_{t'}^{\intercal}(\boldsymbol{y}_{t'} - \boldsymbol{S}_{t'}\boldsymbol{b}_{t'k})\right]\right\} \\
&\times \prod_{k=1}^{K} \exp\left\{-\frac{1}{2\sigma_k^2}\left[\boldsymbol{\beta}_k^{\intercal}(\sum_{t'=1}^{t} \mathbf{1}_k(z_{t'}) \boldsymbol{S}_{t'}^{\intercal}\boldsymbol{S}_{t'})\boldsymbol{\beta}_k\right]\right\}.
\end{aligned}
$$

The sufficient statistics $\boldsymbol{s_t}$ for the MSSR_m model given ($\{\boldsymbol{y}_i^{\intercal}\}_{1:t}$, $z_{1:t}$, $\{\boldsymbol{b}_i^{\intercal}\}_{1:t}$) for $t = 1, \ldots, T$ can be written as

$$
\boldsymbol{s_t} = \left\{ \sum_{t'=1}^{t} \boldsymbol{b}_{t'k}^{\intercal} \boldsymbol{b}_{t'k}, \sum_{t'=1}^{t} \mathbf{1}_k(z_{t'}), \sum_{t'=1}^{t} \mathbf{1}_k(z_{t'}) \boldsymbol{S}_{t'}^{\intercal} \boldsymbol{y}_{t'}^{*}, \sum_{t'=1}^{t} \mathbf{1}_k(z_{t'}) \boldsymbol{y}_{t'}^{*\intercal} \boldsymbol{y}_{t'}^{*}, \sum_{t'=1}^{t} \mathbf{1}_k(z_{t'}) \boldsymbol{S}_{t'}^{\intercal} \boldsymbol{S}_{t'} \right\}_{k=1}^{K}
$$

where $\boldsymbol{y}_{t'}^{*} = \boldsymbol{y}_{t'} - \boldsymbol{S}_{t'}\boldsymbol{b}_{t'k}$.

6.10 Appendix C

6.10.1 Proof of Proposition 2

Our online SMC algorithm is run for time steps $t = 1, \ldots, T$, and samples the random variables $\boldsymbol{\theta}_t = \{\boldsymbol{\theta}_t^{(i)}\}_{i=1}^{N}$, $\boldsymbol{x}_t = \{\boldsymbol{x}_t^{(i)}\}_{i=1}^{N}$, $\boldsymbol{A}_t = \{A_t^{(i)}\}_{i=1}^{N}$. The distributions of these random variables are:

$$
\begin{aligned}
A_t^{(i)} &\sim W_{t-1}^{(i)}, && (27)\\
\boldsymbol{x}_t^{(i)} &\sim p(\boldsymbol{x}_t | \boldsymbol{\theta}_{t-1}^{A_t^{(i)}}, \boldsymbol{y}_t), && (28)\\
\boldsymbol{\theta}_t^{(i)} &\sim p(\boldsymbol{\theta}_t | \boldsymbol{x}_{1:t}^{(i)}, \boldsymbol{y}_{1:t}). && (29)
\end{aligned}
$$

Note that Equation 29 is true only if the Gibbs chain for $\boldsymbol{\theta}_t^{(i)}$ reaches stationary.

The distribution of all random variables is

$$
\psi_{N,T}(\boldsymbol{x}_{1:T-1}, \boldsymbol{\theta}_{0:T-1}, \boldsymbol{A}_{1:T-1}) = \left\{ \prod_{i=1}^{N} p(\boldsymbol{\theta}_0^{(i)}) \right\} \prod_{t=1}^{T-1} \left\{ \prod_{i=1}^{N} W_{t-1}^{A_t^{(i)}} p(\boldsymbol{x}_t | \boldsymbol{\theta}_{t-1}^{A_t^{(i)}}, \boldsymbol{y}_t) p(\boldsymbol{\theta}_t | \boldsymbol{x}_{1:t}^{(i)}, \boldsymbol{y}_{1:t}) \right\}. \quad (30)
$$

To prove

$$
E_{\psi_{N,T}(\boldsymbol{x}_{1:T-1}, \boldsymbol{\theta}_{0:T-1}, \boldsymbol{A}_{1:T-1})} \left(\prod_{t=1}^{T} \frac{1}{N} \sum_{i=1}^{N} p(\boldsymbol{y}_t | \boldsymbol{\theta}_{t-1}^{(i)}) \right) = p(\boldsymbol{y}_{1:T}), \quad (31)
$$

we solve the integral

$$
\begin{aligned}
& E_{\psi_{N,T}(\boldsymbol{x}_{1:T-1},\boldsymbol{\theta}_{0:T-1},\boldsymbol{A}_{1:T-1})}\left(\prod_{t=1}^{T}\frac{1}{N}\sum_{i=1}^{N}p(\boldsymbol{y}_t|\boldsymbol{\theta}_{t-1}^{(i)})\right) \\
= & \int\prod_{t=1}^{T}\frac{1}{N}\sum_{i=1}^{N}p(\boldsymbol{y}_t|\boldsymbol{\theta}_{t-1}^{(i)})\Big\{\prod_{i=1}^{N}p(\boldsymbol{\theta}_0^{(i)})\Big\}\prod_{t=1}^{T-1}\Big\{\prod_{i=1}^{N}W_{t-1}^{A_t^{(i)}}p(\boldsymbol{x}_t|\boldsymbol{\theta}_{t-1}^{A_t^{(i)}},\boldsymbol{y}_t) \\
& \times p(\boldsymbol{\theta}_t|\boldsymbol{x}_{1:t}^{(i)},\boldsymbol{y}_{1:t})\Big\}d\boldsymbol{x}_{1:T-1}d\boldsymbol{\theta}_{0:T-1}d\boldsymbol{A}_{1:T-1} \\
= & \int\Big\{\int\frac{1}{N}\sum_{i=1}^{N}p(\boldsymbol{y}_T|\boldsymbol{\theta}_{T-1}^{(i)})\Big\{\prod_{i=1}^{N}W_{T-2}^{A_{T-1}^{(i)}}p(\boldsymbol{x}_{T-1}|\boldsymbol{\theta}_{T-2}^{A_{T-1}^{(i)}},\boldsymbol{y}_{T-1}) \\
& \times p(\boldsymbol{\theta}_{T-1}|\boldsymbol{x}_{1:T-1}^{(i)},\boldsymbol{y}_{1:T-1})\Big\}d\boldsymbol{x}_{T-1}d\boldsymbol{\theta}_{T-1}d\boldsymbol{A}_{T-1}\Big\} \\
& \times\prod_{t=1}^{T-1}\frac{1}{N}\sum_{i=1}^{N}p(\boldsymbol{y}_t|\boldsymbol{\theta}_{t-1}^{(i)})\Big\{\prod_{i=1}^{N}p(\boldsymbol{\theta}_0^{(i)})\Big\} \\
& \times\prod_{t=1}^{T-2}\Big\{\prod_{i=1}^{N}W_{t-1}^{A_t^{(i)}}p(\boldsymbol{x}_t|\boldsymbol{\theta}_{t-1}^{A_t^{(i)}},\boldsymbol{y}_t)p(\boldsymbol{\theta}_t|\boldsymbol{x}_{1:t}^{(i)},\boldsymbol{y}_{1:t})\Big\}d\boldsymbol{x}_{1:T-2}d\boldsymbol{\theta}_{0:T-2}d\boldsymbol{A}_{1:T-2}
\end{aligned}
$$

We first consider the integral over $(\boldsymbol{x}_{T-1},\boldsymbol{\theta}_{T-1},\boldsymbol{A}_{T-1})$,

$$
\begin{aligned}
& \int\frac{1}{N}\sum_{i=1}^{N}p(\boldsymbol{y}_T|\boldsymbol{\theta}_{T-1}^{(i)})\Big\{\prod_{i=1}^{N}W_{T-2}^{A_{T-1}^{(i)}}p(\boldsymbol{x}_{T-1}|\boldsymbol{\theta}_{T-2}^{A_{T-1}^{(i)}},\boldsymbol{y}_{T-1}) \\
& \times p(\boldsymbol{\theta}_{T-1}|\boldsymbol{x}_{1:T-1}^{(i)},\boldsymbol{y}_{1:T-1})\Big\}d\boldsymbol{x}_{T-1}d\boldsymbol{\theta}_{T-1}d\boldsymbol{A}_{T-1} \\
= & \sum_{i=1}^{N}W_{T-2}^{(i)}p(\boldsymbol{y}_T|\boldsymbol{\theta}_{T-2}^{(i)},\boldsymbol{x}_{1:T-2}^{(i)},\boldsymbol{y}_{1:T-1}).
\end{aligned}
$$

Hence,

$$
\begin{aligned}
& E_{\psi_{N,T}(\boldsymbol{x}_{1:T-1},\boldsymbol{\theta}_{0:T-1},\boldsymbol{A}_{1:T-1})}\left(\prod_{t=1}^{T}\frac{1}{N}\sum_{i=1}^{N}p(\boldsymbol{y}_t|\boldsymbol{\theta}_{t-1}^{(i)})\right) \\
& = \int\Big\{\int\sum_{i=1}^{N}W_{T-2}^{(i)}p(\boldsymbol{y}_T|\boldsymbol{\theta}_{T-2}^{(i)},\boldsymbol{x}_{1:T-2}^{(i)},\boldsymbol{y}_{1:T-1})\frac{1}{N}\sum_{i=1}^{N}p(\boldsymbol{y}_{T-1}|\boldsymbol{\theta}_{T-2}^{(i)}) \\
& \times\Big\{\prod_{i=1}^{N}W_{T-3}^{A_{T-2}^{(i)}}p(\boldsymbol{x}_{T-2}|\boldsymbol{\theta}_{T-3}^{A_{T-2}^{(i)}},\boldsymbol{y}_{T-2})p(\boldsymbol{\theta}_{T-2}|\boldsymbol{x}_{1:T-2}^{(i)},\boldsymbol{y}_{1:T-2})\Big\}d\boldsymbol{x}_{T-2}d\boldsymbol{\theta}_{T-2}d\boldsymbol{A}_{T-2}\Big\} \\
& \times\prod_{t=1}^{T-2}\frac{1}{N}\sum_{i=1}^{N}p(\boldsymbol{y}_t|\boldsymbol{\theta}_{t-1}^{(i)})\Big\{\prod_{i=1}^{N}p(\boldsymbol{\theta}_0^{(i)})\Big\}\prod_{t=1}^{T-3}\Big\{\prod_{i=1}^{N}W_{t-1}^{A_t^{(i)}}p(\boldsymbol{x}_t|\boldsymbol{\theta}_{t-1}^{A_t^{(i)}},\boldsymbol{y}_t) \\
& \times p(\boldsymbol{\theta}_t|\boldsymbol{x}_{1:t}^{(i)},\boldsymbol{y}_{1:t})\Big\}d\boldsymbol{x}_{1:T-2}d\boldsymbol{\theta}_{0:T-2}d\boldsymbol{A}_{1:T-2}.
\end{aligned}
$$

We then consider the integral over $(\boldsymbol{x}_{T-2}, \boldsymbol{\theta}_{T-2}, \boldsymbol{A}_{T-2})$, recall that $W_{t-1}^{(i)} = p(\boldsymbol{y}_t|\boldsymbol{\theta}_{t-1}^{(i)})/\sum_{i=1}^N p(\boldsymbol{y}_t|\boldsymbol{\theta}_{t-1}^{(i)})$ for $t = 1, \cdots, T$,

$$
\begin{aligned}
& \int \sum_{i=1}^N W_{T-2}^{(i)} p(\boldsymbol{y}_T|\boldsymbol{\theta}_{T-2}^{(i)}, \boldsymbol{x}_{1:T-2}^{(i)}, \boldsymbol{y}_{1:T-1}) \frac{1}{N} \sum_{i=1}^N p(\boldsymbol{y}_{T-1}|\boldsymbol{\theta}_{T-2}^{(i)}) \\
& \times \Big\{ \prod_{i=1}^N W_{T-3}^{A_{T-2}^{(i)}} p(\boldsymbol{x}_{T-2}|\boldsymbol{\theta}_{T-3}^{A_{T-2}^{(i)}}, \boldsymbol{y}_{T-2}) p(\boldsymbol{\theta}_{T-2}|\boldsymbol{x}_{1:T-2}^{(i)}, \boldsymbol{y}_{1:T-2}) \Big\} d\boldsymbol{x}_{T-2} d\boldsymbol{\theta}_{T-2} d\boldsymbol{A}_{T-2} \Big\} \\
= & \int \frac{1}{N} \sum_{i=1}^N p(\boldsymbol{y}_{T-1}|\boldsymbol{\theta}_{T-2}^{(i)}) p(\boldsymbol{y}_T|\boldsymbol{\theta}_{T-2}^{(i)}, \boldsymbol{x}_{1:T-2}^{(i)}, \boldsymbol{y}_{1:T-1}) \\
& \times \Big\{ \prod_{i=1}^N W_{T-3}^{A_{T-2}^{(i)}} p(\boldsymbol{x}_{T-2}|\boldsymbol{\theta}_{T-3}^{A_{T-2}^{(i)}}, \boldsymbol{y}_{T-2}) p(\boldsymbol{\theta}_{T-2}|\boldsymbol{x}_{1:T-2}^{(i)}, \boldsymbol{y}_{1:T-2}) \Big\} d\boldsymbol{x}_{T-2} d\boldsymbol{\theta}_{T-2} d\boldsymbol{A}_{T-2} \Big\} \\
= & \int \frac{1}{N} \sum_{i=1}^N p(\boldsymbol{y}_{T-1:T}|\boldsymbol{\theta}_{T-2}^{(i)}, \boldsymbol{x}_{1:T-2}^{(i)}, \boldsymbol{y}_{1:T-2}) \\
& \times \Big\{ \prod_{i=1}^N W_{T-3}^{A_{T-2}^{(i)}} p(\boldsymbol{x}_{T-2}|\boldsymbol{\theta}_{T-3}^{A_{T-2}^{(i)}}, \boldsymbol{y}_{T-2}) p(\boldsymbol{\theta}_{T-2}|\boldsymbol{x}_{1:T-2}^{(i)}, \boldsymbol{y}_{1:T-2}) \Big\} d\boldsymbol{x}_{T-2} d\boldsymbol{\theta}_{T-2} d\boldsymbol{A}_{T-2} \Big\} \\
= & \sum_{i=1}^N W_{T-3}^{(i)} p(\boldsymbol{y}_{T-1:T}|\boldsymbol{\theta}_{T-3}^{(i)}, \boldsymbol{x}_{1:T-3}^{(i)}, \boldsymbol{y}_{1:T-2}).
\end{aligned}
$$

Similarly, we can integrate over $(\boldsymbol{x}_t, \boldsymbol{\theta}_t, \boldsymbol{A}_t)$ $(t = T-3, T-4, \ldots, 1)$ and get

$$
\begin{aligned}
& E_{\psi_{N,T}(\boldsymbol{x}_{1:T-1}, \boldsymbol{\theta}_{0:T-1}, \boldsymbol{A}_{1:T-1})} \left(\prod_{t=1}^T \frac{1}{N} \sum_{i=1}^N p(\boldsymbol{y}_t|\boldsymbol{\theta}_{t-1}^{(i)}) \right) \\
= & \int \sum_{i=1}^N W_0^{(i)} p(\boldsymbol{y}_{2:T}|\boldsymbol{\theta}_0^{(i)}, \boldsymbol{y}_1) \times \frac{1}{N} \sum_{i=1}^N p(\boldsymbol{y}_1|\boldsymbol{\theta}_0^{(i)}) \times \prod_{i=1}^N p(\boldsymbol{\theta}_0^{(i)}) d\boldsymbol{\theta}_0 \\
= & \int \frac{1}{N} \sum_{i=1}^N p(\boldsymbol{y}_1|\boldsymbol{\theta}_0^{(i)}) p(\boldsymbol{y}_{2:T}|\boldsymbol{\theta}_0^{(i)}, \boldsymbol{y}_1) \times \prod_{i=1}^N p(\boldsymbol{\theta}_0^{(i)}) d\boldsymbol{\theta}_0 \\
= & \int \frac{1}{N} \sum_{i=1}^N p(\boldsymbol{y}_{1:T}|\boldsymbol{\theta}_0^{(i)}) \times \prod_{i=1}^N p(\boldsymbol{\theta}_0^{(i)}) d\boldsymbol{\theta}_0 \\
= & p(\boldsymbol{y}_{1:T}).
\end{aligned}
$$

This proves the proposition.

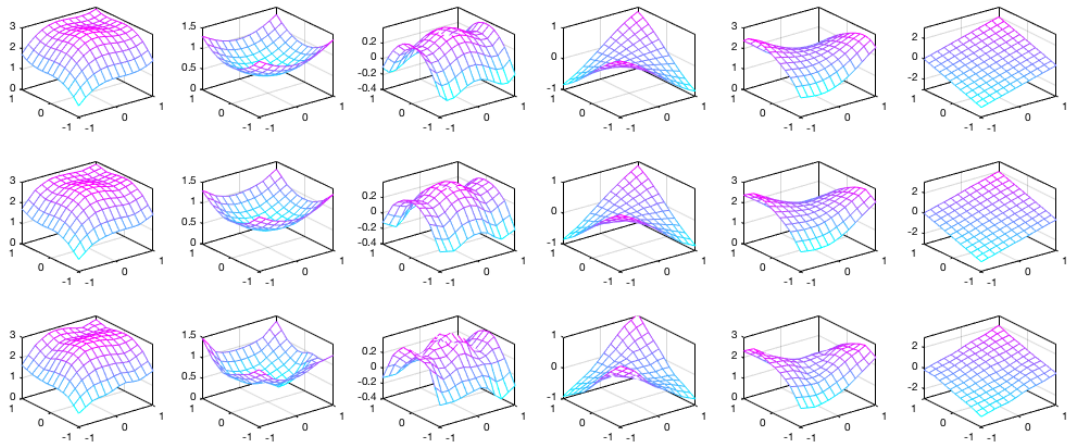


Figure 6.12: True surfaces versus estimated common features based on MSSR_{m} model: the top row displays the true surfaces of simulated by functions $f_1, \ldots, f_6$ uniformly over the rectangular domain $[-1, 1] \times [-1, 1]$, the middle row and bottom row are the corresponding MCMC fitted mean surfaces and online SMC fitted mean surfaces at $T = 5,000$ with $d = 6 \times 6$, $n.min = 40$.

6.11 Appendix D

6.11.1 Results of simulation study.

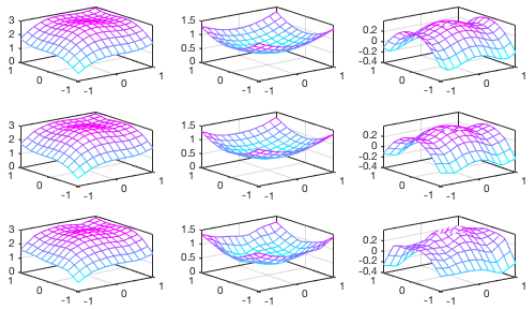


Figure 6.11: True surfaces versus estimated common features based on MSSR_{m} model: the top row displays the true surfaces of simulated by functions $f_1, \ldots, f_3$ uniformly over the rectangular domain $[-1, 1] \times [-1, 1]$, the middle row and bottom row are the corresponding MCMC fitted mean surfaces and online SMC fitted mean surfaces at $T = 5,000$ with $d = 6 \times 6$, $n.min = 20$.

As shown in Figure 6.11 -6.12, the estimated mean surfaces of proposed online SMC algorithm (common features) for each cluster as comparable as that of the MCMC algorithm.

As indicated in Figure 6.13, online SMC algorithm achieves similar performance as the MCMC algorithm in terms of $\xi_1^2, \xi_2^2, \xi_3^2$.

As indicated in Figure 6.14, online SMC algorithm achieves similar performance as the MCMC algorithm in terms of $\xi_1^2, \xi_2^2, \xi_3^2, \xi_4^2$. And there exists a small bias between the true value and the posterior mean provided by online SMC method for ξ_5^2, ξ_6^2 and the bias gets smaller as t increases.

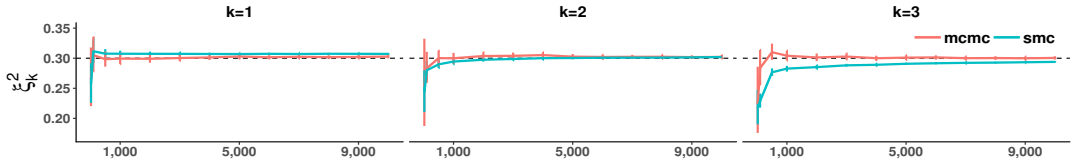


Figure 6.13: Estimated parameter of $\xi_k^2, k = 1, 2, 3$ with 95% equal tailed credible interval of online SMC algorithm versus MCMC algorithm when $K = 3$ and the number of observations increases from $t = 20$ to $10,000$.

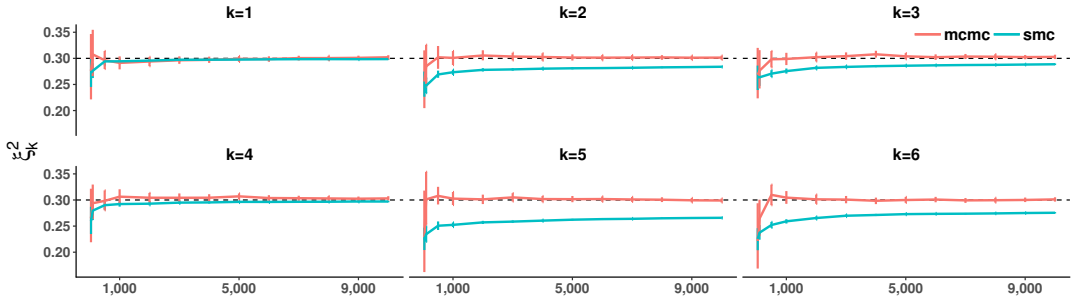


Figure 6.14: Estimated parameter $\xi_k^2, k = 1, \ldots, 6$ with 95% equal tailed credible interval of online SMC algorithm versus MCMC algorithm when $K = 6$ and the number of observations increases from $t = 40$ to $10,000$.

6.12 Appendix E

6.12.1 Real data analysis.

WAIC WAIC (Watanabe, 2010; Gelman et al., 2014) is often used to measure model predictive accuracy in Bayesian model and in our proposed online SMC algorithm, we can compute WAIC via Eq.(32) and computed at time T after we obtaining all the estimates of model parameters.

$$WAIC = -2 \sum_{t=1}^{T} logE_\theta[p(\boldsymbol{y}_t|\theta)] + 2 \sum_{t=1}^{T} V(log(p(\boldsymbol{y}_t|\boldsymbol{\theta}))). \tag{32}$$

In our proposed online SMC algorithm, we use $\sum_{i=1}^{N} p(\boldsymbol{y}_t|\boldsymbol{\theta}_T^{(i)})/N$ to approximate $E_{\boldsymbol{\theta}}[p(\boldsymbol{y}_t|\boldsymbol{\theta})]$ and $V(log(p(\boldsymbol{y}_t|\boldsymbol{\theta})))$ is approximated via follows

$$\frac{\sum_{i=1}^{N}(log(p(\boldsymbol{y}_t|\boldsymbol{\theta}_T^i)))^2 - \frac{(\sum_{i=1}^{N} log(p(\boldsymbol{y}_t|\boldsymbol{\theta}_T^{(i)})))^2}{N}}{N-1},$$

where $p(\boldsymbol{y}_t|\boldsymbol{\theta}_T^{(i)})$ is estimated by $\sum_{k=1}^{K} p(\boldsymbol{y}_t|\boldsymbol{\theta}_T^{(i)}, z_t = k)p(z_t = k)$.

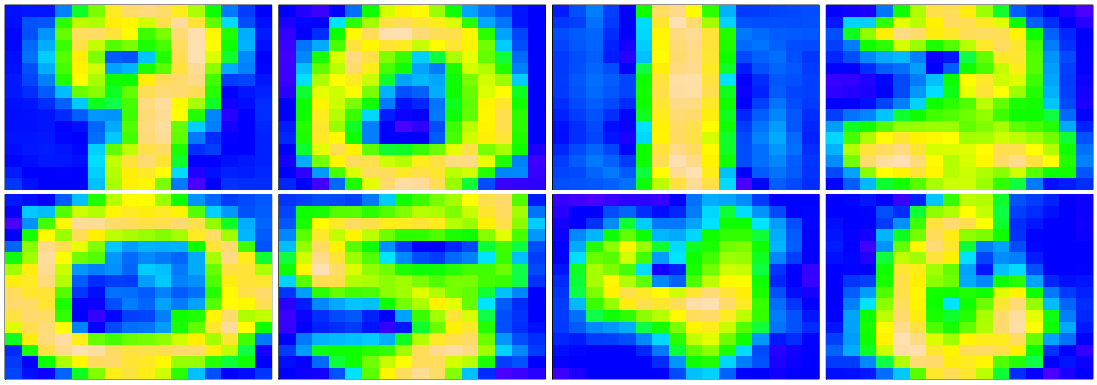


Figure 6.15: Online SMC estimated common features from MSSR_{m} model of handwritten number images by cluster when $K = 8$, $d = 8 \times 8$, labeled as the 1^{st} cluster to the 8^{th} cluster in left-to-right, top-to-bottom order.

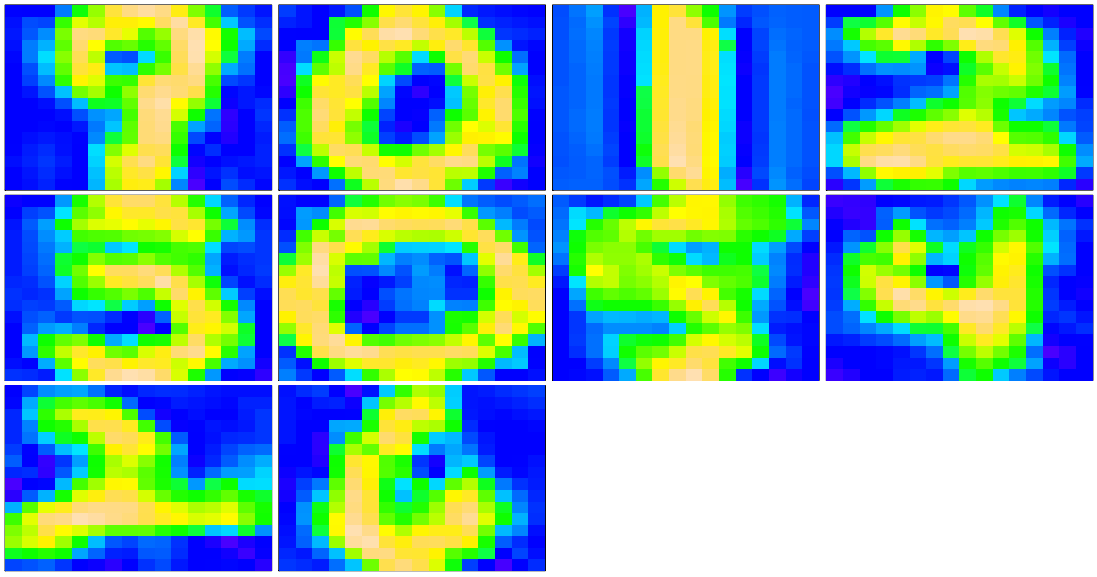


Figure 6.16: Online SMC estimated common features from MSSR_{m} model of handwritten number images by cluster when $K = 10$, $d = 8 \times 8$, labeled as the 1^{st} cluster to the 10^{th} cluster in left-to-right, top-to-bottom order

Handwritten number images As indicated in Figure 6.15 and Figure 6.16, when $K = 8$, the MSSR_{m} model is able to capture the common features of the most of the 10 digits as well as subgroup of 0, except $3, 8$, and as indicated by the 1^{st} cluster, $7, 9$ share some same common features. As indicated in Figure 6.16, when $K = 10$, the MSSR_{m} model is able to capture the common features for most of the 10 digits as well as subgroup of $0, 2$, except 3, and $7, 9$ (the 1^{st} cluster) share some common features, as well as 8 and 2 (the 9^{th} cluster).

Chapter 7

Discussion

In this section, we summarize this thesis work and provide some discussion of future work.

7.1 Summary

In this thesis, we have presented several novel statistical machine learning methods in computational genetics. Our first application gives the fastest known algorithms for computing the kinship coefficient given a known pedigree. Our proposed algorithms are applicable to large pedigrees, since they either have efficient polynomial or linear running-times. Our proposed algorithms are also applicable to the scenarios where the founders of the given pedigree are inbred and compute the appropriate inbreeding-adjusted kinship coefficients.

Secondly, we develop a Bayesian bivariate spatial group lasso model to jointly investigate the relationship between genetic variation and brain structure. Our proposed model is one of the first explicitly spatial hierarchical models for imaging genetics and neuroimaging to allow for both spatial correlation and bilateral cross-hemisphere correlation.

Thirdly, motivated by disease prediction via gene expression data, we propose a Bayesian nonparametric model, the Random Tessellation Process (RTP). The RTP provides a framework that includes the MP and the binary space partitioning-tree process as special cases. In addition, we provide random forest versions of the RTP.

Fourthly, we focus on Bayesian nonparametric methods for capturing shapes in images by partitioning space with curves. In this work, we propose a novel Bayesian nonparametric approach, Random Tessellation with Splines (RTS), to partition a domain with curves, which enables complex data shapes to be acquired. We develop a parallel algorithm for inference, which could decrease the computational time efficiently with multiple CPUs.

Finally, in Chapter 6, we extend the mixtures of spatial spline regression with mixed-effects model under the Bayesian framework to accommodate streaming image data. We propose a sequential Monte Carlo (SMC) algorithm to analyze online fashion image data. The existence of model sufficient statistics improves the efficiency of the proposed online SMC algorithm. The algorithm only requires storage of the model sufficient statistics and every data point is only used once.

7.2 Future work

All of the proposed algorithms in Chapter 2 have running times that are parameterized by the number of individuals. Edges in a pedigree graph corresponds to the kinship coefficients defined by inheritance paths, hence it is possible that the number of edges in pedigree graphs provides a true lower-bound on running-time of the computation of the kinship coefficients. We leave it as an open problem whether there is an efficient algorithm, which has the number of operations parameterized by the number of edges in the pedigree graph. Another open problem in computation of the kinship coefficients for a known pedigree is whether there is a sparse algorithm, which would only compute the non-zero entries in the kinship coefficient matrix. If designed properly, such algorithms would need far less space and far less number of operations, since it would not need to represent the entire kinship matrix.

In Chapter 3, we uses a shrinkage prior with group penalization for the coefficients of each SNP in the regression model. Within the current context, there are many future developments of interest. For example, using a group spike-and-slab prior as an alternative to the Bayesian group lasso prior. Our current methodology is best suited for situations where the targeted SNPs is relatively small and the number of ROIs is moderate. Extending the applicability of the methodology to settings with massive numbers of genetic and neuroimaging variables is an avenue for future work, such as combining variational Bayes with better optimization schemes, reduced rank regression and the use of parallel matrix computations, the use of divide and conquer strategies such as the consensus Monte Carlo algorithm (Scott et al., 2016) as well as splitting up the brain into a smaller number of sub-regions might lead to feasible Monte Carlo implementations for such settings.

Chapter 4 describes a framework for viewing Bayesian nonparametric methods based on space partitioning as Random Tessellation Processes, derives inference using sequential Monte Carlo and investigates the relationships between gene profiles and disease status. There are many extensions of RTPs of interest, such as improved SMC sampling for Markov jump processes (Hajiaghayi et al., 2014), hierarchical likelihoods and online methods (Lakshminarayanan et al., 2015), analysis of minimax convergence rates (Mourtada et al., 2018) and Bayesian additive regression trees as in Chipman et al. (2010); Lakshminarayanan et al. (2015). Moreover, we could apply RTPs to data that naturally displays tessellation and cracking, such as sea ice (Godlovitch, 2011).

Chapter 5 presents a novel Bayesian nonparametric method for estimating shapes within images and develops an inference algorithm that is “embarrassingly parallel". One future direction of this work is to extend the RTS to capture shapes of 3D objects (e.g. manifolds).

Chapter 6 introduces a new approach for online image clustering. There are several lines for our future work to improve surface clustering. First of all, one limitation of the current model is that we fix the number of clusters. It is of interest to automatically select the number of clusters K by model and data. One possible way to loosen this constraint is to treat K as a parameter, and introduce a Dirichlet process prior for K. Second, the covariance matrix of the random effects is proportional to an identity matrix. A more realistic assumption is to incorporate spatial correlation in the random

effects. For example, assuming the random effects arise from a Gaussian process with a suitable spatial covariance. Images for one specific cluster share common features. Another line of future work is to conduct clustering based on the common features by principle component analysis for image data.

Bibliography

M. Abney. A graphical algorithm for fast computation of identity coefficients and generalized kinship coefficients. *Bioinformatics*, 25(12):1561–1563, 2009.

R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk. Slic superpixels. Technical report, 2010.

R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk. SLIC superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2274–2282, 2012.

E. Alpaydin. *Introduction to Machine Learning*. MIT press, 2020.

P. Baldi and S. Brunak. *Bioinformatics: the Machine Learning Approach*. MIT press, 2001.

H. B. Barlow. Unsupervised learning. *Neural computation*, 1(3):295–311, 1989.

M. R. Barnes, J. Huxley-Jones, P. R Maycox, M. Lennon, A. Thornber, F. Kelly, S. Bates, A. Taylor, J. Reid, N. Jones, and J. Schroeder. Transcription and pathway analysis of the superior temporal cortex and anterior prefrontal cortex in schizophrenia. *Journal of Neuroscience Research*, 89(8): 1218–1227, 2011.

M. J. Beal et al. *Variational algorithms for approximate Bayesian inference*. university of London, 2003.

A. L. Beam and I. S. Kohane. Big data and machine learning in health care. *Journal of the American Medical Association*, 319(13):1317–1318, 2018.

M. A. Berger. *An Introduction to Probability and Stochastic Processes*. Springer Texts in Statistics, 2012.

A. Bhattacharya and D. B. Dunson. Nonparametric Bayesian density estimation on manifolds with applications to planar shapes. *Biometrika*, 97(4), 2010.

F. D. Bowman. Spatio-temporal modeling of localized brain activity. *Biostatistics*, 6(4):558–575, 2005.

F. D. Bowman, B. Caffo, S. S. Bassett, and C. Kilts. A bayesian hierarchical framework for spatial modeling of fMRI data. *NeuroImage*, 39:146–156, 2008.

L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Chapman and Hall/CRC, 1984.

C. M. Carvalho, M. S. Johannes, H. F. Lopes, N. G. Polson, et al. Particle learning and smoothing. *Statistical Science*, 25(1):88–106, 2010a.

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

P. Castellano, L. Prevedel, and A. A. Eugenin. HIV-infected macrophages and microglia that survive acute infection become viral reservoirs by a mechanism involving Bim. *Scientific Reports*, 7, 2017.

F. Chamroukhi. Bayesian mixtures of spatial spline regressions. *arXiv preprint arXiv:1508.00635*, 2015.

F. Chamroukhi and H. D. Nguyen. Model-based clustering and classification of functional data. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 9(4):e1298, 2019.

F. Chamroukhi, A. Samé, G. Govaert, and P. Aknin. A hidden process regression model for functional data description. Application to curve discrimination. *Neurocomputing*, 73(7-9):1210–1221, 2010.

H. A. Chipman, E. I. George, and R. E. McCulloch. BART: Bayesian additive regression trees. *Annals of Applied Statistics*, 4(1):266–298, 2010.

S. N. Chiu, D. Stoyan, W. S. Kendall, and J. Mecke. *Stochastic Geometry and its Applications*. Wiley Series in Probability and Statistics, 2013.

N. Chopin. Central limit theorem for sequential Monte Carlo methods and its application to Bayesian inference. *Annals of Statistics*, 32(6):2385–2411, 2004.

E. H. Corder, A. M. Saunders, W. J. Strittmatter, D. E. Schmechel, P. C. Gaskell, G. Small, A. D. Roses, J. L. Haines, and M. A. Pericak-Vance. Gene dose of apolipoprotein E type 4 allele and the risk of Alzheimer's disease in late onset families. *Science*, 261(5123):921–923, 1993.

A. L. Cunningham, H. Naif, N. Saksena, G. Lynch, J. Chang, S. Li, R. Jozwiak, M. Alali, B. Wang, and W. Fear. HIV infection of macrophages and pathogenesis of AIDS dementia complex: interaction of the host cell and viral genotype. *Journal of Leukocyte Biology*, 62(1):117–125, 1997.

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

G. Derado, F. D. Bowman, L. Zhang, and Alzheimer's Disease Neuroimaging Initiative. Predicting brain activity using a Bayesian spatial model. *Statistical Methods in Medical Research*, 22(4): 382–397, 2013.

J. Diebolt and C. P. Robert. Estimation of finite mixture distributions through Bayesian sampling. *Journal of the Royal Statistical Society. Series B (Methodological)*, 56(2):363–375, 1994.

A. Doucet, S. Godsill, and C. Andrieu. On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and Computing*, 10(3):197–208, 2000.

A. Doucet, N. De Freitas, and N. Gordon. An introduction to sequential Monte Carlo methods. In *Sequential Monte Carlo methods in practice*. Springer, 2001.

A. Doucet, M. Briers, and S. Sénécal. Efficient block sampling strategies for sequential Monte Carlo methods. *Journal of Computational and Graphical Statistics*, 15(3):693–711, 2006.

N. R. Draper and H. Smith. *Applied Regression Analysis*. John Wiley & Sons, 1998.

J. Duchon. Splines minimizing rotation-invariant semi-norms in Sobolev spaces. In *Constructive Theory of Functions of Several Variables*. Springer, 1977.

J. G. Dy and C. E. Brodley. Feature selection for unsupervised learning. *Journal of Machine Learning Research*, 5(Aug):845–889, 2004.

J. Eu-Ahsunthornwattana, E. N. Miller, M. Fakiola, S. M. B. Jeronimo, J. M. Blackwell, et al. Comparison of methods to account for relatedness in genome-wide association studies with family-based data. *PLOS Genetics*, 10(7):e1004445, 2014.

X. Fan, B. Li, Y. Wang, Y. Wang, and F. Chen. The Ostomachion process. In *Proceedings of the Thirtieth Conference of the Association for the Advancement of Artificial Intelligence*, 2016.

X. Fan, B. Li, and S. Sisson. The binary space partitioning-tree process. In *Proceedings of the 35th International Conference on Artificial Intelligence and Statistics*, 2018.

X. Fan, B. Li, and S. Sisson. Binary space partitioning forests. *arXiv preprint 1903.09348*, 2019.

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

P. Fearnhead and L. Meligkotsidou. Filtering methods for mixture models. *Journal of Computational and Graphical Statistics*, 16(3):586–607, 2007.

T. S. Ferguson. A Bayesian analysis of some nonparametric problems. *Annals of Statistics*, 1(2): 209–230, 1973.

B. Fischl. FreeSurfer. *NeuroImage*, 62(2):774–781, 2012.

W. A. Freije, F. E. Castro-Vargas, Z. Fang, S. Horvath, T. Cloughesy, L. M. Liau, P. S. Mischel, and S. F. Nelson. Gene expression profiling of gliomas strongly predicts survival. *Cancer Research*, 64(18):6503–6510, 2004.

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

J. Friedman, T. Hastie, and R. Tibshirani. *The Elements of Statistical Learning*, volume 1. Springer series in statistics, 2009.

N. Friel and A. N. Pettitt. Marginal likelihood estimation via power posteriors. *Journal of the Royal Statistical Society: Series B (Methodological)*, 70(3):589–607, 2008.

M. Gatz, C. A. Reynolds, L. Fratiglioni, B. Johansson, J. A. Mortimer, S. Berg, A. Fiske, and N. L. Pedersen. Role of genes and environments for explaining Alzheimer disease. *Archives of general psychiatry*, 63(2):168–174, 2006.

R. Gaudin. *2011 Olympus BioScapes Digital Imaging Competition® (2012) CIL:41568, Homo sapiens, macrophage. CIL. Dataset.* `https://doi.org/doi:10.7295/W9CIL41568`.

Y. Gazit, J. W. Baish, N. Safabakhsh, M. Leunig, L. T. Baxter, and R. K. Jain. Fractal characteristics of tumor vascular architecture during tumor growth and regression. *Microcirculation*, 4(4), 1997.

S. Ge, S. Wang, Y. W. Teh, L. Wang, and L. Elliott. Random tessellation forests. In *Advances in Neural Information Processing Systems*, 2019.

T. Ge, J. Feng, D. P. Hibar, P. M. Thompson, and T. E. Nichols. Increasing power for voxel-wise genome-wide association studies: the random field theory, least square kernel machines and fast permutation procedures. *NeuroImage*, 63:858–873, 2012.

A. E. Gelfand and S. Banerjee. Multivariate spatial process models. *Handbook of Spatial Statistics*, pages 495–515, 2010.

A. E. Gelfand and P. Vounatsou. Proper multivariate conditional autoregressive models for spatial data analysis. *Biostatistics*, 4(1):11–15, 2003.

A. Gelman and X. L. Meng. Simulating normalizing constants: From importance sampling to bridge sampling to path sampling. *Statistical Science*, pages 163–185, 1998.

A. Gelman, J. Hwang, and A. Vehtari. Understanding predictive information criteria for Bayesian models. *Statistics and Computing*, 24(6):997–1016, 2014.

E. I. George. Sampling random polygons. *Journal of Applied Probability*, 24(3):557–573, 1987.

P. Geurts, D. Ernst, and L. Wehenkel. Extremely randomized trees. *Machine Learning*, 63(1):3–42, 2006.

J. Geweke. Interpretation and inference in mixture models: Simple MCMC works. *Computational Statistics & Data Analysis*, 51(7):3529–3550, 2007.

W. R. Gilks and C. Berzuini. Following a moving target—Monte Carlo inference for dynamic Bayesian models. *Journal of the Royal Statistical Society: Series B (Methodological)*, 63(1): 127–146, 2001.

D. Godlovitch. *Idealised models of sea ice thickness dynamics*. PhD thesis, University of Victoria, 2011.

K. Greenlaw, E. Szefer, J. Graham, M. Lesperance, F. S. Nathoo, and Alzheimer's Disease Neuroimaging Initiative. A Bayesian group sparse multi-task regression model for imaging genetics. *Bioinformatics*, 33(16):2513–2522, 2017.

B. Grün and F. Leisch. Dealing with label switching in mixture models under genuine multimodality. *Journal of Multivariate Analysis*, 100(5):851–861, 2009.

K. Gu, D. Pati, and D. B. Dunson. Bayesian hierarchical modeling of simply connected 2d shapes. *arXiv preprint arXiv:1201.1658*, 2012.

M. Hajiaghayi, B. Kirkpatrick, L. Wang, and A. Bouchard-Côté. Efficient continuous-time Markov chain estimation. In *Proceedings of the 31st International Conference on Machine Learning*, 2014.

P. Hall, D. S. Poskitt, and B. Presnell. A functional data—analytic approach to signal discrimination. *Technometrics*, 43(1):1–9, 2001.

P. Halmos. *Measure Theory*. Springer, 1974.

L. A. Hannah and D. B. Dunson. Multivariate convex regression with adaptive partitioning. *Journal of Machine Learning Research*, 14(1):3261–3294, 2013.

D. P. Hibar, J. L. Stein, O. Kohannim, N. Jahanshad, A. J. Saykin, L. Shen, S. Kim, N. Pankratz, T. Foroud, M. J. Huentelman, et al. Voxelwise gene-wide association study (vGeneWAS): multivariate gene-based association testing in 731 elderly subjects. *NeuroImage*, 56:1875–1891, 2011.

D. P. Hibar, J. L. Stein, M. E. Renteria, A. Arias-Vasquez, S. Desrivières, N. Jahanshad, R. Toro, L. Wittfeld, K.and Abramovic, M. Andersson, et al. Common genetic variants influence human subcortical brain structures. *Nature*, 520(7546):224, 2015.

T. Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning*, 42 (1-2):177–196, 2001.

K. Hrecka, C. Hao, M. Gierszewska, S. K. Swanson, M. Kesik-Brodacka, S. Srivastava, L. Florens, M. P. Washburn, and J. Skowronski. Vpx relieves inhibition of HIV-1 infection of macrophages mediated by the SAMHD1 protein. *Nature*, 474(7353), 2011.

C. Huang, P. Thompson, Y. Wang, Y. Yu, J. Zhang, D. Kong, R. R. Colen, R. C. Knickmeyer, H. Zhu, Alzheimer's Disease Neuroimaging Initiative, et al. FGWAS: Functional genome wide association analysis. *NeuroImage*, 159:107–121, 2017.

M. Huang, T. Nichols, C. Huang, Y. Yu, Z. Lu, R. C. Knickmeyer, Q. Feng, H. Zhu, Alzheimer's Disease Neuroimaging Initiative, et al. FVGWAS: Fast voxelwise genome wide association analysis of large-scale imaging genetic data. *NeuroImage*, 118:613–627, 2015.

L. Hubert and . Arabie. Comparing partitions. *Journal of Classification*, 2(1):193–218, 1985.

B. Inkster, T. E. Nichols, P. G. Saemann, D. P. Auer, F. Holsboer, P. Muglia, and P. M. Matthews. Pathway-based approaches to imaging genetics association studies: Wnt signaling, GSK3beta substrates and major depression. *NeuroImage*, 53:908–917, 2010.

A. J. Izenman. Linear discriminant analysis. In *Modern multivariate statistical techniques*, pages 237–280. Springer, 2013.

A. Jacquard. Genetic information given by a relative. *Biometrics*, 28(4):1101–1114, 1972.

G. James, D. Witten, T. Hastie, and R. Tibshirani. *An introduction to Statistical Learning*. Springer, 2013.

G. M. James and T. J. Hastie. Functional linear discriminant analysis for irregularly sampled curves. *Journal of the Royal Statistical Society: Series B (Methodological)*, 63(3):533–550, 2001.

G. M. James and C. A. Sugar. Clustering for sparsely sampled functional data. *Journal of the American Statistical Association*, 98(462):397–408, 2003.

A. Jasra, C. C. Holmes, and D. A. Stephens. Markov chain Monte Carlo methods and the label switching problem in Bayesian mixture modeling. *Statistical Science*, 2005.

X. Jin, B. P. Carlin, and S. Banerjee. Generalized hierarchical multivariate CAR models for areal data. *Biometrics*, 61(4):950–961, 2005.

H. M. Kang et al. Variance component model to account for sample structure in genome-wide association studies. *Nature Genetics*, 42(4):348–354, 2010.

G. Karigl. A recursive algorithm for the calculation of identity coefficients. *Annals of Human Genetics*, 45(3):299, 1981.

C. Kemp, J. B. Tenenbaum, T. L. Griffiths, T. Yamada, and N. Ueda. Learning systems of concepts with an infinite relational model. In *Proceedings of the 20th Conference on the Association for the Advancement of Artificial Intelligence*, 2006.

A. E. Khandani, A. J. Kim, and A. W. Lo. Consumer credit-risk models via machine-learning algorithms. *Journal of Banking & Finance*, 34(11):2767–2787, 2010.

J. F. Kingman. *Poisson Processes*. Oxford University Press, 1996.

B. Kirkpatrick. Haplotype versus genotypes on pedigrees. *Algorithms for Molecular Biology*, 6(10): 136–147, 2011.

B. Kirkpatrick. Non-identifiable pedigrees and a Bayesian solution. *Int. Symp. on Bioinformatics Res. and Appl. (ISBRA)*, pages 139–152, 2012.

D. Kong, K. S. Giovanello, Y. Wang, W. Lin, E. Lee, Y. Fan, P. Murali Doraiswamy, and H. Zhu. Predicting Alzheimer's disease using combined imaging-whole genome SNP data. *Journal of Alzheimer's Disease*, 46(3):695–702, 2015.

D. Kong, B. An, J. Zhang, and H. Zhu. L2RM: Low-Rank Linear Regression Models for High-Dimensional Matrix Responses. *Journal of the American Statistical Association*, 115(529): 403–424, 2020.

H. Koppensteiner, R. Brack-Werner, and M. Schindler. Macrophages and their relevance in Human Immunodeficiency Virus Type I infection. *Retrovirology*, 9(1):1–11, 2012.

S. B. Kotsiantis, I. Zaharakis, and P. Pintelas. Supervised machine learning: a review of classification techniques. *Emerging Artificial Intelligence Applications in Computer Engineering*, 160:3–24, 2007.

S. Kurtek, A. Srivastava, E. Klassen, and Z. Ding. Statistical modeling of curves using shapes and related features. *Journal of the American Statistical Association*, 107(499):1152–1165, 2012.

M. Kyung, J. Gill, M. Ghosh, G. Casella, et al. Penalized regression, standard errors, and Bayesian lassos. *Bayesian Analysis*, 5(2):369–411, 2010.

B. Lakshminarayanan, D. M. Roy, and Y. W. Teh. Mondrian forests: Efficient online random forests. In *Proceedings of the 28th Conference on Neural Information Processing Systems*, 2014.

B. Lakshminarayanan, D. M. Roy, and Y. W. Teh. Particle Gibbs for Bayesian additive regression trees. In *Proceedings of the 18th International Conference on Artificial Intelligence and Statistics*, 2015.

A. L. Leutenegger, B. Prum, et al. Estimation of the inbreeding coefficient through use of genomic data. *The American Journal of Human Genetics*, 73(3):516–523, 2003.

A. Liaw and M. Wiener. Classification and Regression by `randomForest`. *R News*, 2(3):18–22, 2002.

L. Lin, N. Mu, P. Cheung, D. Dunson, et al. Extrinsic gaussian processes for regression and classification on manifolds. *Bayesian Analysis*, 14(3):907–926, 2019.

M. Lin, R. Chen, J. S. Liu, et al. Lookahead strategies for sequential Monte Carlo. *Statistical Science*, 28(1):69–94, 2013.

J. Liu and V. D. Calhoun. A review of multivariate analyses in imaging genetics. *Frontiers in Neuroinformatics*, 8:29, 2014.

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

Z. H. Lu, Z. Khondker, J. G. Ibrahim, Y. Wang, H. Zhu, Alzheimer's Disease Neuroimaging Initiative, et al. Bayesian longitudinal low-rank regression models for imaging genetic data from longitudinal studies. *NeuroImage*, 149:305–322, 2017.

A. Lucchi, K. Smith, R. Achanta, V. Lepetit, and P. Fua. A fully automated approach to segmentation of irregularly shaped cellular structures in EM images. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2010.

G. Lucotte, F. Loirat, and S. Hazout. Pattern of gradient of apolipoprotein E allele* 4 frequencies in western Europe. *Human Biology*, 1997.

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

Y. C. MacNab. Linear models of coregionalization for multivariate lattice data: a general framework for coregionalized multivariate CAR models. *Statistics in Medicine*, 35(21):3827–3850, 2016.

N. Malfait and J. O. Ramsay. The historical functional linear model. *Canadian Journal of Statistics*, 31(2):115–128, 2003.

J. M. Marin, k. Mengersen, and C. P. Robert. Bayesian modelling and inference on mixtures of distributions. *Handbook of statistics*, 25:459–507, 2005.

G. Matheron. Principles of geostatistics. *Economic Geology*, 58(8):1246–1266, 1963.

P. R. Maycox, F. Kelly, A. Taylor, S. Bates, J. Reid, R. Logendra, et al. Analysis of gene expression in two large schizophrenia cohorts identifies multiple changes associated with nerve terminal function. *Molecular Psychiatry*, 14(12):1083–1094, 2009.

J. E. Merrill and I. S. Chen. HIV-1, macrophages, glial cells, and cytokines in AIDS nervous system disease. *The FASEB Journal*, 5(10), 1991.

D. Meyer, E. Dimitriadou, K. Hornik, A. Weingessel, and F. Leisch. *e1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), Technische Universität Wien. R package version 1. 7–1*, 2019.

M. Mohri, A. Rostamizadeh, and A. Talwalkar. *Foundations of Machine Learning*. MIT press, 2018.

J. S. Morris, P. J. Brown, R. C. Herrick, K. A. Baggerly, and K. R. Coombes. Bayesian analysis of mass spectrometry proteomic data using wavelet-based functional mixed models. *Biometrics*, 64 (2):479–489, 2008.

M. E. Mortenson. *Mathematics for Computer Graphics Applications*. Industrial Press Inc., 1999.

J. Mourtada, S. Gaïffas, and E. Scornet. Minimax optimal rates for Mondrian trees and forests. *arXiv preprint 1803.05784*, 2018.

H. Muller. Surface reconstruction-an introduction. In *Scientific Visualization Conference (dagstuhl'97)*. IEEE, 1997.

W. Nagel and V. Weiss. Crack STIT tessellations: Characterization of stationary random tessellations stable with respect to iteration. *Advances in Applied Probability*, 37(4):859–883, 2005.

F. S. Nathoo, M. L. Lesperance, A. B. Lawson, and C. B. Dean. Comparing variational Bayes with Markov chain Monte Carlo for Bayesian computation in neuroimaging. *Statistical Methods in Medical Research*, 22(4):398–423, 2013.

F. S. Nathoo, A. Babul, A. Moiseev, N. Virji-Babul, and M. F. Beg. A variational Bayes spatiotemporal model for electromagnetic brain mapping. *Biometrics*, 70(1):132–143, 2014.

F. S. Nathoo, L. Kong, and H. Zhu. A review of statistical methods in imaging genetics. *Canadian Journal of Statistics*, 47(1):108–131, 2019.

H. D. Nguyen, G. J. McLachlan, and I. A. Wood. Mixtures of spatial spline regressions for clustering and classification. *Computational Statistics & Data Analysis*, 93:76–85, 2016.

J. T. Ormerod and M. P. Wand. Explaining variational approximations. *The American Statistician*, 64(2):140–153, 2010.

T. Park and G. Casella. The Bayesian LASSO. *Journal of the American Statistical Association*, 103 (482):681–686, 2008.

W. D Penny, N. J Trujillo-Barreto, and K. J. Friston. Bayesian fMRI time series analysis with spatial priors. *NeuroImage*, 24:350–362, 2005.

A. Piccolboni and D. Gusfield. On the complexity of fundamental computational problems in pedigree analysis. *Journal of Computational Biology*, 10(5):763–773, 2003.

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

K. Puolamäki and S. Kaski. Bayesian solutions to the label switching problem. In *International Symposium on Intelligent Data Analysis*. Springer, 2009.

S. Purcell, B. Neale, K. Todd-Brown, L. Thomas, et al. PLINK: a tool set for whole-genome association and population-based linkage analyses. *The American journal of human genetics*, 81 (3):559–575, 2007.

T. Rainforth and F. Wood. Canonical correlation forests. *arXiv preprint 1507.05444*, 2015.

C. S. Rakovski and D. O. Stram. A kinship-based modification of the armitage trend test to address hidden population structure and small differential genotyping errors. *PLOS ONE*, 4(6):e5825, 06 2009.

Q. Ren, S. Banerjee, A. O. Finley, and J. S. Hodges. Variational Bayesian methods for spatial data analysis. *Computational Statistics & Data Analysis*, 55(12):3197–3217, 2011.

X. Ren and J. Malik. Learning a classification model for segmentation. In *Proceedings Ninth IEEE International Conference on Computer Vision*. IEEE, 2003.

C. Robert and G. Casella. *Monte Carlo Statistical Methods*. Springer Science & Business Media, 2013.

D. M. Roy and Y. W. Teh. The Mondrian process. In *Proceedings of the 22nd Conference on Neural Information Processing Systems*, 2008.

S. Russell and P. Norvig. Artificial intelligence: a modern approach. 2002.

L. M. Sangalli, J. O. Ramsay, and T. O. Ramsay. Spatial spline regression models. *Journal of the Royal Statistical Society: Series B (Methodological)*, 75(4):681–703, 2013.

S. L. Scott, A. W. Blocker, F. V. Bonassi, H. A. Chipman, E. I. George, and R. E. McCulloch. Bayes and big data: The consensus Monte Carlo algorithm. *International Journal of Management Science and Engineering Management*, 11(2):78–88, 2016.

X. Shen, X. Papademetris, and R. T. Constable. Graph-theory based parcellation of functional subunits in the brain from resting-state fMRI data. *NeuroImage*, 50:1027–1035, 2010.

M. Silver, G. Montana, T. E. Nichols, Alzheimer's Disease Neuroimaging Initiative, et al. False positives in neuroimaging genetics using voxel-based morphometry data. *NeuroImage*, 54:992–1000, 2011.

B. D. Sivazlian. On a multivariate extension of the gamma and beta distributions. *SIAM Journal on Applied Mathematics*, 41(2), 1981.

J. L. Stein, X. Hua, S. Lee, A. J. Ho, A. D. Leow, A. W. Toga, A. J. Saykin, L. Shen, T. Foroud, N. Pankratz, et al. Voxelwise genome-wide association study (vGWAS). *NeuroImage*, 53: 1160–1174, 2010.

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

F. C. Stingo, M. Guindani, M. Vannucci, and V. D. Calhoun. An integrative Bayesian modeling approach to imaging genetics. *Journal of the American Statistical Association*, 108(503):876–891, 2013.

B. Su and D. Liu. *Computational Geometry: Curve and Surface Modeling*. Elsevier, 2014.

D. Sun, Z. Wang, and S. Yu. Kinship accuracy: Comparing algorithms for large pedigrees. *Stanford Undergraduate Research Journal*, 13:97–102, 2014.

E. Szefer, D. Lu, F. Nathoo, M. F. Beg, J. Graham, et al. Multivariate association between single-nucleotide polymorphisms in Alzgene linkage regions and structural changes in the brain: discovery, refinement and validation. *Statistical Applications in Genetics and Molecular Biology*, 16(5-6): 367–386, 2017.

A. L. Tarca, V. J. Carey, X.W. Chen, R. Romero, and S. Drăghici. Machine learning and its applications to biology. *PLOS Computational Biology*, 3(6), 2007.

M. Teng, T. D. Johnson, and F. S. Nathoo. Time series analysis of fMRI data: Spatial modelling and Bayesian computation. *Statistics in Medicine*, 37(18):2753–2770, 2018.

M. Teng, F. S. Nathoo, and T. D. Johnson. Bayesian analysis of functional magnetic resonance imaging data with spatially varying auto-regressive orders. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 68(3):521–541, 2019.

E. A. Thompson. *Pedigree Analysis in Human Genetics*. Johns Hopkins University Press, Baltimore, 1985.

P. M. Thompson, T. Ge, D. C. Glahn, N. Jahanshad, and T. E. Nichols. Genetics of the Connectome. *NeuroImage*, 80:475–488, 2013.

T. Thornton and M. S. McPeek. Case-control association testing with related individuals: A more powerful quasi-likelihood score test. *The American Journal of Human Genetics*, 81:321–337, 2007.

T. M. Tomita, J. Browne, C. Shen, J. L. Patsolic, J. Yim, C. E. Priebe, R. Burns, M. Maggioni, and J. T. Vogelstein. Random projection forests. *arXiv preprint 1506.03410*, 2015.

A. Vehtari, A. Gelman, and J. Gabry. Practical Bayesian model evaluation using leave-one-out cross-validation and WAIC. *Statistics and Computing*, 27(5):1413–1432, 2017.

M. Vounou, T. E. Nichols, G. Montana, Alzheimer's Disease Neuroimaging Initiative, et al. Discovering genetic associations with high-dimensional neuroimaging phenotypes: A sparse reduced-rank regression approach. *NeuroImage*, 53:1147–1159, 2010.

J. Wakefield. A Bayesian measure of the probability of false discovery in genetic epidemiology studies. *The American Journal of Human Genetics*, 81(2):208–227, 2007.

H. Wang, F. Nie, H. Huang, S. L. Risacher, A. J. Saykin, L. Shen, and Alzheimer's Disease Neuroimaging Initiative. Identifying disease sensitive and quantitative trait-relevant biomarkers from multidimensional heterogeneous imaging genetics data via sparse multimodal multitask learning. *Bioinformatics*, 28(12):127–136, 2012.

S. Wang. *Advanced Monte Carlo methods and applications*. PhD thesis, Simon Fraser University, 2019.

Y. C. Wang, M. Burke, and R. E. Kraut. Gender, topic, and audience response: an analysis of user-generated content on facebook. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2013.

S. Watanabe. Asymptotic equivalence of Bayes cross validation and widely applicable information criterion in singular learning theory. *Journal of Machine Learning Research*, 11(Dec):3571–3594, 2010.

S. N. Wood, M. V. Bravington, and S. L. Hedley. Soap film smoothing. *Journal of the Royal Statistical Society: Series B (Methodological)*, 70(5):931–955, 2008.

C. G. Woods, J. Cox, et al. Quantification of homozygosity in consanguineous individuals with autosomal recessive disease. *The American Journal of Human Genetics*, 78:889–896, 2006.

M. Yang and R. Modarres. Multivariate tests of uniformity. *Statistical Papers*, 58(3):627–639, 2017.

J. Yu et al. A unified mixed-model method for association mapping that accounts for multiple levels of relatedness. *Nature Genetics*, 38:203 – 208, 2005.

Q. Zheng and C. Bourgain. KinInbcoef: Calculation of kinship and inbreeding coefficients, 2009.

B. Zhou. Image segmentation using slic superpixels and affinity propagation clustering. *Int. J. of Science and Research*, 4(4):1525–1529, 2015.

H. Zhu, Z. Khondker, Z. Lu, and J. G. Ibrahim. Bayesian generalized low rank regression models for neuroimaging phenotypes and genetic markers. *Journal of the American Statistical Association*, 109(507):977–990, 2014.