Clustering Profiles in Generalized Linear Mixed Models Settings using Bayesian Nonparametric Statistics

by

Predrag Mizdrak

A thesis submitted to the

Faculty of Graduate and Postdoctoral Affairs

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Mathematics and Statistics

Ottawa-Carleton Institute for Mathematics and Statistics

Department of School of Mathematics and Statistics

Carleton University

Ottawa, Ontario

August, 2018

Abstract

Generalized linear mixed models are used to model clustered and longitudinal data in which the distribution of the response variable is a member of the exponential family. This thesis introduces a novel method for simultaneous clustering of such data and estimation of parameters of the underlying generalized linear mixed models.

Clustering has been extensively studied for both cross-sectional and longitudinal data. In longitudinal data, one has to take into account the association between observations taken on the same individual. This has found applications in epidemiology, genetics, biology, market research, economics, and many other areas.

Generalized linear mixed models consist of two sets of parameters: fixed effects parameters that associate covariates to the response at the population level, and random effects parameters that associate covariates to the response at the individual level. We introduce a method that identifies homogeneous groups in the data based on similarities among random effects parameters that are obtained when homogeneous groups are modeled using generalized linear mixed models. We achieve this by placing a Dirichlet Process prior on random effects parameters, which induces clustering of random effects and subsequently the clustering of profiles. As a result, our method simultaneously groups profiles into clusters and estimates model

parameters of each cluster without assuming that the number of clusters is known in advance . The fixed effects parameters are shared by all clusters. However, each cluster has its own random effects parameter that is shared by all profiles in it.

We have tested our method on both simulated data and data from public health domain. In simulations, we have shown that the method manages to recover the correct number of clusters, successfully clusters profiles and correctly estimates model parameters. In public health clustered data, our method produces parameter estimates that are very close to those obtained by a frequentist maximum likelihood method, while identifying groups of homogeneous health regions that reveal certain properties of the underlying survey population that cannot be easily obtained using other methods.

Similar methods have been proposed for longitudinal data with continuous responses. This thesis extends these models in novel ways to clustered and longitudinal data, where the distribution of the response variable can be any member of the exponential family.

Acknowledgments

First and foremost, I owe an immense gratitude to my thesis supervisor, Prof. Sanjoy K. Sinha, whose guidance, support, motivation, expertise and patience has made the work on thesis possible. Second, special thanks go to Dr. Karla Fox for numerous suggestions that have made significant improvements to this thesis. And finally, to my family and friends, who have been there for me from day one, I say: Thank you from the bottom of my heart.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

In this chapter, we first introduce the basic concepts of longitudinal data as a special type of clustered data. We then introduce the clustering of profiles, where we define a profile as a collection of correlated variables. These could be profiles in longitudinal data or clusters in clustered data. Following that we present our statement of problems, and outline how the rest of the thesis is organized.

1.1 Longitudinal Studies

A longitudinal study is a type of study in which a set of units (or individuals) are followed for a period of time, and multiple observations are recorded on each individual. Different numbers of observations may be taken on each individual, and the time at which those observations are taken may be different across individuals.

The defining characteristic of longitudinal studies is that they allow the study of change in response over time, with potentially some covariates (other than time) also changing. This within-subject change can only be captured by longitudinal studies. Observations recorded from the same individuals are not independent, and

this association needs to be taken into account in order to produce valid results.

Observations taken from the same individual may be considered as belonging to the same cluster. We can say that longitudinal data are a type of clustered data, with the main difference being that the observations in longitudinal data must have temporal order while observations in clustered data need not be ordered. In fact, longitudinal data is the most common type of clustered data. Our method applies to all types of clustered data, though for clarity we will address in most chapters a longitudinal problem.

There are two types of longitudinal models [2]: marginal models and mixed-effects models. Marginal models capture population-averaged mean structure, while mixed effects models capture conditional mean structure (conditional on the subject-specific random effect). The models considered in this thesis are mixed-effects models.

We next describe the notation that we use throughout this thesis. Let $\boldsymbol{Y}$ be the response variable, and let $\boldsymbol{X}$ and $\boldsymbol{Z}$ be a vector or a matrix of covariates associated with fixed and random effects, respectively. We denote fixed effects parameters by $\boldsymbol{\beta}$ and random effects associated with an individual i by $\boldsymbol{b}_i$. The general form of our model is $E(\boldsymbol{Y}|\boldsymbol{X},\boldsymbol{Z}) = f(\boldsymbol{X\beta} + \boldsymbol{Zb})$, where $f(\cdot)$ is a given function (identity function in case of linear mixed models or an inverse of a suitable link function in generalized linear mixed models). Let y_{ij} denote the j^{th} observation of the i^{th} individual, $i = 1, 2, \ldots, N$ and $j = 1, 2, \ldots, n$, and let $\boldsymbol{y}_i = (y_{i1}, y_{i2}, \ldots y_{in})^t$ be the vector of all observations from individual i. Then $\boldsymbol{y} = (\boldsymbol{y}_1^t, \boldsymbol{y}_2^t, \ldots, \boldsymbol{y}_N^t)^t$ is the vector of all observations in the study. Let $\boldsymbol{X}_{ij}$ be a $p \times 1$ vector of covariates linking the fixed effect parameter $\boldsymbol{\beta}$ to y_{ij}, and let $\boldsymbol{X}_i$ be a matrix, of $n \times p$ size,

of all covariates associated with the individual i. Similarly, let $\boldsymbol{Z}_{ij}$ denote a $q \times 1$ vector of covariates linking the random effect parameters $\boldsymbol{b}_i$ to y_{ij}, and let $\boldsymbol{Z}_i$ be a matrix of $n \times q$ size of all covariates associated with an individual i. Then $\boldsymbol{X}$ and $\boldsymbol{Z}$ are matrices of all covariate linking fixed and random effects parameters of individuals to response variable. The mean structure of a linear mixed effects model is $E(\boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{Z}) = \boldsymbol{X}^t\boldsymbol{\beta} + \boldsymbol{Z}^t\boldsymbol{b}$, where $\boldsymbol{b}$ is the vector of all random effects parameters for N individuals. i.e., $\boldsymbol{b} = (\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_N)^t$.

1.2 Grouping or Clustering Data

Clustering has been an active area of research in both computer science and statistics for a long time. It is a process of grouping similar objects so that objects in the same group are as similar (homogeneous) as possible, while objects in different groups are as dissimilar (heterogeneous) as possible. There are different ways of classifying clustering methods. We can classify them as distance-based methods, in which similarity between objects is determined using some distance-based function, and model-based clustering methods, which assume that the data have been generated by some probabilistic model.

Many, especially earlier, methods were distance-based, such as [3] and [4], with some of them extended to specifically accommodate longitudinal data [5], [6]. This of course makes sense as it is the most common type of clustered data.

Model-based clustering methods assume that the population from which the data is sampled is heterogeneous, and that it can be modeled using a mixture

model [7] consisting of a finite number of mixture components, where each mixture component models a homogeneous subpopulation. A model of a mixture component can be as simple as a distribution (most often a multivariate normal distribution), or as complex as any other statistical model may be. In a simple case when a model of a mixture component is multivariate normal, clustering often proceeds by decomposing the covariance matrix and imposing certain constraints on the new matrices. Common decompositions include the modified Cholesky decomposition [8], [9] and the eigen-decomposition of the matrix [10], [11].

Methods in which a number of clusters is not known in advance often fit more than one model, using different numbers of clusters, and then perform model comparison to find a model that is most plausible given the observed data.

Methods based on Bayesian nonparametric statistics do not assume that the number of clusters is known in advance. Recently, a few methods have been proposed for clustering longitudinal data in which the response variable is continuous and therefore can be modeled using the mixture of linear mixed models [12], [13], [14]. In these methods, the data are clustered as a consequence of putting Dirichlet Process [15] prior on unit-specific parameters, which induces clustering of unit-specific parameters, and by that, also clustering of units and their respective profiles. We extend these methods to clustered data in which the response variable has a distribution that is a member of the exponential family and the model contains both fixed and random effects parameters.

1.3 Statement of the Problems

In this thesis, we propose a novel method that we call the Generalized linear mixed model clustering using Dirichlet Process (GLMM-DP), that allows one to simultaneously cluster profiles in generalized linear mixed models and estimate parameters of such models. The method allows the complexity of the model to grow as more data become available, without over-fitting the model. It differs from the existing methods in that it is the first method of its kind that allows one to cluster profiles in which the distribution of the response variable can be any member of the exponential family, while the existing methods allow clustering of profiles with only continuous outcome.

There are many cases when the entirety of a data set is heterogeneous, and the number of homogeneous groups contained within it is not known in advance. In many cases, identification of homogeneous groups is the first step in the study, followed by model building at some later time. This is done in practice because a model built on homogeneous subgroups of data should be more parsimonious since units in an homogeneous group exhibit more similar behavior as compared to those in a heterogeneous group.

Although most analyses include both steps, researchers often treat these two steps independently. In this thesis we approach the two problems simultaneously. First, we derive technical details of the GLMM-DP method in a case where a single univariate response is recorded for each unit, possibly at different time points. Then, we evaluate our method with a simulated data set using two models: one with a continuous response and another with a count response.

Next, we apply the GLMM-DP method to a real data set and compare the results to those obtained from an existing method in a frequentist domain. We show that the GLMM-DP method, when used alone or in combination with other methods, can provide insight into the data that may not be available otherwise. Finally, we extend the GLMM-DP method to multivariate setting, where more than one outcome may be observed at each time point, and evaluate it with a simulated data set.

1.4 Organization of Thesis

This thesis is organized as follows. Because of combination of Bayesian and classical statistics that our work is based on, the first three chapters provide background information for the reader. In Chapter 2, we review the basic ideas behind linear models, generalized linear models, linear mixed models and generalized linear mixed models, from the perspective of their use and application in longitudinal studies. Next, in Chapter 3, we review the basic concepts and ideas in Bayesian nonparametric statistics, and approximation methods of estimating the posterior distribution of model parameters. We review the rationale and advantages behind the Bayesian nonparametric statistics, with the focus on Dirichlet Process and Dirichlet Process mixture models.

The mathematical approach devised in this thesis is contained in Chapter 4, where we present the details of the GLMM-DP method. We derive the details of the method, including specification of its prior and posterior distributions of the model parameters. We also describe a label switching solution that is necessary in

order to be able to perform inference on a mixture component level. Without this solution, it is impossible to correctly identify cluster parameters. Next, we evaluate the GLMM-DP method on simulated data in Chapter 5, using two models: one with a continuous response and another with count response. We simulate a number of different scenarios with varying values of mixed effects parameters and numbers of observations recorded on each individual.

In Chapter 6, we apply the GLMM-DP method to a public health survey data set. We compare the results to those obtained from a frequentist (glmmML) method. We obtain very similar parameter estimates. However, in addition to obtaining results that are very similar to those obtained from a frequentist method, we show that the GLMM-DP method also identifies three distinct clusters in the data, which reveal certain properties of the underlying survey population that could not be obtained using only ordinary ML method. Then, we extend the GLMM-DP method to a multivariate settings in Chapter 7 and evaluate its performance on a simulated data set in which the response variable consists of two outcomes: one of continuous type and the other of count type. Finally, we close this thesis with Chapter 8 in which we summarize our conclusions and provide some advice for future research.

Chapter 2

Overview of Linear and Generalized Linear Models

In this chapter we first introduce general linear models and generalized linear models. Next, we describe general linear mixed models as an extension of general linear models. Finally, we introduce generalized linear mixed models by comparing them to previous models and outlining properties in which they differ.

We present these models only as they apply in the study of longitudinal data. Similar models that may have been used historically in analyzing longitudinal data (due to computational convenience), such as ANOVA, are not discussed in this chapter. The main reason being is that ANOVA imposes unrealistic restrictions on longitudinal data and clustered data in general. These include oversimplification of covariance matrices, enforcing the same covariance matrix on all units, and/or inability to handle unbalanced or incomplete data [2].

2.1 General Linear Models

General linear models are statistical models in which the mean of the response variable Y is assumed to be a linear function of parameters β, which represent the effects of predictors X. If $\boldsymbol{Y}_i$ is a vector of n observations of an individual i, $\boldsymbol{Y}_i = (Y_{i1}, Y_{i2}, \ldots, Y_{in})^t$, and $\boldsymbol{X}_i$ is a matrix of corresponding covariates, then the model may be written as

$$\boldsymbol{Y}_i = \boldsymbol{X}_i\boldsymbol{\beta} + \boldsymbol{\epsilon}_i, \tag{2.1}$$

where $\boldsymbol{\epsilon}_i = (\epsilon_1, \epsilon_2, \ldots, \epsilon_n)^t$ is a vector of random errors in the regression model.

The response vector $\boldsymbol{Y}_i$ is said to consist of two components: the systematic component ($\boldsymbol{X}_i\boldsymbol{\beta}$) which describes linear relationships between the response variable and the predictors, and the random component ($\boldsymbol{\epsilon}_i$), which specifies the distribution of the error and hence the distribution of the response vector $\boldsymbol{Y}_i$. We assume that the response vector $\boldsymbol{Y}_i$ is distributed according to a multivariate normal distribution with the mean $\boldsymbol{\mu}_i$ and covariance matrix $\boldsymbol{\Sigma}_i$, which is written as $\boldsymbol{Y}_i \sim \text{MVN}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$, where

$$\boldsymbol{\mu}_i = E(\boldsymbol{Y}_i|\boldsymbol{X}_i) = \boldsymbol{X}_i\boldsymbol{\beta} \tag{2.2}$$

and

$$\boldsymbol{\Sigma}_i = \text{Cov}(\boldsymbol{Y}_i|\boldsymbol{X}_i). \tag{2.3}$$

In longitudinal studies, observations from the same individual are typically not independent. The dependence between observations may be captured by the above covariance matrix $\boldsymbol{\Sigma}_i$, in which the off-diagonal elements would not be equal to zero. The case where all off-diagonal entries are equal to zero would correspond to independent observations from the same individual. This is very unlikely in longitudinal

studies. In general, the covariance matrix $\boldsymbol{\Sigma}_i$ embeds different association levels among observations from the same individual (note its index i), and in general, it may depend on covariates $\boldsymbol{X}_i$.

Here, the model consists of two parameters: $\boldsymbol{\beta}$ and $\boldsymbol{\Sigma}$, where $\boldsymbol{\beta}$ is generally the main focus of the study, while $\boldsymbol{\Sigma}$ is often considered a "nuisance" parameter - we need to take it into account in order to produce valid inferences but may not be interested in its estimates.

Two common ways of modeling the covariance matrix are: 1) by explicitly specifying the covariance matrix [16] (such as the compound symmetry, Toeplitz covariance pattern, autoregressive, banded, exponential or other); or 2) by introducing random effects parameters which induce a covariance structure in the model. The models discussed in this thesis have their covariance structure induced by random effects parameters.

When the model is fully specified, the most natural estimation method for the model parameters is the maximum likelihood method. Here, the likelihood of the model parameters, given the observed data set consisting of n dependent observations on each of N units, is given by

$$L(\boldsymbol{\beta}, \boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_N | \boldsymbol{y}_1, \ldots, \boldsymbol{y}_N) = \prod_{i=1}^{N} f(\boldsymbol{y_i} | \boldsymbol{\beta}, \boldsymbol{\Sigma}_i) \tag{2.4}$$

where

$$f(\boldsymbol{y_i} | \boldsymbol{\beta}, \boldsymbol{\Sigma}_i) = \frac{1}{\sqrt{(2\pi)^n |\boldsymbol{\Sigma}_i|}} \exp\left\{ -\frac{1}{2} (\boldsymbol{y}_i - \boldsymbol{\mu}_i)^t \boldsymbol{\Sigma}_i^{-1} (\boldsymbol{y}_i - \boldsymbol{\mu}_i) \right\}. \tag{2.5}$$

In this case, it can be seen that the dependence between observations from the same individual are captured by multivariate normal distribution.

The maximum likelihood estimates of $\boldsymbol{\beta}$ and $\boldsymbol{\Sigma}_i$ are obtained by maximizing the log-likelihood of the data (Eq.(2.4)). In these models, the covariance matrix $\boldsymbol{\Sigma_i}$ is often a function of some parameters $\boldsymbol{\theta}$, called the dispersion parameters, $\boldsymbol{\Sigma}_i \equiv \boldsymbol{\Sigma}_i(\boldsymbol{\theta})$, and is estimated by first estimating the parameters $\boldsymbol{\theta}$ and then plugging their estimates into the matrix $\boldsymbol{\Sigma}_i$, i.e., $\hat{\boldsymbol{\Sigma}}_i = \boldsymbol{\Sigma}_i(\hat{\boldsymbol{\beta}})$. The estimates of $\boldsymbol{\theta}$ are obtained numerically.
Then in these parametric models, the maximum likelihood estimate of $\boldsymbol{\beta}$ is given by

$$\hat{\boldsymbol{\beta}} = \left\{ \sum_{i=1}^{N} \left(\boldsymbol{X}_i \hat{\boldsymbol{\Sigma}}_i^{-1} \boldsymbol{X}_i \right) \right\}^{-1} \sum_{i=1}^{N} \left(\boldsymbol{X}_i \hat{\boldsymbol{\Sigma}}_i^{-1} \boldsymbol{y}_i \right), \tag{2.6}$$

and is an unbiased estimate of $\boldsymbol{\beta}$.
The asymptotic distribution of $\hat{\boldsymbol{\beta}}$ is multivariate normal with the mean being the true value of $\boldsymbol{\beta}$ and the covariance given by

$$\mathrm{Cov}(\hat{\boldsymbol{\beta}}) = \left\{ \sum_{i=1}^{N} \left(\boldsymbol{X}_i \hat{\boldsymbol{\Sigma}}_i^{-1} \boldsymbol{X}_i \right) \right\}^{-1}. \tag{2.7}$$

It has been shown, however, that Eq.(2.6) is a valid estimate of $\boldsymbol{\beta}$ even when the normality assumption does not hold. This is true as long as the data is complete [2].

In longitudinal studies in particular, the main disadvantage of the maximum likelihood method is that it produces biased estimate of the covariance matrix $\boldsymbol{\Sigma}_i$. This shortcoming may be addressed using the restricted maximum likelihood method [17], which produces unbiased estimates of both $\boldsymbol{\beta}$ and $\boldsymbol{\Sigma}_i$.
As with other parametric methods, once the estimates of $\boldsymbol{\beta}$ and $\boldsymbol{\Sigma}_i$ are obtained, the inference on $\boldsymbol{\beta}$ often proceeds with the Wald test or the likelihood ratio test [18].

2.2 Generalized Linear Models

General linear models, introduced in the previous sections, are models that are based on three core assumptions: independence of responses, linearity of responses relative to regression parameters, and a constant variance of the error term. Generalized linear models ([19], [20]) relax the last two assumptions in the following way: (1) the mean of the response is not a linear function of regression parameters but rather, it is associated with a linear function of parameters through some non-linear function, called the link function; and (2) the variance of response is not constant but is now a function of the mean. The independence assumption remains in both type of models. That is, neither model is suited for problems with multicollinearity or clustered data.

Linear models assume independence between responses. Multiple responses from the same unit could be stacked up into a vector and its distribution (in case of normal) would be completely defined. This is not the case with generalized linear models: univariate distributions of vector components do not translate to equivalent multivariate distribution (other than in case of normal). This makes traditional generalized linear models not well suited for the analysis of longitudinal data. We consider an extension of generalized linear models later in this chapter that addresses this difficulty. In the remainder of this section, we introduce the exponential family of distributions and then mention some of the basic concepts from generalized linear models.

2.2.1 Exponential family of distributions

A univariate random variable Y is said to have a distribution that is a member of an exponential family of distributions [18], if its distribution may be written in the

following form

$$f(y;\theta,\nu) = \exp\left\{\frac{y\theta - q(\theta)}{\nu} + k(y,\nu)\right\}, \tag{2.8}$$

where θ is the canonical parameter of the distribution, $q(\cdot)$ and $k(\cdot)$ are distribution specific functions. Exponential family of distributions includes both continuous distributions(i.e. normal, gamma, beta, Dirichlet, chi-squared), and discrete distributions(i.e. binomial, negative binomial, multinomial, Poisson, geometric). For example, the Poisson distribution with mean μ may be represented in the above form with $\theta = \log(\mu)$, $q(\theta) = \exp(\theta)$, and $\nu = 1$, while for the normal distribution with mean μ and variance σ^2, the exponential representation would have $\theta = \mu$, $q(\theta) = \frac{\theta^2}{2}$ and $\nu = \sigma^2$.

Using the maximum likelihood theory, we obtain general expressions for both mean and variance of Y. In fact,

$$E(Y) = \frac{\partial q(\theta)}{\partial \theta} \tag{2.9}$$

and

$$\mathrm{Var}(Y) = \frac{\partial^2 q(\theta)}{\partial \theta^2}\nu. \tag{2.10}$$

In this modeling framework, it is common to define $\frac{\partial^2 q}{\partial \theta^2}$ as the variance function $\mathrm{V}(\cdot)$, which is a function of the mean μ, so that $\mathrm{Var}(Y) = \mathrm{V}(\mu)\nu$, where $\mathrm{V}(\mu) = \frac{\partial^2 q}{\partial \theta^2}$.

The following table shows a few common distributions from an exponential family, and their parameterization in the above form.

The exponential family of distributions is a conjugate family [18] for the same class of distributions. Moving forward with our work, we will make note that in Bayesian

Table 2.1: Common distributions in exponential family of distributions

	Normal(μ, σ^2)	Poisson(μ)	Binomial(n, μ)
θ	μ	$\log(\mu)$	$\log(\frac{\mu}{1-\mu})$
ν	σ^2	1	1
$b(\theta)$	$\frac{\mu^2}{2}$	μ	$-n\log(1-\mu)$
$k(y, \nu)$	$-[\frac{y^2}{2\sigma^2} + \log(\sqrt{2\pi}\sigma)]$	$-\log(y!)$	$\log\binom{n}{y}$
V(μ)	1	μ	$\mu(1-\mu)$

settings, this means that the posterior distribution of the parameter θ, denoted here by f$(\theta|y)$, is a member of the exponential family, assuming that the prior of the same parameter, denoted by $f(\theta)$, as well as the sampling distribution of the data, $f(y|\theta)$, are also members of the exponential family.

2.2.2 Overview of Generalized Linear Models

The full specification of generalized linear models consists of three components: (1) a random component; (2) a systematic component; and (3) the link function.

The random component specifies that the distribution of the response variable belongs to the exponential family, as is described in the previous section. The systematic component or linear predictor specifies the linear form of the relationship between predictors and a function of the mean of the response variable. Finally, the link function specifies how one can use a transformation of the mean of the response to relate to the linear predictor component.

To illustrate this, suppose we want to explain a response Y using p predictors $X_1, \ldots X_p$ using a generalized linear model. Given N observations $(y_1, y_2, \ldots, y_N)$ of Y and N $p-$dimensional vectors $\boldsymbol{X}_i = (X_{i1}, X_{i2}, \ldots, X_{ip})^t$, one for each response y_i,

the systematic component of the model, or linear predictor, may be written as

$$\eta_i = \boldsymbol{X}_i^t \boldsymbol{\beta} = \beta_1 X_{i1} + \beta_2 X_{i2} + \cdots + \beta_p X_{ip}, \tag{2.11}$$

where $\boldsymbol{\beta}$ is a vector of fixed effects. The canonical link function $h(\cdot)$, which is a one-to-one continuous differentiable function, describes the relation between the above systematic component and the mean of Y_i, $\mu_i = E(Y_i \mid \beta, X_i)$, i.e.,

$$h(\mu_i) = \eta_i. \tag{2.12}$$

Common canonical link functions include the identity function for the normal distribution, log-function for the Poisson distribution and logit function for the Bernoulli distribution, as shown in Table 2.1.

Similar to general linear models, given that the distribution of the response is fully specified, maximum likelihood is the most natural estimation method. However, unlike in general linear models, there is no closed form solution for parameter estimates, and one often resorts to numerical methods. One such method is based on the Fisher scoring technique [21], an iterative method which states that the parameter estimate at the current iteration ($\boldsymbol{\theta}^{(m+1)}$) is obtained by using the parameter estimate in the previous iteration ($\boldsymbol{\theta}^{(m)}$), or a starting value of the parameter ($\boldsymbol{\theta}^{(0)}$) in the first iteration, and updating it by the product of the inverse Fisher information matrix and the score equation vector, both evaluated at $\boldsymbol{\theta}^{(m)}$, i.e.,

$$\boldsymbol{\theta}^{(m+1)} = \boldsymbol{\theta}^{(m)} + [I(\boldsymbol{\theta}^{(m)})]^{-1} \left. \frac{\partial l}{\partial \boldsymbol{\theta}} \right|_{\boldsymbol{\theta}^{(m)}}, \tag{2.13}$$

where $(i,j)^{th}$ element of the Fisher information matrix $I(\boldsymbol{\theta})$ is defined as

$$I(\boldsymbol{\theta})[i,j] = -E\Big(\frac{\partial^2 l(\boldsymbol{\theta})}{\partial\beta_i \partial\beta_j}\Big). \tag{2.14}$$

In case of generalized linear models, the above expression becomes [22]

$$\boldsymbol{\beta}^{(m+1)} = \boldsymbol{\beta}^{(m)} + (\boldsymbol{X}'\boldsymbol{W}\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{W}\boldsymbol{\Delta}(\boldsymbol{y}-\boldsymbol{\mu}), \tag{2.15}$$

where $\boldsymbol{\beta}$ are parameters being estimated, $\boldsymbol{X}$ is the matrix of explanatory variables, $\boldsymbol{y}$ is the vector of responses, $\boldsymbol{\mu}$ is the mean of the response $\boldsymbol{y}$, $\boldsymbol{W}$ is a diagonal matrix in which the i^{th} diagonal entry is $(V(\mu_i)h(\mu_i)^2)^{-1}$ and $\boldsymbol{\Delta}$ is a diagonal matrix in which the i^{th} diagonal entry is $h(\mu_i)$.

2.3 General Linear Mixed Models

In the case of the general linear models and generalized linear models, we have seen that there are limitations due to the assumptions that need to be altered so as to be able to model clustered or longitudinal data. In this section, we introduce general linear mixed models as an extension of general linear models and then describe some of the most common estimation methods.

2.3.1 Description

In the first section of this chapter, we introduced general linear models, which are simple models that describe the response variable as a linear function of regression parameters. In the first section we noted that there was only one random component that was associated with the error. The response is modeled at the

population level and is equally applicable to all units in the population. In the model, $E(\boldsymbol{Y}|\boldsymbol{X}) = \boldsymbol{X}\boldsymbol{\beta}$, there is only one parameter vector $\boldsymbol{\beta}$. The model implicitly assumes that the population of interest is homogeneous. However, that may not always be the case. When the population is not homogeneous, the change in the response variable may not be the same for all units.

We can now consider the case where in order to attempt to deal with this heterogeneity, we divide regression parameters into those that are common for all individuals, and those that vary across individuals. While the former are considered fixed effect in general linear models, the latter are considered random and are described by some probability distribution. This leads to general linear mixed models [23], [24] which extend general linear models by introducing random effect parameters, as in the following

$$\boldsymbol{Y}_i = \boldsymbol{X}_i\boldsymbol{\beta} + \boldsymbol{Z}_i\boldsymbol{b}_i + \boldsymbol{\epsilon}_i, \tag{2.16}$$

where $\boldsymbol{X}_i$ and $\boldsymbol{\beta}$ are as in the first section, $\boldsymbol{b}_i$ is a vector of random effects, and $\boldsymbol{Z}_i$ is a matrix of covariates linking $\boldsymbol{b}_i$ to $\boldsymbol{Y}_i$. Here, the column space of $\boldsymbol{Z}_i$ is often taken to be a subset of the column space of $\boldsymbol{X}_i$. Further assumptions include: $E(\boldsymbol{b}_i) = \boldsymbol{0}$, $\text{Cov}(\boldsymbol{b}_i) = \boldsymbol{G}$, $E(\boldsymbol{\epsilon}_i) = \boldsymbol{0}$ and $\text{Cov}(\boldsymbol{\epsilon}_i) = \boldsymbol{R}_i$, where $\boldsymbol{G}$ and $\boldsymbol{R}_i$ are suitable positive definite matrices. The error $\boldsymbol{\epsilon}_i$ is assumed to be normal and independent of random effects parameters $\boldsymbol{b}_i$. The distribution of $\boldsymbol{b}_i$ is most often assumed to be normal (this leads to special type of linear mixed models called the Gaussian linear mixed models), though in practice that assumption may not always hold [25]. The assumption of the mean of $\boldsymbol{b}_i$ being zero vector, leads to nice representation of the model parameters: $\boldsymbol{\beta}$ represents the effect of covariates $\boldsymbol{X}$ on the response $\boldsymbol{Y}$ at the

population level (as with general linear models), while $\boldsymbol{b}_i$ models deviation of the i^{th} unit from the population mean.

So we can see that with the additional random component, unlike general linear models, general linear mixed models allow one to make inferences both at the population level and at the individual level. The mean structure at the population level is captured by the marginal mean $E(\boldsymbol{Y}_i) = \boldsymbol{X}_i\boldsymbol{\beta}$, which is obtained by averaging out conditional means over all individuals, where the conditional mean is defined as $E(\boldsymbol{Y}_i|\boldsymbol{b}_i) = \boldsymbol{X}_i\boldsymbol{\beta} + \boldsymbol{Z}_i\boldsymbol{b}_i$. Additionally, we can note that the two mean structures also have different covariances: the conditional variance of $\boldsymbol{Y}_i$ is $\text{Cov}(\boldsymbol{Y}_i|\boldsymbol{b}_i) = \boldsymbol{R}_i$ and the marginal covariance of $\boldsymbol{Y}_i$ is $\boldsymbol{V}_i(\boldsymbol{\theta}) \equiv \boldsymbol{V}_i = \text{Cov}(\boldsymbol{Y}_i) = \boldsymbol{Z}_i\boldsymbol{G}\boldsymbol{Z}_i^t + \boldsymbol{R}_i$, where we parameterize the covariance matrix $\boldsymbol{V}_i$ by some parameter vector $\boldsymbol{\theta}$.

We can see from this that the random effects induce covariance structures among the components of $\boldsymbol{Y}_i$, and allows for explicit analysis of both within-subject variation ($\boldsymbol{R}_i$) and between-subject variation ($\boldsymbol{G}$).

2.3.2 Parameter Estimation

The most common method of estimating fixed effects parameters in Gaussian linear mixed models is the maximum likelihood method, first used by Hartley [26]. Here, the log-likelihood function is given by

$$l(\boldsymbol{\beta}, \boldsymbol{\theta}) = \text{const} - \sum_{i=1}^{N} \frac{1}{2}\log(|\boldsymbol{V}_i(\boldsymbol{\theta})|) - \frac{1}{2}(\boldsymbol{y}_i - \boldsymbol{X}_i\boldsymbol{\beta})^t\boldsymbol{V}_i(\boldsymbol{\theta})^{-1}(\boldsymbol{y}_i - \boldsymbol{X}_i\boldsymbol{\beta}). \quad (2.17)$$

The point estimates of $\boldsymbol{\beta}$ and $\boldsymbol{\theta}$ (and therefore the variance $\boldsymbol{V}_i(\boldsymbol{\theta})$) are obtained by differentiating the log-likelihood with respect to $\boldsymbol{\beta}$ and $\boldsymbol{\theta}$ and solving the following

equations

$$\begin{aligned}
\frac{\partial l}{\partial \boldsymbol{\beta}} &= \sum_{i=1}^{N} \left\{ \boldsymbol{X}_i^t \boldsymbol{V}_i(\boldsymbol{\theta})^{-1} \boldsymbol{y}_i - \boldsymbol{X}_i^t \boldsymbol{V}_i(\boldsymbol{\theta})^{-1} \boldsymbol{X}_i \boldsymbol{\beta} \right\} = \mathbf{0}, \\
\frac{\partial l}{\partial \theta_r} &= \sum_{i=1}^{N} \frac{n}{2} \log(|\boldsymbol{V}_i(\boldsymbol{\theta})|) - \\
&\quad \frac{1}{2} \left\{ (\boldsymbol{y}_i - \boldsymbol{X}_i \boldsymbol{\beta})^t \boldsymbol{V}_i(\boldsymbol{\theta})^{-1} \frac{\partial \boldsymbol{V}_i(\boldsymbol{\theta})}{\partial \theta_r} (\boldsymbol{y}_i - \boldsymbol{X}_i \boldsymbol{\beta}) - \operatorname{tr}\left(\boldsymbol{V}_i(\boldsymbol{\theta})^{-1} \frac{\partial \boldsymbol{V}_i(\boldsymbol{\theta})}{\partial \theta_r} \right) \right\} = 0
\end{aligned} \tag{2.18}$$

for $r = 1, 2, \ldots, q$. The point estimate of $\boldsymbol{\beta}$ is the same as in Eq.(2.6), and the point estimates of $\boldsymbol{V}(\boldsymbol{\theta})$ depends on its parametrization of $\boldsymbol{\theta}$ and the point estimates of $\boldsymbol{\theta}$, but in general is biased and inconsistent [27]. Using restricted maximum likelihood methods [28] corrects the bias, and leads to consistent and asymptotically normal estimates.

2.3.3 Parameter Prediction

It is well known in classical statistics that one cannot use the same approach for the random and fixed effects parameters. While fixed effects parameters are estimated, the random effects parameters are predicted. It is easy to show that the best linear unbiased predictor (BLUP) of random effects parameter $\boldsymbol{b}_i$ is its conditional mean given $\hat{\boldsymbol{\beta}}$ and response $\boldsymbol{Y}_i$

$$\hat{\boldsymbol{b}}_{i_{BLUP}} = E(\boldsymbol{b}_i | \boldsymbol{Y}_i) = \boldsymbol{G} \boldsymbol{Z}_i^t \boldsymbol{V}_i^{-1}(\boldsymbol{\theta})(\boldsymbol{Y}_i - \boldsymbol{X}_i \hat{\boldsymbol{\beta}}), \tag{2.19}$$

where $\boldsymbol{V}_i$ is defined in the previous section. Using the MLE or restricted MLE estimate of $\boldsymbol{V}_i(\boldsymbol{\theta})$, we obtain the empirical best linear unbiased predictor or EBLUP, as

$$\hat{\boldsymbol{b}}_{i_{EBLUP}} = E(\boldsymbol{b}_i \mid \boldsymbol{Y}_i) = \boldsymbol{G} \boldsymbol{Z}_i^t \hat{\boldsymbol{V}}_i^{-1}(\boldsymbol{\theta})(\boldsymbol{Y}_i - \boldsymbol{X}_i \hat{\boldsymbol{\beta}}). \tag{2.20}$$

2.4 Generalized Linear Mixed Models

In this section, we introduce generalized linear mixed models in terms of how they compare to the previous models, and then describe some of the common estimation methods.

2.4.1 Description

Generalized linear mixed models extend generalized linear models by adding random effects parameters to this linear predictor, or equivalently, they extend general linear mixed models by loosening the distributional and linearity assumptions. The linear predictor in generalized linear mixed models, for j^{th} response of i^{th} unit, is defined as

$$\eta_{ij} = \boldsymbol{X}_{ij}^t\boldsymbol{\beta} + \boldsymbol{Z}_{ij}^t\boldsymbol{b}_i. \tag{2.21}$$

As with general linear mixed models, the rationale for introducing random effects parameters into generalized linear models (to get generalized linear mixed models) is to address the case where there is heterogeneity in the data.

However, unlike in general linear mixed models, fixed effects parameters $\boldsymbol{\beta}$ do not have the same interpretation in the conditional model $E(\boldsymbol{Y}_i|\boldsymbol{b}_i) = \boldsymbol{X}_i\boldsymbol{\beta} + \boldsymbol{Z}_i\boldsymbol{b}_i$ as in the marginal model $E(\boldsymbol{Y}_i) = \boldsymbol{X}_i\boldsymbol{\beta}$. This is due to the fact that the expectation operator is a linear operator and the link function in generalized linear models is non-linear function (except in trivial case when it is an identity transformation). It is well known that the sum of a nonlinear function is not the same as a nonlinear function of the sum. Therefore, parameters in generalized linear mixed models have subject-specific interpretation, and are most useful when the main objective is to make inferences at the subject level, as opposed to the whole population. Models

that are targeted for population-level inferences, of which there are multiple types (called marginal models) are not discussed in this thesis.

2.4.2 Parameter Estimation

Unlike general linear mixed models, parameter estimation in generalized linear mixed models is computationally very challenging. Several classes of methods have been proposed. Numerical integration is often used in low-dimensional parameter spaces [29]. However, many generalized linear mixed models include high-dimensional parameter spaces which makes numerical integration techniques quite challenging. For these other situations, Monte Carlo Expectation Maximization (MCEM) method [30] is used. The MCEM method is an estimation method based on the maximum-likelihood. It uses Monte Carlo method to approximate the conditional expectation in the expectation step of the expectation maximization (EM) [31] algorithm. Additionally, estimation by parts [32] can be used as another likelihood-based estimation method.

As an estimation method based on approximation, one often uses penalized quasi-likelihood (PQL) [33] that is based on an approximation of the marginal quasi-likelihood [20] using the Laplace approximation [34].

2.5 Summary

In this chapter, we have reviewed four large classes of statistical models. We started off by introducing general linear models. We described their main characteristics and the most common estimation methods, including the maximum likelihood and restricted maximum likelihood. We then introduced generalized linear models as

an extension to general linear models. In the context of generalized linear models, we described exponential family of distributions and the important role it plays in generalized linear models.

Following the introduction of two fixed effects models, we introduced two classes of models that include random effects parameters, and are often referred to as the mixed effects models, since they contain both fixed and random effects parameters. These include general linear mixed models and generalized linear mixed models. We started by introducing general linear mixed models as an extension of general linear models, and described how its estimation actually consists of two parts: estimation of fixed effects parameters and prediction of random effects parameters. We described how optimal predictions of random effects are obtained. Lastly, we introduced generalized linear mixed models as an extension of general linear mixed models and described how they differ from the other three classes of models.

Chapter 3

Bayesian Nonparametric Statistics

In this section, we first introduce the basic concepts of Bayesian nonparametric statistics, with the main focus on Dirichlet Process and Dirichlet Process mixture models. Then, we review two classes of methods of approximating posterior distribution of model parameters: the simple Monte Carlo and Markov chain Monte Carlo methods. Finally, we review few common sampling algorithms in both Bayesian parametric and nonparametric statistics.

3.1 Introduction

To summarize a set of N observations $\boldsymbol{y} = \{y_1, y_2, \ldots, y_N\}$, statistics (as a discipline) uses probability theory to describe the underlying mechanism that could have generated the observations. In parametric statistics, design based or classical, the probability model is fully specified by a family of probability distributions and a parameter vector Θ. One of the main objectives in frequentist statistics is to accurately estimate the subset of the parameter space which identifies the plausible probability distribution of the model that could have generated the observed data. Knowing the parameters of the distribution equates to knowing everything one needs to know

about the data, assuming that the assumption of the parametric family describing the probability model is correct.

In the next section, we introduce the basic concepts in Bayesian (parametric) statistics that are relevant to this thesis.

3.1.1 Bayesian Parametric Statistics

Unlike frequentist statistics in which unknown parameters are considered fixed, Bayesian parametric statistics describes unknown parameters through their probability distributions. In fact, in different stages of a statistical process, more than one probability model may be used to describe the same set of unknown parameters. In Bayesian statistics, one views the statistical inference as a process of updating uncertainty about parameters.

Prior to observing the data, the knowledge about unknown parameters may be encapsulated by a probability distribution called the prior distribution $P(\Theta)$. Data generation mechanism specifies uncertainty of observing realizations of a variable Y through another probability model, called the likelihood and is denoted by $L(\Theta|\boldsymbol{y})$. After observing the data, we update the uncertainty about the parameters of interest through another distribution called the posterior distribution $P(\Theta|\boldsymbol{y})$. These three probability models are linked through the Bayes' theorem [35], as follows:

$$P(\Theta|\boldsymbol{y}) = \frac{P(\Theta, \boldsymbol{y})}{P(\boldsymbol{y})} = \frac{P(\Theta) \times P(\boldsymbol{y}|\Theta)}{P(\boldsymbol{y})} \propto P(\Theta) \times L(\Theta|\boldsymbol{y}), \tag{3.1}$$

which says that the posterior probability of parameters is fully determined by the prior distribution of the parameters $P(\Theta)$ and the likelihood function $L(\Theta|\boldsymbol{y})$ (also called the sampling distribution).

The prior probability $P(\Theta)$ encapsulates one's knowledge about the parameters of interest before observing the data. Common types of priors include non-informative priors, informative priors and weakly informative priors. Non-informative priors, such as Jeffrey's prior [36], "provide little information relative to data" [37]. Informative priors deliberately insert knowledge about parameters into the model, using subject matter knowledge or perhaps using results from previous experiments in which case they are also called the power priors [38]. Weakly informative priors contain some information about the prior but "without attempting to fully capture one's scientific knowledge about the underlying parameter" [35].

All summaries in Bayesian statistics are carried out using the posterior distribution. This may include the location and dispersion of parameters Θ. Parameters may also be summarized through credibility regions. A region S is $100(1-\alpha)\%$ credibility region if

$$\int_S P(\Theta|\boldsymbol{y})d\Theta = 1-\alpha. \tag{3.2}$$

A region S which satisfies the above property, and which does not contain "smaller" regions that satisfy the same property, is called the Highest Posterior Density (HPD) region. When Θ is univariate, we call this region the HPD interval, which is much closer to everyday interpretation of results of inference than are confidence intervals in frequentist statistics [39].

In Bayesian parametric statistics, the family of probability distributions is assumed to be known. Depending on the support of the variable of interest (defined as the subset of its domain of positive probability), this could be the normal, beta,

gamma, Poisson or some other well-known distribution. Only parameters of these distributions are assumed unknown, and in Bayesian statistics, the uncertainty of the knowledge about them is described by a probability distribution. This makes the parameter space in Bayesian parametric statistics very simple - most often it is either a vector space, such as the space of all vectors or matrices, and probability distributions over these spaces are assumed to be well known.

In Bayesian statistics we assume that the observations are exchangeable. This is a weaker assumption than independence and assumes that the order in which data are observed is not important. In other words, the joint distribution of the data is the same for any permutation of it. Independence implies exchangeability but exchangeability does not imply independence.

3.1.2 Bayesian Nonparametric Statistics

Bayesian nonparametric statistics, like frequentist nonparametric statistics, relaxes the assumption that the probability distribution comes from a known family. Here the probability distribution is an unknown entity, and in spirit of Bayesian statistics, it needs to be described by some probability distribution. Therefore, we need a probability distribution to describe another probability distribution.

As an example, consider again a sample $\boldsymbol{y} = \{y_1, y_2, \ldots, y_N\}$ consisting of N observations. In Bayesian parametric statistics, one may assume that each observation y_i comes from the normal distribution with unknown mean and variance, i.e., $Y \sim \mathrm{N}(\mu, \sigma^2)$, and would proceed by describing the uncertainty about the parameters $\boldsymbol{\theta} = (\mu, \sigma^2)$ with a prior distribution. However, in Bayesian nonparametric statistics, one may assume that the distribution of Y is unknown, i.e., $Y \sim F$, where F may be an arbitrary probability distribution. We don't have a general probability

distribution that can be any distribution out there. But we could express some preference (or indifference) about some class of probability distributions for our F and would need a mechanism to describe an uncertainty about this (as we do with all unknowns in Bayesian statistics). Probabilities are a good way to do describe uncertainties. Therefore, we need some kind of a probability distribution Λ that describes an uncertainty about F, i.e., $F \sim \Lambda$. This is the case where a distribution becomes a random object - it is random because there is a process/mechanism out there (in this example it is Λ) that generates these distributions probabilistically.

The space of all probability distributions is very large, and setting a probability distribution on that space is not as easy as setting probability distribution on vector spaces of finite dimension. The main difficulty is that the space of all probability distributions is of infinite dimension, while a vector space in Bayesian parametric statistics has a finite dimension.

To illustrate this further, consider again the above example where F is now a discrete distribution. Modeling this distribution using the standard Bayesian approach would be very easy if the number of different categories that a random variable may take was known in advance. Dirichlet distribution would be a natural choice for its prior. But what if we do not know in advance the number of categories a variable may take? One option would be to fit different models, assuming a known number of categories for each model, and then compare model fits and find the most plausible model. Another approach would be to place a prior (Λ in the above example) on all discrete distributions, without specifying the number of categories in advance, and letting the data choose the most appropriate model. There is a prior in Bayesian nonparametric statistics that can do exactly that, and is called the

Dirichlet Process prior [15]. We describe it in the next section.

Now, consider a case when our distribution F is continuous. In Bayesian nonparametric settings, a prior Λ would be required on all continuous distribution functions, and to make this prior practical, we need to be able to update the prior with the evidence presented with the observed data. The space of all continuous distribution functions is quite rich and Bayesian nonparametric statistics presents many priors for modeling continuous distribution functions, one of which is the Dirichlet Process mixture prior [40], also presented in the next section.

The unknown entities in our models need not be constrained to only probability distributions. They could be more complex structures, such as continuous functions [41], binary matrices with infinite number of columns [42], or even graphs [43]. Whatever it is, we need to specify a probability model on these spaces. This is quite challenging. It is also the reason why we have so many different types of priors in Bayesian nonparametric statistics, because each structure may require more than one type of a prior.

The above does not mean that we do not use parametric distribution functions in our models. In fact, we do, and we may use infinitely many of them (theoretically). We use parametric models to build nonparametric models. In that sense, Bayesian nonparametric statistics is a Bayesian parametric statistics with infinite number of parameters. The number of parameters are determined by the data - we let the data decide the complexity of the model, instead of us imposing model restrictions on the data.

3.2 Dirichlet Process

In this section, we first review the Dirichlet distribution, which is the core ingredient of the Dirichlet Process. In fact, the Dirichlet Process may be considered as a generalization of the Dirichlet distribution to an infinite dimensional spaces. Then, we introduce the Dirichlet Process and describe some of its key properties. Finally, we introduce three different representations of the Dirichlet Process, which are often used in sampling in Dirichlet Process and Dirichlet Process mixture models.

3.2.1 Dirichlet Distribution

A Dirichlet distribution [1] is a continuous multivariate generalization of the Beta distribution [1]. To see this, let θ_1 have beta distribution with shape parameters $\boldsymbol{\alpha} = \{\alpha_1, \alpha_2\}$. Then the density function of θ_1 is

$$P(\theta_1|\boldsymbol{\alpha}) = \frac{\Gamma(\alpha_1 + \alpha_2)}{\Gamma(\alpha_1)\Gamma(\alpha_2)}\theta_1^{\alpha_1-1}(1-\theta_1)^{\alpha_2-1},$$

where $\Gamma(\theta)$ it the standard gamma function. The support of θ_1 is $(0, 1)$. Equivalently, by introducing a "dummy" variable θ_2 such that $\theta_1 + \theta_2 = 1$, we can express the above univariate probability density function as a bivariate density function of $\boldsymbol{\theta} = (\theta_1, \theta_2)^t$ as follows

$$P(\boldsymbol{\theta}|\boldsymbol{\alpha}) = \frac{\Gamma(\sum_{i=1}^{2}\alpha_i)}{\prod_{i=1}^{2}\Gamma(\alpha_i)}\prod_{i=1}^{2}\theta_i^{\alpha_i-1}.$$

Extending the above bivariate variable $\boldsymbol{\theta}$ to a D-variate variable $\boldsymbol{\theta} = \{\theta_1, \theta_2, \ldots, \theta_D\}$, such that $\theta_i \geq 0, \sum_{i=1}^{D}\theta_i = 1$, with parameters $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_D)$, we get the

probability density function of the Dirichlet distribution with parameter $\boldsymbol{\alpha}$, given as

$$P(\boldsymbol{\theta}|\boldsymbol{\alpha}) = \frac{\Gamma(\sum_{i=1}^{D} \alpha_i)}{\prod_{i=1}^{D} \Gamma(\alpha_i)} \prod_{i=1}^{D} \theta_i^{\alpha_i - 1}, \tag{3.3}$$

and we write $\boldsymbol{\theta} \sim \text{Dirichlet}(\boldsymbol{\alpha})$. We say that the support of $\boldsymbol{\theta}$ is a $D-1$ simplex. Assuming $(\theta_1, \ldots, \theta_D) \sim \text{Dirichlet}(\alpha_1, \ldots, \alpha_D)$, the expected value and variance of the i^{th} component of $\boldsymbol{\theta}$ are

$$E(\theta_i) = \frac{\alpha_i}{\alpha}, \text{ and } \text{Var}(\theta_i) = \frac{\alpha_i(\alpha - \alpha_i)}{\alpha^2(\alpha + 1)}, \tag{3.4}$$

where $\alpha = \sum_{i=1}^{D} \alpha_i$ is often referred to as the concentration parameter and determines how much the distribution varies around its expected value. Dirichlet distribution is often re-parameterized so that $\boldsymbol{\alpha} = \alpha \frac{\boldsymbol{\alpha}}{\alpha} = \alpha \boldsymbol{p}$. where $\boldsymbol{p}$ is new parameter whose components sum to one. This is useful when working with probabilities and will be used in this thesis in such context.

Figure 3.1, borrowed from [1], shows densities of Dirichlet distributions with different parameter values.

The Dirichlet distribution may be constructed using the gamma distribution: if $\theta_i \sim \text{Gamma}(\alpha_i, 1)$ are independent, then $\theta = \sum_{i=1}^{D} \theta_i \sim \Gamma(\alpha)$, where $\alpha = \alpha_1 + \alpha_2 + \cdots + \alpha_D$, and

$$\left(\frac{\theta_1}{\theta}, \cdots, \frac{\theta_D}{\theta}\right) \sim \text{Dirichlet}(\alpha_1, \ldots, \alpha_D).$$

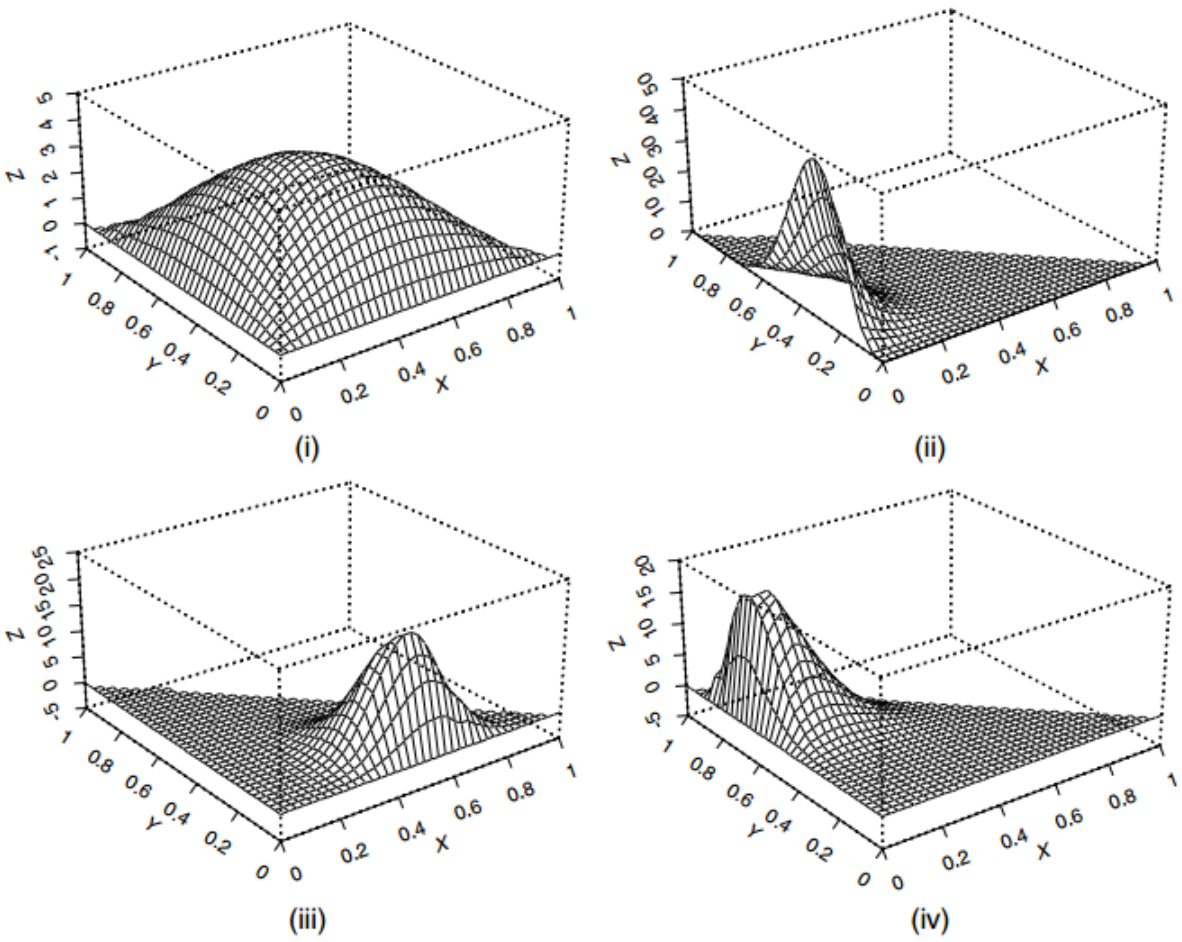

Figure 3.1: Density of Dirichlet distribution ($D = 3$), with different parameter values: (i) $\alpha_1 = \alpha_2 = \alpha_3 = 2$; (ii) $\alpha_1 = 1, \alpha_2 = 5, \alpha_3 = 10$; (iii) $\alpha_1 = 10, \alpha_2 = 3, \alpha_3 = 8$; (iv) $\alpha_1 = 2, \alpha_2 = 10, \alpha_3 = 4$. Figure from [1].

The above property may be used to prove the following two properties of the Dirichlet distribution: agglomeration and decimation. If $\boldsymbol{\theta} = \{\theta_1, \theta_2, \ldots, \theta_D\} \sim$ Dirichlet$(\alpha_1, \alpha_2, \ldots, \alpha_D)$, then for any partition $(P_1, P_2, \ldots, P_K)$ of $\{1, 2, \ldots, D\}$, the agglomeration property holds:

$$\left(\sum_{i \in P_1} \theta_i, \ldots, \sum_{i \in P_K} \theta_i\right) \sim \text{Dirichlet}\left(\sum_{i \in P_1} \alpha_i, \ldots, \sum_{i \in P_K} \alpha_i\right). \tag{3.5}$$

Additionally, for $(\eta_1, \eta_2) \sim$ Dirichlet$(\alpha_1\delta_1, \alpha_1\delta_2)$, where $\delta_1 + \delta_2 = 1$, the decimation property holds:

$$(\theta_1\eta_1, \theta_1\eta_2, \theta_2, \ldots, \theta_D) \sim \text{Dirichlet}(\alpha_1\delta_1, \alpha_1\delta_2, \alpha_2, \ldots, \alpha_D).$$

One of the key properties of the Dirichlet distribution is that it is a conjugate prior of the multinomial distribution. The multinomial distribution is a generalization of the binomial distribution - it models the probability of counts of each of D possible outcomes, where the total number of counts (n) if fixed. Its probability mass function is given by the following expression

$$P(\boldsymbol{Y} = (y_1, y_2, \ldots, y_D)|\boldsymbol{\theta}) = \frac{n!}{y_1!y_2!\ldots y_n!}\theta_1^{y_1}\theta_2^{y_2}\ldots\theta_D^{y_D},$$

when $\sum_{i=1}^{D} y_i = n$ and 0 otherwise.

Following the above notation, if $\boldsymbol{Y}|\boldsymbol{\theta} \sim$ Multinomial$(\boldsymbol{\theta})$ and if $\boldsymbol{\theta} \sim$ Dirichlet$(\boldsymbol{\alpha})$, then the posterior distribution of $\boldsymbol{\theta}$ is also Dirichlet but with updated parameters: $\boldsymbol{\theta}|\boldsymbol{y} \sim$ Dirichlet$(\boldsymbol{y} + \boldsymbol{\alpha}) \equiv$ Dirichlet$(\alpha_1 + y_1, \alpha_2 + y_2, \ldots, \alpha_D + y_D)$. We see that the parameter of the k^{th} component of the Dirichlet distribution α_k is updated by adding to it the total number of observations that have fallen into the k^{th} category, $k = 1, 2. \ldots, D$. Due to this property, the components of the parameter vector $\boldsymbol{\alpha}$ are

often called pseudo counts.

3.2.2 Dirichlet Process

In this section, we first describe the intuition behind the Dirichlet Process, followed by a more formal definition of it. We also introduce a few main properties of the Dirichlet Process that are used in the rest of the section and in the thesis.

A histogram, first introduced by Karl Pearson in 1895 [44], is a simple non-parametric method of estimating density of continuous probability distributions. When used in Bayesian settings, which is then known as Bayesian histogram [45], it may lead to adequate approximation of the true density, though it suffers from sensitivity due to the number of bins used as well as the placement of knots that define their boundaries.

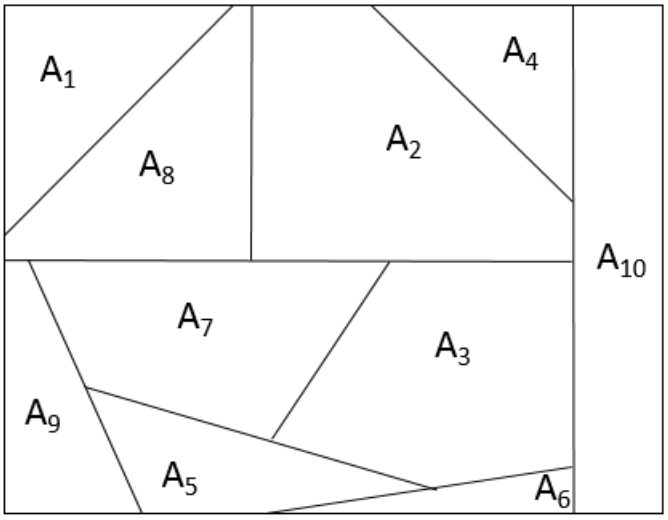

Figure 3.2: Partition of a parameter space

Consider an unknown distribution function G with the density function $f(\cdot)$. To estimate this distribution in frequentist setting, we could partition the range of the parameter space as in Figure 3.2, count the number of observations that fall into area A_i (we denote these counts by $n_i, i = 1, \ldots, 10$) and then approximate $G(A_i) = \int_{A_i} f(\theta)d\theta$ by n_i/n, where n is the total number of observations. Assuming we can get as much data as needed, this approximation may be obtained at an arbitrary level of precision [46].

In Bayesian settings, if G is a random probability distribution, then the vector $(G(A_1), \ldots, G(A_{10}))$ is a random vector. A natural choice for the prior on the given partition would be the Dirichlet distribution (due to its conjugacy with the multinomial distribution). Then, the posterior distribution of G given the data would also be Dirichlet. We would like this to hold for any partitioning of the parameter space, which is required if there was to be a distribution of G. Ferguson [47] showed that such distribution of G exists by proving that the Kolmogorov's consistency theorem holds [48]. That distribution of G is called the Dirichlet Process.

It is immediately clear how the agglomeration property of the Dirichlet distribution in Eq.(3.5) plays an important role in the Dirichlet Process. This is so since the distribution of $(G(A_1), \ldots, G(A_{10})$ is Dirichlet, as is the distribution of any vector that is constructed after two or more subsets of the partition have been merged. The decimation property guarantees that the same would hold if the existing partitions are subdivided into smaller partitions.

More formally, the Dirichlet Process is a special type of a stochastic process, the realization of which is a probability distribution. Stochastic processes are often

defined in terms of probability distribution they induce on a finite subset of random variables. For example, a stochastic process $(Y_1, Y_2, Y_3, \dots)$ is called the Gaussian process [49] if a subset of D random variables $(Y_{i1}, Y_{i2}, \dots, Y_{iD})$, $D > 0$, has a D-dimensional normal distribution. Similarly, the Dirichlet Process is a type of a stochastic process which induces a D-dimensional Dirichlet distribution on any finite subset of D random variables $(Y_{i1}, Y_{i2}, \dots, Y_{iD})$.

To summarize, given a base probability distribution G_0, and a positive scalar α, a probability distribution G is Dirichlet Process distributed, denoted by $G \sim \mathrm{DP}(\alpha, G_0)$ if the marginal distribution of any finite partition of the parameter space $\{A_1, A_2, \dots, A_n\}$ has Dirichlet distribution, i.e.,

$$P(G(A_1), G(A_2), \dots, G(A_n)|\alpha, G_0) \sim \mathrm{Dirichlet}(\alpha G_0(A_1), \alpha G_0(A_2), \dots, \alpha G_0(A_n)). \tag{3.6}$$

The two parameters α and G_0 that define the Dirichlet Process have an important conceptual interpretation. The base distribution is the mean of the Dirichlet Process - it is our best guess at the unknown distribution G, while the scalar parameter α controls the variability of the realized distribution function around the base distribution G_0. For any subset A_i of the parameter space, we have

$$E(G(A_i)) = G_0(A_i) \tag{3.7}$$

and

$$Var(G(A_i)) = \frac{G_0(A_i)(1 - G_0(A_i))}{\alpha + 1}. \tag{3.8}$$

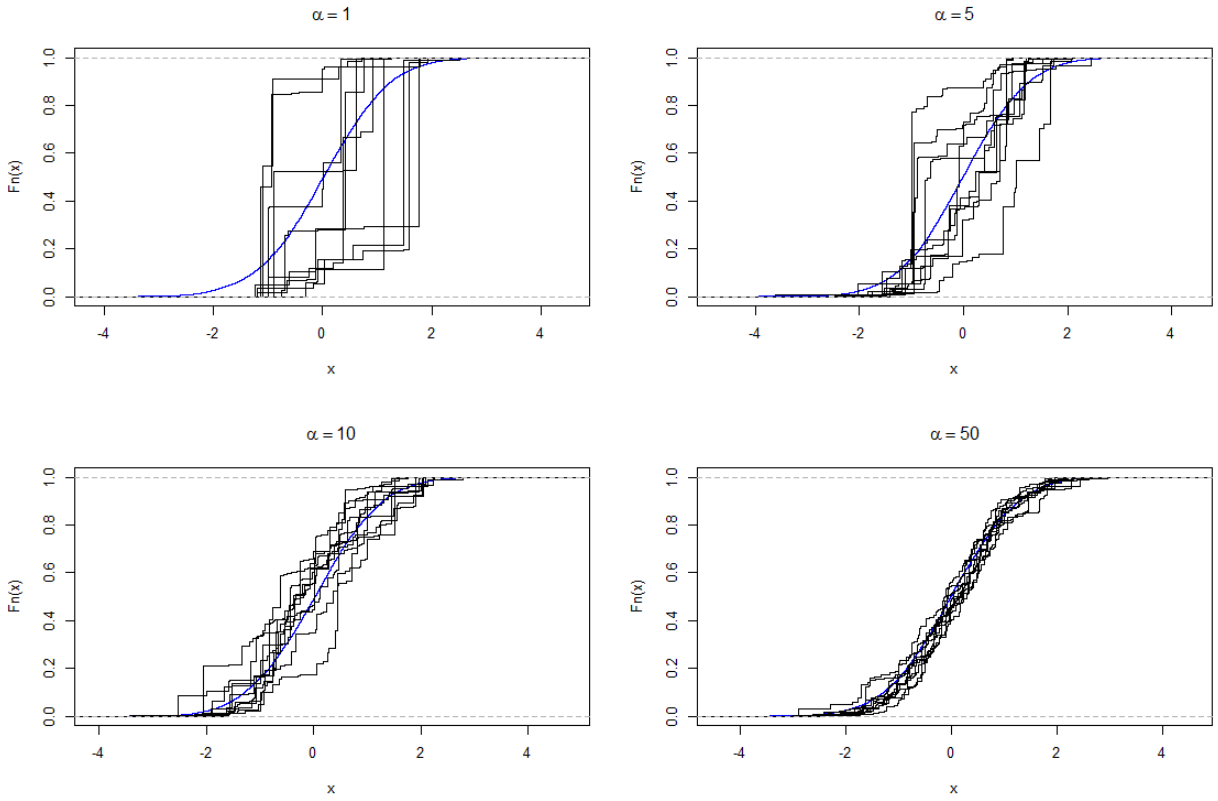

Figure 3.3: Empirical cumulative distribution functions from $\mathrm{DP}(\alpha, N(0,1))$ (ten for each value of α), for four different values of $\alpha = 1, 5, 10, 50$. The true cumulative distribution function is shown in blue.

This clearly shows why α is also called the precision parameter.

Figure 3.3 shows 10 empirical cumulative distribution functions in each plot, based on draws from a Dirichlet Process with the precision parameter taking one of the following four values: 1, 5, 10, and 50. The base distribution is set to $N(0, 1)$. We see that for smaller values of α, the distribution G varies a lot around the base distribution G_0, while for larger values of α, distributions drawn from $\mathrm{DP}(\alpha, G_0)$ resembles more to G_0.

Dirichlet Process is a conjugate prior to itself. For illustration of this property, let G be a probability distribution drawn from the Dirichlet Process with the base distribution G_0 and scalar parameter α, i.e., $G \sim DP(\alpha, G_0)$. Having observed the distribution G, we may draw observations from it. Assume we have drawn a sample of size N from G, $\boldsymbol{y} = \{y_1, y_2, \ldots, y_N\}$. The observed sample gives us information about the unknown distribution G, and we may use it to estimate G. To estimate G, we start with an arbitrary partition of the sample space. Let $\{A_1, A_2, \ldots, A_n\}$ be one such partition. Then, due to conjugacy of the Dirichlet and the multinomial distribution, we have

$$P(G(A_1), \ldots, G(A_n)|\boldsymbol{y}) \sim \text{Dirichlet}\left(\alpha G_0(A_1) + n_1, \ldots, \alpha G_0(A_n) + n_n\right), \tag{3.9}$$

where n_i is the number of data points in the sample $\boldsymbol{y}$ that fall into the set A_i. Since the above holds for any partition of the parameter space, it follows that

$$G(y)|\alpha, \boldsymbol{y}, G_0 \sim \mathrm{DP}(\alpha + N, \frac{\alpha}{\alpha + N} G_0(y) + \frac{1}{\alpha + N} \sum_{i=1}^{N} \delta_{y_i}(y)), \tag{3.10}$$

where $\delta_{y_i}(y) = 1$ if $y = y_i$ and zero otherwise. The above expression shows that the posterior distribution of a probability distribution G is also Dirichlet Process

distributed but with updated parameters. This shows that the Dirichlet process is a conjugate prior to itself. The new value of the concentration parameter of updated Dirichlet Process is incremented by the number of observations taken from the random distribution G, while the base distribution of the updated Dirichlet Process is a weighted average between the base distribution of the initial process (G_0) and the empirical distribution $\sum_{i=1}^{N} \delta_{y_i}(y)/N$, since the Eq.(3.10) may be written as

$$G(y)|\alpha, \boldsymbol{y}, G_0 \sim \mathrm{DP}\Big(\alpha + N, \frac{\alpha}{\alpha + N} G_0(y) + \frac{N}{\alpha + N} \frac{\sum_{i=1}^{N} \delta_{y_i}(y)}{N}\Big). \tag{3.11}$$

Using the expression in Eq.(3.7), the expected probability of any subset A of the parameter space is

$$E(G(A)|\alpha, \boldsymbol{y}, G_0) = P(y_{n+1} \in A|\alpha, \boldsymbol{y}, G_0) = \frac{1}{\alpha + N}\Big(\alpha G_0(A) + \sum_{i=1}^{N} \delta_{y_i}(A)\Big), \tag{3.12}$$

which clearly shows that the posterior distribution of the base distribution of updated Dirichlet Process is also the predictive distribution of the new observation. This is an important property of the Dirichlet Process that may be used in sampling. Given any subset A of the parameter space, and using Eq.(3.12), we have

$$\lim_{N\to\infty} E(G(A)|\alpha, G_0, \boldsymbol{y}) = \sum_{i=1}^{\infty} \pi_i \delta_{y_i^*}(A)$$

where y_i^* are unique values of y_i, and π_i is the limiting empirical frequency of y_i^*. The above expression shows that probability distributions drawn from a Dirichlet Process are discrete with probability one [50].

3.2.3 Representations of the Dirichlet Process

In the previous section we introduced the Dirichlet Process and described few of its key properties. However, in order to use it, we need to be able to generate a random distribution from it. Given parameters of the Dirichlet Process (the concentration parameter α and the base distribution G_0), how do we generate a random distribution from it? Furthermore, in order to be able to use a random object in Bayesian statistics, we need to be able to derive the posterior distribution of the object given the observed data. We also need to be able to sample from the posterior distribution. In the previous section, we showed that some of these tasks may be simple in case of Dirichlet Process. However, as will be seen in the remainder of this chapter, Dirichlet Process is used in construction of more complex probabilistic models, and updating posterior distributions in these models, or performing other Bayesian tasks, may be quite challenging. Therefore, in the remainder of this section, we introduce three common representations of a Dirichlet Process. Each representation leads to a specific sampling method.

Stick-breaking representation

Stick-breaking representation of the Dirichlet Process was first introduced by Sethuraman [51]. To generate an instance of a random distribution G from a Dirichlet Process with base distribution G_0 and positive scalar α, Sethuraman suggests generating two sequences of random variables:

- Draw W_i from $\text{Beta}(1, \alpha), i = 1, 2, \ldots,$ where W_i is independent of $W_j, i \neq j$; and
- Draw Y_i independently from $G_0, i = 1, 2, \ldots.$

Then, letting $\pi_i = W_i \prod_{j=1}^{i-1}(1 - W_j), i = 1, 2, \ldots,$ we can express G as an infinite mixture of discrete distributions (each defined at a single atom from G_0), i.e.,

$$G(y) = \sum_{i=1}^{\infty} \pi_i \delta_{y_i}(y) \tag{3.13}$$

where $\delta_{y_i}(\cdot)$ is the point mass at y_i. The atoms y_i are determined by the support of the base distribution, and the probability of sampling an atom y_i is determined by the mixing probability $\pi_1, \pi_2, \ldots$. So, for example, if we want a prior for distributions whose support is the real line, then the candidate for the base distribution G_0 might be the normal distribution or the t-distribution. However, if we want a prior for a distribution whose support is the set of only positive real numbers, then the gamma distribution might be a good candidate.

The metaphor behind the stick-breaking representation is that if we consider a stick of length one, we can break it into two pieces: the location where we break the stick is determined by W_1 and the length of the first piece is π_1. Then we repeatedly break the remaining part of the stick, at (a relative) location determined by W_i, and assign the weight π_i to the part of the stick being broken off (we continue braking the stick from the same side). This process may continue indefinitely. In practice though, it is often stopped at some point where the accuracy of the resulting distribution G is deemed good enough.

Using Eq.(3.4) and the above construction process, we have

$$E(W_i) = \frac{1}{1 + \alpha},$$

which shows that smaller values of alpha will result in fewer atoms from G_0 being

assigned larger probabilities, while for larger values of α, these probabilities will be spread out over larger set of atoms from G_0.

Polya Urn representation

Taking into consideration that the draws from a Dirichlet Process are going to result in duplicates, Eq.(3.10) may be expressed in terms of unique observations y_i^*, as follows

$$G(y)|\alpha, \boldsymbol{y}, G_0 = \mathrm{DP}(\alpha + N, \frac{\alpha}{\alpha + N} G_0(y) + \frac{1}{\alpha + N} \sum_{k=1}^{K} N_k \delta_{y_k^*}(y), \tag{3.14}$$

where N_k is the number of observations that are equal to y_k^*. In the above notation, out of N observations, only K are unique.

Using the results from the previous section in Eq.(3.12), we may express the predictive distribution of y_{N+1}^* as follows

$$y_{N+1}^* = y|y_1^*, \ldots, y_K^*, \alpha, G_0 \sim \frac{1}{\alpha + N} \left(\alpha G_0(y) + \sum_{k=1}^{K} N_k \delta_{y_k^*}(y) \right). \tag{3.15}$$

The predictive distribution of the new observation specifies that its value is equal to one of the existing observations with probability that is proportional to the number of observations (N_k) that have the same value as that observation, or its value will be drawn from the base distribution G_0 and this will happen with probability proportional to the concentration parameter α. This representation is known as the Polya Urn representation of the Dirichlet Process [52].

The above is analogous to a Polya Urn scheme, and can be described as follows: we start with an empty urn. First, we draw a color from G_0, paint a ball with

the chosen color and place the ball into the urn. Then, at each subsequent step, we either select a new color from G_0, paint a new ball with that color and drop it into the urn, or we select a ball from the urn, observe its color and then return it into the urn along with another ball that we paint with the same color. We perform these two steps (the choice of selecting a color from G_0 or picking a ball from the urn) with probabilities $\frac{\alpha}{\alpha+n}$ and $\frac{n}{\alpha+n}$, respectively.

Polya Urn scheme produces an exchangeable sample.

Chinese Restaurant Process representation

The previous section shows that samples from a distribution that is drawn from the Dirichlet Process will have duplicates. This process induces clustering: a sample of size N will be partitioned into K clusters, where all observations in the same cluster will have the same parameter values. Introducing a latent variable c for each observation, we may denote this as $c_i = k$, which means that i^{th} observation is allocated to cluster k.

Given a sample of N observations, clustered into K unique clusters, the new observation y_{N+1} will be allocated to cluster c according to the following probability

$$P(c_{n+1} = c|c_1, c_2, \ldots, c_N, \alpha) = \frac{1}{\alpha + N}\big(\sum_{i=1}^{K} N_k \delta_{y_i^*}(c) + \alpha\delta_{y^*_{K+1}}(c)\big), \tag{3.16}$$

where $\delta_{y_i^*}(c) = 1$ if y_i^* is allocated to cluster c and 0 otherwise. The $(n+1)^{st}$ item will be assigned to one the existing clusters with probability proportional to the number of observations in the given cluster, or it will be assigned to a new cluster with the probability proportional to the concentration parameter α. This process exhibits the rich-gets-richer scheme, since larger clusters will keep getting larger

The above representation of the Dirichlet Process was first introduced by Pitman [53]. It is a metaphor for sitting arrangement at a Chinese restaurant, which is assumed to have infinite number of tables, each table being able to serve infinite number of customers, and all customers at the same table ordering the same dish. Customers are analogous to observations, the tables are analogous to clusters of observations, and table dish is analogous to a parameter of the cluster. So, a customer entering a restaurant will sit at one of the existing tables with probability proportional to the number of customers who have already chosen that table. A new customer may also choose to sit at a new table with probability proportional to α and order a new dish.

Joint distribution of seating arrangements (or partitions) is invariant to the order of customer arrivals, and therefore, the Chinese Restaurant Process induces an exchangeable distribution over partitions. Asymptotically, as $N \to \infty$, the number of occupied tables approaches to $\alpha \log(N)$ almost surely.

3.3 Dirichlet Process Mixture

A Dirichlet Process mixture is a type of mixture model in which the number of mixing components may be infinite, and the parameters of the model are distributed according to a Dirichlet Process. So, if y follows Dirichlet Process mixture distribution, and the underlying Dirichlet Process is $\mathrm{DP}(\alpha, G_0)$, then

$$f(y) = \sum_{k=1}^{\infty} \pi_k f(y|\theta_k^*)$$

where $\theta_k^* \sim G_0, k = 1, 2, \ldots, \infty$ and the mixing weights π_k are probability weights sampled according to the $DP(\alpha, G_0)$, as described in Eq.(3.13). An infinite mixture model does not mean that there are indeed an infinite number of mixture components. This is obviously impossible - we cannot have more components than we have data points. What we mean by "infinite" number of mixture components is that the upper limit on the number of mixture components is not fixed. Given any model derived from N data points, the number of mixture components may grow as more data points are added to the model.

Dirichlet Process is not well suited for modeling continuous distribution functions, primarily because realizations of probability distributions from it are discrete with probability one [47]. However, a continuous density function of y may be obtained as a general kernel mixture model

$$f(y|P) = \int K(y|\theta) dP(\theta),$$

where P is a mixing distribution and $K(\cdot)$ is a kernel function. Lo [54] has suggested that continuous distribution functions may be modeled as mixture distributions in which the Dirichlet Process is used as the prior of the mixing distribution. Therefore, by convolving a known kernel function with a Dirichlet Process, we obtain a Dirichlet Process mixture distribution, and it allows us to approximate any continuous density function to an arbitrary degree of precision.

The following is an equivalent representation of a Dirichlet Process mixture

model but in a hierarchical form

$$\begin{aligned} y_i &\propto F(y_i|\theta_i), \\ \theta_i &\propto G, \\ G &\propto \mathrm{DP}(\alpha, G_0), \end{aligned} \tag{3.17}$$

where the unique values of θ_i (these are θ_k^* in Eq.(3.3)) are independently and identically distributed from G_0.

It is sometimes useful to include an explicit allocation of observations to clusters, using the latent variable c_i for observation i, as described in Section 3.2.3. Then the Dirichlet Process mixture model may be represented as the limit of finite mixture of K components, where $K \to \infty$, as follows

$$\begin{aligned} y_i|c_i, \boldsymbol{\theta^*} &\propto F(y_i|\theta^*_{c_i}), \\ c_i|\boldsymbol{\pi} &\propto \mathrm{Multinomial}(\pi_1, \pi_2, \ldots, \pi_K), \\ \theta^*_{c_i} &\propto G_0, \\ (\pi_1, \pi_2, \ldots, \pi_K) &\propto \mathrm{Dirichlet}(\frac{\alpha}{K}, \ldots, \frac{\alpha}{K}), \end{aligned} \tag{3.18}$$

where $\boldsymbol{\pi}$ is the vector of mixing weights distributed according to the Dirichlet distribution, and α and G_0 are the parameters of the Dirichlet Process, as described in the previous sections.

3.4 Review of Stochastic Approximation Techniques

Many problems in Bayesian statistics come down to evaluating integrals in high dimensional spaces. For example, the posterior distribution of a parameter $\boldsymbol{\theta}$ given the data $\boldsymbol{y}$,

$$p(\boldsymbol{\theta}|\boldsymbol{y}) = \frac{p(\boldsymbol{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{\int p(\boldsymbol{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta}} = \frac{p(\boldsymbol{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{m(\boldsymbol{y})}$$

involves evaluation of the marginal likelihood of $\boldsymbol{y}$, $m(\boldsymbol{y}) = \int p(\boldsymbol{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta}$. While in some cases we may get away without evaluating $m(\boldsymbol{y})$, there are cases where we have to evaluate it. For example, under the quadratic loss function, the optimal estimator of some function $h(\cdot)$ of $\boldsymbol{\theta}$, is the expected value of $h(\boldsymbol{\theta})$ with respect to the posterior distribution of $\boldsymbol{\theta}$ given $\boldsymbol{y}$, i.e., the optimal Bayes estimator of $h(\boldsymbol{\theta})$ is $\int h(\boldsymbol{\theta})p(\boldsymbol{\theta}|\boldsymbol{y})d\boldsymbol{\theta}$. Another example where one may have to evaluate the marginal likelihood of $\boldsymbol{y}$ is in evaluating the Bayes factor when performing model comparison. Bayes factor is defined as

$$B(\boldsymbol{y}) = \frac{\int p_1(\boldsymbol{y}|\boldsymbol{\theta}_1)p_1(\boldsymbol{\theta}_1)d\boldsymbol{\theta}_1}{\int p_2(\boldsymbol{y}|\boldsymbol{\theta}_2)p_2(\boldsymbol{\theta}_2)d\boldsymbol{\theta}_2},$$

where $p_i(\cdot)$ is the probability in model M_i. Basically, an expectation of any function, with respect to the posterior distribution of $\boldsymbol{\theta}$, would require evaluating the expression $m(\boldsymbol{y})$.

Monte Carlo simulation techniques are a class of techniques that generate random samples in order to approximate integrals in high-dimensional spaces [55]. We briefly review two approximation techniques: simple Monte Carlo integration in which the generated random variables are independent and identically distributed, and Markov chain Monte Carlo techniques in which the generated random variables

are dependent.

3.4.1 Simple Monte Carlo Methods

In this section, we use univariate notation since simple Monte Carlo methods are most commonly applied in low dimensional spaces.

To see how simple Monte Carlo methods work, we consider an integral of function $h(y)$ for which there is no analytic solution. Assume we can decompose $h(y)$ into a product of $f(y)$ and $p(y)$, where $p(y)$ is a probability density function from which we can easily generate random samples. Then the integral of $h(y)$ is the expected value of $f(y)$ where the expectation is taken with respect to $p(y)$, i.e.,

$$\int h(y)dy = \int f(y)p(y)dy = E_p(f(Y)).$$

Simple Monte Carlo approximation methods are based on the idea that by taking a sample of N independent and identically distributed random variables Y_i from $p(y)$, $\boldsymbol{y} = \{y_1, y_2, \ldots, y_N\}$, we may approximate $E_p(f(Y))$ with the sample average $\hat{h}(\boldsymbol{y})$, where

$$\hat{h}(\boldsymbol{y}) = \frac{1}{N}\sum_{i=1}^{N} f(y_i).$$

By Strong Law of Large numbers, $\hat{h}(\boldsymbol{y})$ converges almost surely to $\int h(y)dy$. Furthermore, if the expectation of $f^2(Y)$ is finite with respect to $p(y)$, then the variance of $\hat{h}(\boldsymbol{y})$ may be approximated by

$$\mathrm{V}(\hat{h}(\boldsymbol{y})) = \frac{1}{N^2}\sum_{i=1}^{N}\left[f(y_i) - \hat{h}(\boldsymbol{y})\right]^2.$$

Evaluating the integral of $h(y)$ when $p(y)$ is one of the standard distributions is very easy, since most of software packages have implementation of generating random samples from $p(y)$. However, when $p(y)$ is not a standard distribution function or when it is not possible to sample directly from it (using standard techniques), then other sampling methods may need to be used. This may include inverse transform sampling, importance sampling [55], rejection sampling [55], or some other type of sampling that facilitate generation of independent sample.

3.4.2 Markov Chain Monte Carlo Methods

Markov chain is a type of a stochastic process that models a sequence of random variables. The sequence of random variables $\{\boldsymbol{\theta}_i : i \in T\}$, indexed by some index set T, describes the transition between different states of the underlying parameter space, and is formalized by the transition kernel. Transition kernel is the conditional probability of moving from state $(i-1)$ to state (i), denoted by $f(\boldsymbol{\theta}_i|\boldsymbol{\theta}_{i-1})$.

Markov chain is a stochastic process that satisfies the Markov property, which states that the probability of moving to the future state $\boldsymbol{\theta}_i$ depends only on the current state $\boldsymbol{\theta}_{i-1}$, that is,

$$f(\boldsymbol{\theta}_i|\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_{i-1}) = f(\boldsymbol{\theta}_i|\boldsymbol{\theta}_{i-1}), \text{for } i \geq 2.$$

The transition kernel $f(\boldsymbol{\theta}_i|\boldsymbol{\theta}_{i-1})$ along with the probability at the initial state $\boldsymbol{\theta}_1$, defines the joint distribution of any finite subset of the sequence of $\{\boldsymbol{\theta}_i : i \in T\}$, since

for a subset of size N we have

$$f(\boldsymbol{\theta}_1\, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_N) = f(\boldsymbol{\theta}_1) f(\boldsymbol{\theta}_2|\boldsymbol{\theta}_1) \ldots f(\boldsymbol{\theta}_N|\boldsymbol{\theta}_{N-1}).$$

There are several properties of a Markov chain that are particularly important for the study of Markov chain Monte Carlo (MCMC) methods. First, a Markov chain is said to be Harris recurrent if, given a particular state, the probability that the chain visits it infinitely often is one, regardless of what the initial state was. Second, a Markov chain is periodic if it returns to an initial state at regularly spaced intervals; otherwise, it is aperiodic. A chain is said to be ergodic if it is aperiodic and Harris recurrent. Finally, a Markov chain is said to be stationary, if its transition probability does not depend on the particular value of the index.

Robert [55] defines Markov chain Monte Carlo as any method that produces an ergodic Markov chain whose stationary distribution is the target distribution of interest. This target distribution is most often the posterior distribution of the model parameters.

In the above notation, a state $\boldsymbol{\theta}_i$ is an arbitrary state in a Markov chain. However, when this state represents a value of a parameter $\boldsymbol{\theta}$ at iteration i, then we denote it by $\boldsymbol{\theta}^{(i)}$. To refer to a particular component, say k^{th} component, of vector $\boldsymbol{\theta}$ at iteration i, we use notation $\boldsymbol{\theta}_k^{(i)}, 1 \leq k \leq K$.

Given a chain with N samples of parameter $\boldsymbol{\theta}$, and a real-valued function $h(\boldsymbol{\theta})$, we define the ergodic average of $h(\boldsymbol{\theta})$ as

$$\hat{h}(\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N} h(\boldsymbol{\theta}^{(i)}).$$

The key theorem in Markov chain Monte Carlo methods is the Ergodic Theorem [56] which states that if the chain is ergodic with stationary distribution f and with finite first moment of $h(\boldsymbol{\theta})$ with respect to the stationary distribution f, then the ergodic average of $h(\boldsymbol{\theta})$ converges almost surely to expected value of $h(\boldsymbol{\theta})$ under the same distribution, i.e.,

$$\hat{h}(\boldsymbol{\theta}) \longrightarrow \mathrm{E}_f(h(\boldsymbol{\theta})) \text{ as } N \to \infty \text{ a.s.}$$

In the remainder of this section, we describe two of the most commonly used families of MCMC sampling algorithms. We describe these methods in the context of sampling from the posterior distribution of $\boldsymbol{\theta}$.

3.4.3 Gibbs Sampling

In most practical problems, a parameter vector $\boldsymbol{\theta} = \{\theta_1, \theta_2, \ldots, \theta_K\}$ may be high dimensional. Gibbs sampling [57] method draws samples from the posterior distribution of $\boldsymbol{\theta}$ by iteratively sampling one parameter component $\theta_i, 1 \leq i \leq K$, at a time. A parameter θ_i is sampled from its conditional distribution (denoted by $f_i(\theta_i|\cdot)$ below), assuming all remaining parameters (denoted by $\boldsymbol{\theta_{-i}}$) are constant. Therefore, the sample of $\boldsymbol{\theta}$'s is built one parameter component at a time, as the following algorithm shows:

Initialize a starting value of $\boldsymbol{\theta}$ as $\boldsymbol{\theta}^{(0)}$;

for *i=1 to M* **do**

Sample $\boldsymbol{\theta}^{(i)}$ as follows:

Sample $\theta_1^{(i)} \sim f_1(\theta_1|\boldsymbol{\theta}_{-1}^{(i-1)}, \boldsymbol{y})$;

Sample $\theta_2^{(i)} \sim f_2(\theta_2|\boldsymbol{\theta}_{-2}^{(i-1)}, \boldsymbol{y})$;

...

Sample $\theta_K^{(i)} \sim f_k(\theta_K|\boldsymbol{\theta}_{-K}^{(i-1)}, \boldsymbol{y})$;

end

Algorithm 1: Gibbs sampling from $f(\boldsymbol{\theta}|\boldsymbol{y})$

Algorithm 1 shows pseudo code steps of sampling from posterior distribution using Gibbs algorithm. It is immediately clear that in order to be able to use Gibbs sampling, one needs to derive conditional distributions of parameters of interest and these conditional distributions have to have a known form.

Sampling one parameter at a time may be very inefficient [58], especially when the parameters are highly correlated. A common workaround to improving efficiency is to sample multiple parameters at a time - this technique is known as Blocked Gibbs sampling and has been known to improve convergence [59]. Integrating out some parameters is also common - this technique leads to Collapsed Gibbs sampling algorithms.

Gibbs sampling is often the first choice of Markov chain Monte Carlo sampling methods. However, many posterior distributions in practice do not have a known conditional distribution of either some or any parameters of interest, in which

case, other sampling methods need to be used. One of the more common alternative sampling methods is the Metropolis-Hastings method, described in the next section.

3.4.4 Metropolis-Hastings Sampling

Similar to Gibbs sampling method, the Metropolis-Hasting method is not a single method but rather a class of methods. These methods are based on papers of Metropolis [60] and Hastings [61]. They sample parameters from the distribution using the full joint density of the target distribution. To draw samples from the target distribution f($\boldsymbol{\theta}|\boldsymbol{y}$), we use a proposal distribution $q(\boldsymbol{\theta})$ to draw a candidate parameter $\boldsymbol{\theta}^{(cand)}$ representing a state where the Markov chain may move next. Then we move probabilistically to the new state.

Initialize starting value at $\boldsymbol{\theta}^{(0)} \sim q(\boldsymbol{\theta})$;

for *i=1 to M* **do**

1: Propose $\boldsymbol{\theta}^{(cand)} \sim q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(i-1)})$;

2: Calculate acceptance probability as:

$\alpha(\boldsymbol{\theta}^{(cand)}, \boldsymbol{\theta}^{(i-1)}) = \min\big(1, \frac{q(\boldsymbol{\theta}^{(i-1)}|\boldsymbol{\theta}^{(cand)}) f(\boldsymbol{\theta}^{cand}|\boldsymbol{y})}{q(\boldsymbol{\theta}^{(cand)}|\boldsymbol{\theta}^{(i-1)}) f(\boldsymbol{\theta}^{(i-1)}|\boldsymbol{y})}\big)$;

3: Sample $u \sim \text{Uniform}(0, 1)$;

if $u \leq \alpha(\boldsymbol{\theta}^{(cand)}, \boldsymbol{\theta}^{(i-1)})$ **then**

4: Accept proposal: $\boldsymbol{\theta}^{(i)} \leftarrow \boldsymbol{\theta}^{(cand)}$;

else

5: Reject proposal: $\boldsymbol{\theta}^{(i)} \leftarrow \boldsymbol{\theta}^{(i-1)}$;

end

end

Algorithm 2: Metropolis-Hastings sampling from $f(\boldsymbol{\theta}|\boldsymbol{y})$

The proposal distribution is chosen so that is has the same support as the target distribution and is easy to sample from. Its variance controls the acceptance rate and is often tuned [62] so that the acceptance rate is between 20% and 50% - a suggested rate in practical applications [63], [64].

There is huge literature on many types of Metropolis-Hastings algorithms. Some (very simple) examples include: symmetric Metropolis-Hastings algorithms in which the proposal distribution is symmetric, Random Walk algorithms in which the current state is perturbed by an independent and identically distributed offset from the proposal distribution, independence chains in which the proposal distribution is independent of the current state, and many others. Common proposal distributions

include the Normal distribution [65], t-distribution [66], or an approximation of the posterior distribution using the normal distribution [63], [67], [68].

It is also common to use a combination of Gibbs and Metropolis-Hastings algorithms: those parameters which have a known conditional distribution are sampled using the Gibbs sampling, and those that do not, are sampled using more powerful sampling methods, including some type of Metropolis-Hastings algorithm.

3.5 Sampling in Dirichlet Process Mixture Models

In this section, we describe several common sampling methods in Dirichlet Process mixture models, which are either based on Gibbs sampling or Metropolis-Hastings sampling method.

After having observed the data $\boldsymbol{y} = \{y_1, y_2, \ldots, y_N\}$, we would like to derive the posterior distribution of $\boldsymbol{\theta_c}$, $f(\boldsymbol{\theta_c}|\boldsymbol{y})$, where $\boldsymbol{c}$ is the allocation vector as defined in Section 3.2.3. We basically want to sample from the conditional distribution of θ_{c_i}, given the remaining $\boldsymbol{\theta}$ parameters (denoted by $\boldsymbol{\theta}_{-c_i}$), and given the observed data $\boldsymbol{y}$ and the other model parameters - these are not shown in the expressions below.

The posterior of $\boldsymbol{\theta_c}$ is just the product of its prior and the likelihood

$$f(\boldsymbol{\theta_c}|\boldsymbol{y}) \sim f(\boldsymbol{\theta_c}) f(\boldsymbol{y}|\boldsymbol{\theta_c}).$$

The first part of the above expression may be written as the product of conditional distributions of its unique parameters ($\theta^*_{c_i}$), given all other model parameters, which, due to the exchangeability of $\boldsymbol{\theta}$ and Polya Urn representation, as described in Section

3.2.3 and Eq. (3.15), can be written in the following form

$$f(\theta_{c_i}|\alpha, \boldsymbol{y}, G_0) \sim \alpha G_0 + \sum_{k=1}^{K} N_k \delta_{\theta^*_{c_k}}. \tag{3.19}$$

Then multiplying the above priors with the likelihood $f(\boldsymbol{y}|\boldsymbol{\theta_c})$ we get the posterior distribution of θ_{c_i} as

$$\theta_{c_i}|\boldsymbol{\theta_{-c_i}}, \boldsymbol{y} \propto \kappa \left\{ \sum_{j \neq i} f(y_i|\theta_{c_j}) \delta_{\theta_{c_j}}(\theta_{c_i}) + \alpha \Big(\int f(y_i \boldsymbol{\theta}) dG_0(\boldsymbol{\theta}) \Big) H_i(\theta_{c_i}) \right\}, \tag{3.20}$$

where κ is the normalizing constant, α is the concentration parameter of the Dirichlet Process $G(\alpha, G_0)$, and $H_i(\theta)$ is the posterior distribution of the parameter θ given the prior distribution G_0 and a single observation y_i. The above method was used by Escobar [69] and Escobar [70], and while easy to implement in conjugate case, it suffers from poor mixing and slow convergence.

Sampling the concentration parameter α of the Dirichlet Process prior is explained in the next chapter.

The above expression Eq.(3.20) shows how to sample parameters associated with each observation from their posterior distribution. Using similar approach with model (3.18), we can sample allocation variables c_i associated with each observation i, instead of parameters associated with those observations, as we have done above. Then instead of using Polya Urn representation of the Dirichlet Process to formulate the prior of parameters $\boldsymbol{\theta}$, we use the Chinese Restaurant Process to formulate the prior of allocation vector $\boldsymbol{c}$, $p(\boldsymbol{c})$, and derive the posterior distribution of c_i given all other model parameters [71], [72]. The expressions of these posterior distributions

are as follows

$$p(c_i = c|\boldsymbol{c}_{-i}, y_i, \boldsymbol{\theta}_{-c_i}) = \begin{cases} \kappa \frac{n_{-i,c}}{N-1+\alpha} f(y_i|\theta_c^*), & \text{if } c = c_j \text{ for some } j \neq i; \\ \kappa \frac{\alpha}{N-1+\alpha} \int f(y_i|\theta) dG_0(\theta), & \text{if } c_i \neq c_j \text{ for all } j \neq i. \end{cases} \tag{3.21}$$

Similar to Eq.(3.20), the above posterior probabilities are very easy to derive when the prior distribution is conjugate to the likelihood. Few methods have been proposed to handle non-conjugate cases, the most common of which is Algorithm 8 in [73]. We use this method in this thesis and provide detailed explanation of it in the next chapter.

Exploring the posterior sample space is challenging. Samplers may get stuck in local maximum and fail to properly explore the whole space. To break from the local maximum, extensions to the above algorithms have been proposed in both conjugate [74] and non-conjugate cases [75]. However, these methods may not be very efficient in high dimensional spaces.

Other common sampling methods include slice sampling [76], [77], retrospective sampling [78], or methods based on approximating the stick breaking representation of the Dirichlet Process [79], [80].

3.6 Summary

This section provides a short overview of the Bayesian nonparametric statistics with the focus on the Dirichlet Process and Dirichlet Process mixture models. We first describe advantages and rationale for using the Bayesian nonparametric methods, and then review the basic distributions that are related to the Dirichlet Process.

We describe the Dirichlet Process, including its basic properties and advantages, and disadvantages leading to the Dirichlet Process mixture models. We then do a short overview of different representations of the Dirichlet Process and show how they are used in sampling in Dirichlet Process mixture models. We also briefly review the basic concepts in simple Monte Carlo and Markov chain Monte Carlo approximation techniques.

Chapter 4

Clustering Profiles in Generalized Linear Mixed Models

In this chapter, we first review the literature related to clustering longitudinal profiles. The most relevant work related to our thesis includes clustering profiles in Bayesian nonparametric settings with continuous response only. We propose a novel method, called the Generalized linear mixed model clustering using Dirichlet Process (GLMM-DP), which allows one to cluster profiles in which the distribution of an outcome is any member of the exponential family. We model the response using a mixture of GLMM models without pre-specifying the number of mixture of components. The number of mixture components and complexity of the model is fully data-driven, leading to simple non-linear models as in GLMM when only one mixture component is produced, and non-linear models in much broader sense when more than one mixture component is produced. After introducing the GLMM-DP method, we derive details of the posterior distributions of model parameters. We conclude the chapter with the presentation of a label switching solution, which is needed in order to be able to make inferences at a component level. A performance evaluation through a simulation study will be presented in the following chapter.

4.1 Introduction to clustering

Clustering is a process of organizing a set of objects into groups so that objects within the same group exhibit high degree of similarity while objects between different groups are as dissimilar as possible. This problem has been extensively studied in both statistics and computer science, and is often known as segmentation, numerical taxonomy or unsupervised classification [81]. There is a vast literature on clustering [82]. It is supported by solid implementations in various software packages, and has numerous applications in many areas in industry, including market research [83], astronomy [84], biomedical research [85], and social media [86].

There are many ways to group clustering methods. Clustering methods may be divided into hierarchical and partitioning methods [87]. Hierarchical clustering methods are tree-based methods that describe how the final clustering is obtained. They can be either agglomerative or divisive. Agglomerative methods start by placing each object in its own cluster and then recursively merge two closest clusters (creating a parent node in the tree hierarchy) until only one cluster is left. Divisive clustering methods, on the other hand, place all objects into one cluster and then recursively partition clusters (creating child nodes in the tree) until no further partitioning is possible (based on some objective function). K-means algorithm [3] is a well-known example of this type of clustering. A close relative to it that we use in this thesis is the K-medoids method [4]. The K-means method clusters objects around centroids, where a centroid is a center of a set of objects, which does not have to correspond to any particular object in the set. The K-medoids

method, on the other hand, clusters objects around medoids, which are objects from the set whose dissimilarity with the other objects in the same cluster is minimal.

Most of clustering methods were originally developed for cross-sectional data, where observations are assumed to be independent. In longitudinal studies, or more generally, in clustered data studies, independence does not hold, which makes most of early clustering methods not suitable for these types of studies. Attempts have been made to adopt these methods in longitudinal settings [5], [6].

Model-based clustering methods assume that data are observed from a heterogeneous population, and try to model such a population using mixture models. Each data item is assumed to come from one (homogeneous) subpopulation. For example, assuming that there are K different subpopulations, each subpopulation being modeled by the density function $f(y|\theta_k)$, the mixture model may be written as

$$f(y|\Theta) = \sum_{k=1}^{K} \pi_k f(y|\theta_k), \tag{4.1}$$

where Θ is the set of all model parameters, $\Theta = \{\theta_1, \ldots, \theta_K, \pi_1, \ldots, \pi_k\}$. In the above case, we say that Y has a finite mixture distribution, consisting of K mixture components, each component used to model one homogeneous subpopulation. Parameters $\theta_k, 1 \leq k \leq K$, index the component mixture distribution functions, while the mixture weight π_k denotes the probability that a data item comes from the k^{th} subpopulation, and hence $\pi_k > 0$. Since a point must come from one of the existing subpopulations, we also have $\sum_{k=1}^{K} \pi_k = 1$. Model-based clustering may produce results that are practically more meaningful than distance-based methods [88].

In many problems, the number of mixture components is known in advance. However, in many other problems, the number of components is not known in advance. In such cases, it is common to fit multiple models, with different numbers of mixture components, and then compare model fits in order to find a model with the most plausible number of components.

Model-based clustering methods and are well suited for clustering both cross-sectional and longitudinal data. In longitudinal data, multiple observations from the same individual may be stacked up as a single multivariate response, and then multivariate statistics may be used to model the data. The advantage of multivariate statistics is that it allows one to capture association between responses from the same individual, while treating responses from different individuals still independent. For example, assuming that $\boldsymbol{y}$ contains n responses from the same individual, with $\boldsymbol{y} \in \mathbb{R}^n$, we may model it using the multivariate normal distribution with parameters $\boldsymbol{\theta} = (\boldsymbol{\mu}, \boldsymbol{\Sigma})$, i.e.,

$$f(\boldsymbol{y}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^n |\boldsymbol{\Sigma}|}} \exp\left\{ -\frac{1}{2} (\boldsymbol{y} - \boldsymbol{\mu})^t \boldsymbol{\Sigma}^{-1} (\boldsymbol{y} - \boldsymbol{\mu}) \right\}.$$

Some constraints of the covariance matrix are often imposed. For example, using the modified Cholesky decomposition [8], for a given $\boldsymbol{\Sigma}$, we may find a unique lower triangular matrix $\boldsymbol{T}$ and a unique diagonal matrix $\boldsymbol{D}$, such that $\boldsymbol{\Sigma}^{-1} = \boldsymbol{T}^t \boldsymbol{D}^{-1} \boldsymbol{T}$. These matrices ($\boldsymbol{T}$ and $\boldsymbol{D}$) have nice interpretations in longitudinal data. For observations that are assigned to the same cluster, $\boldsymbol{T}$ represents association between observations taken at different time points. The matrix $\boldsymbol{D}$ represents association between observations taken at the same time. McNicholas [9] builds on these properties and suggests eight different types of covariance structures, based on

different restrictions of matrices $\boldsymbol{T}$ and $\boldsymbol{D}$ (constrained vs. unconstrained). This method is implemented in software [89]. McNicholas [90] further extends the task of clustering longitudinal data with missing observations.
Eigen-decomposition is another common type of decomposition of covariance matrix [10], [11] and obviously leads to different types of constraints of new matrices.

Most of the methods assume that the distribution of mixture components is multivariate normal. The multivariate t-distribution is sometimes used as well [91], [92], though less often.

A mixture model may have any model associated with a mixture component, not only a distribution function as alluded above. It could be a linear model [11] or linear mixed model [93], [94], [95]. It could also be a non-linear model [96]. It could be as advanced as any statistical model may be.

The above clustering methods assume that the number of mixture components is either known in advance, or they fit multiple models, each with a different number of components, and then perform a challenging task of model comparisons to find the most plausible model. The most common method of model comparison is performed using the Bayesian information criterion (BIC) [97] and Akaike's Information Criterion (AIC) [98]. Attempts to remove the restriction of a known number of mixture components has led to development of new methods.

Methods based on Dirichlet Process mixture models do not assume that the number of mixture components is known in advance. Yan He [12] models the response variable within the linear mixed models setting, and clusters profiles

based on similarity between profile parameters. These two steps are performed simultaneously. Profiles with similar parameters are assigned to the same cluster, where similarity is determined by the underlying Dirichlet Process prior. Sun [14] proposes a similar method for clustering gene expression profiles, while applying the factor analysis to reduce dimensionality of parameter space (which can be quite large as more and more gene profiles are analyzed). The most common estimation method in mixture models is the Expectation-Maximization (EM) method [31], while MCMC techniques are methods of choice in Bayesian domain. Heinzl [99] however, models random effects in linear mixed models using Dirichlet process but uses the EM method for parameter estimation. This is possible with the stick-breaking representation of the Dirichlet process which can be truncated after certain precision in its representation is achieved, as shown by [79].

The main advantage of using Bayesian nonparametric statistics in grouping clustered or longitudinal data is that we do not have to set the number of mixture components in advance, and that way we can avoid selecting the most appropriate number of mixture components [7]. We let the data determine the complexity of the model (number of components, component parameters and other common parameters). This does not mean that all model parameters can always be easily obtained in Bayesian nonparametric statistics. Marginal parameters or parameters that are common for all components can be easily estimated; however, estimating component level parameters may still be quite challenging. This is due to the label switching problem that we discuss at the end of this chapter. In Bayesian nonparametric statistics, we build a single model only once, and then apply a label switching solution to determine the most plausible number of mixture components and their parameters. Bayesian parametric statistics, on the other hand, may build

many different models before choosing one that best fits the data.

4.2 Profile Clustering in Generalized Linear Mixed Models

In this section we introduce the GLMM-DP method and describe how it differs from other methods in the literature. We then describe how to estimate model parameters. This includes specifying prior distributions of parameters and deriving their posterior distributions.

4.2.1 Model Description

We assume that a response variable Y, observed at two levels ($i = 1, 2, \ldots, N, j = 1, 2, \ldots, n$) has a distribution that is a member of an exponential family. For each observation Y, we have two sets of predictors, X and Z, and correspondingly two sets of parameters, the first of which we refer to as "fixed" effects parameters and the second as "random" effects parameters. Given a link function $h(\cdot)$, we may define the linear predictor as

$$h(E(Y_{ij}|\boldsymbol{\beta}, b_i)) = \eta_{ij} = \boldsymbol{X}_{ij}^t\boldsymbol{\beta} + \boldsymbol{Z}_{ij}^t\boldsymbol{b}_i. \tag{4.2}$$

The linear predictor in Eq.(4.2) and the following distributional assumptions complete the model specification

$$
\begin{aligned}
y_{ij}|\boldsymbol{\beta}, \boldsymbol{b}_i, \nu &\propto \exp\left\{\frac{y_{ij}\eta_{ij} - q(\eta_{ij})}{\nu} + k(y_{ij}, \nu)\right\}, \\
\boldsymbol{b}_i|G &\propto G, \\
G|\alpha, G_0 &\propto \mathrm{DP}(\alpha, G_0), \\
\boldsymbol{\beta}|\mu_\beta, \boldsymbol{\Sigma}_{\boldsymbol{\beta}} &\propto \mathrm{MVN}(\boldsymbol{\mu}_{\boldsymbol{\beta}}, \boldsymbol{\Sigma}_{\boldsymbol{\beta}}), \\
\nu^{-1}|\alpha_\nu, \beta_\nu &\propto \mathrm{Gamma}(\alpha_\nu/2, \beta_\nu^{-1}/2), \\
\beta_\nu^{-1} &\propto \mathrm{Gamma}(\alpha_{\beta_\nu}, \beta_{\beta_\nu}), \\
\alpha &\propto \mathrm{Gamma}(a_\alpha, b_\alpha), \\
G_0 &\equiv \mathrm{MVN}(\boldsymbol{\mu}_{\boldsymbol{b}}, \boldsymbol{\Sigma}_{\boldsymbol{b}}),
\end{aligned}
\tag{4.3}
$$

where $\mathrm{DP}(\alpha, G_0)$ is the Dirichlet Process with the concentration parameter α and the base distribution G_0, as described in Section 3.2.2. Placing a Dirichlet Process prior on the random effects parameters $\boldsymbol{b}_i$ ensures that there would be duplicates among the parameters, resulting in clustering of profiles that share the same random effects based on their cluster assignment.

In the model specified by Eq.(4.3), profiles are grouped by their random effects parameters. This means that all profiles allocated to the same cluster have the same random effect parameter. Therefore, knowing a random effect parameter of a profile is the same as knowing a random effect parameter of a mixture component to which it is assigned. Similar modeling design step was adopted in [12] though in linear mixed models setting. An extension to this method in which random effects may be grouped without enforcing that all of profiles assigned to the same cluster share the same value of the parameter, is discussed in the last chapter of this thesis.

As in Komarek [100], we place the Gamma(α, β) prior on the dispersion parameter ν, and gamma prior on the inverse of its rate hyper-parameter, where Gamma(α, β) is the standard Gamma distribution with the density function given by

$$f(y|\alpha, \beta) = \frac{\beta^{\alpha}}{\Gamma(\alpha)} y^{\alpha-1} e^{-\beta y}. \tag{4.4}$$

4.2.2 Fixed vs. Random Effects Parameters

In frequentist statistics, fixed effect parameters are assumed constant and their estimation is of direct interest to the researcher, while random effects are considered a random sample from the larger population of values and have a distribution associated with them [21]. In linear mixed models and generalized linear mixed models, random effects may also be of direct interest to the researcher (when subject-based inference is the objective of the study).

In Bayesian statistics, all parameters are considered random and are described with a probability distribution. Our interpretation is similar to that of Kreft [101]: fixed effect parameters are parameters that are common across all units, while random effect parameters may vary across units. Therefore, $\boldsymbol{\beta}$ assesses the effect of predictors at the population level and $\boldsymbol{b}_i$ assesses the effect of predictors at the lower-unit level (i^{th} individual). Hence we have only one vector of $\boldsymbol{\beta}$ parameters, while there may as as many as N different vectors in $\boldsymbol{b} = \{\boldsymbol{b}_1, \boldsymbol{b}_2, \cdots, \boldsymbol{b}_N\}$.

Identifiability is a property of a model which guarantees that the parameters of the model, or any function of them, may be correctly estimated from the data, to an arbitrary level of precision assuming that we may obtain an arbitrary amount of

data. For example, in case of linear models, if the expected means of two models are the same, then the parameters of those models must be the same [102]. Parameters in non-identifiable models cannot be correctly estimated regardless of the amount of data one may have.

In order to avoid non-identifiability issues in models with both fixed and random effects parameters, it is common to impose a constraint in which the column space of the covariate matrix $\boldsymbol{Z}$ is a subset of the column space of the matrix $\boldsymbol{X}$. Random effects in such models are interpreted as a deviation (at unit level) from the population mean. We do not impose such constraints. Instead, we require that the two column spaces must have no (non-zero) elements in common. This means that predictors used to explain the outcome $\boldsymbol{Y}$ at two different levels must be different. Komarek [100] uses fixed and random effects parameters in exactly the same way.

In most general linear mixed models and generalized linear mixed models, random effects are used as a device to model the correlation structure between observations within the same level (such as individuals), and are not to be estimated (other than their variance). The objective of our model is to simultaneously estimate model parameters at both levels and cluster profiles based on random effects estimates. This is a natural approach in Bayesian statistics, in which random effects parameters are treated as unknown parameters like fixed effects parameters [103]. There is no need to integrate them out.

4.2.3 Parameter Estimation

We first re-parametrize the model to bring it into a more convenient form. Let the vector $\boldsymbol{c} = (c_1, c_2, \ldots, c_N)$ be a vector of allocation of individual profiles to mixture

components, so that $c_i = k$ means that individual profile i is allocated to mixture component k. Further, given that the parameters $\boldsymbol{b}_i$ in model (4.3) have duplicates, let $\boldsymbol{\phi}$ be the vector of their unique values, i.e., $\boldsymbol{\phi} = \{\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, \ldots, \boldsymbol{\phi}_r\}$, where $r \leq N$. Then model (4.3) may be written as

$$\begin{aligned}
y_{ij}|c_i, \boldsymbol{\phi}, \boldsymbol{\beta} &\propto \mathrm{F}(y_{ij}|\boldsymbol{\beta}, \boldsymbol{\phi}_{c_i}) \\
c_i|\boldsymbol{p} &\propto \text{Multinomial}(p_1, \ldots, p_K), \\
p_1, \ldots, p_K &\propto \text{Dirichlet}\Big(\frac{\alpha}{K}, \ldots, \frac{\alpha}{K}\Big), \\
\boldsymbol{\beta} &\propto \text{MVN}(\boldsymbol{\mu}_{\boldsymbol{\beta}}, \boldsymbol{\Sigma}_{\boldsymbol{\beta}}), \\
\boldsymbol{\phi}_{c_i} &\propto G_0, \\
\nu^{-1}|\alpha_\nu, \beta_\nu &\propto \text{Gamma}(\alpha_\nu/2, \beta_\nu^{-1}/2), \\
\beta_\nu^{-1} &\propto \text{Gamma}(\alpha_{\beta_\nu}, \beta_{\beta_\nu}), \\
\alpha &\propto \text{Gamma}(a_\alpha, b_\alpha), \\
G_0 &\equiv \text{MVN}(\boldsymbol{\mu}_{\boldsymbol{b}}, \boldsymbol{\Sigma}_{\boldsymbol{b}}),
\end{aligned} \tag{4.5}$$

with $K \to \infty$. In the above model, $F(y_{ij}|\boldsymbol{\phi}_{c_i}, \boldsymbol{\beta})$ represents any member of the exponential distribution function, indexed by fixed effects $\boldsymbol{\beta}$ and random effects parameters $\boldsymbol{\phi} = \{\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, \ldots, \boldsymbol{\phi}_r\}$.

In model defined in Eq.(4.5), let the set of all parameters be $\Theta = \{\alpha, \boldsymbol{\beta}, \boldsymbol{\phi}, \boldsymbol{c}, \nu, \beta_\nu\} = \{\alpha, \boldsymbol{\beta}, \boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_r, c_1, \ldots, c_N, \nu, \beta_\nu\}$. The posterior distribution is expressed as the product of the prior distribution $P(\Theta)$ and the likelihood function $L(\Theta) = F(\boldsymbol{y}|\boldsymbol{\beta}, \boldsymbol{\phi})$

$$\begin{aligned}
P(\Theta|\boldsymbol{y}) \propto& P(\Theta) \times L(\Theta) \\
\propto& P(\alpha, \boldsymbol{\beta}, \boldsymbol{\phi}, \boldsymbol{c}) \times L(\Theta) \\
\propto& P(\alpha) \times P(\boldsymbol{c}|\alpha) \times P(\boldsymbol{\beta}) \times P(\boldsymbol{\phi}|c) \times P(\nu|\alpha_\nu, \beta_\nu) \times P(\beta_\nu) \times L(\Theta).
\end{aligned} \tag{4.6}$$

In the above expression, the likelihood function $L(\Theta)$ is

$$\begin{aligned} L(\Theta) =& P(\boldsymbol{y}|\Theta) \\ =& P(\boldsymbol{y}|\alpha, \boldsymbol{\beta}, \boldsymbol{\phi}, \boldsymbol{c}) \\ =& \prod_{i=1}^{N} P(\boldsymbol{y}_i|\boldsymbol{\beta}, \boldsymbol{\phi}_{c_i}) \\ =& \prod_{i=1}^{N}\prod_{j=1}^{n} P(y_{ij}|\beta, \boldsymbol{\phi}_{c_i}) \\ =& \prod_{i=1}^{N}\prod_{j=1}^{n} \exp\left\{\frac{y_{ij}\eta_{ij} - q(\eta_{ij})}{\nu} + k(y_{ij}, \nu),\right\} \end{aligned} \tag{4.7}$$

where the linear predictor has been re-parameterized, i.e., $\eta_{ij} = \boldsymbol{X}_{ij}^t\boldsymbol{\beta} + \boldsymbol{Z}_{ij}^t\boldsymbol{\phi}_{c_i}$. Using the notation in Section 1.1, the above expression becomes

$$\begin{aligned} L(\Theta) =& \prod_{i=1}^{N}\prod_{j=1}^{n} \exp\left\{\frac{y_{ij}\eta_{ij} - q(\eta_{ij})}{\nu} + k(y_{ij}, \nu)\right\} \\ =& \prod_{i=1}^{N} \exp\left\{\frac{\boldsymbol{y_i} \times \boldsymbol{\eta_i} - \mathbf{1} \times q(\boldsymbol{\eta_i})}{\nu} + \mathbf{1} \times k(\boldsymbol{y_i}, \nu)\right\} \\ =& \exp\left\{\frac{\boldsymbol{y}^t \times \boldsymbol{\eta} - \mathbf{1}^t \times q(\boldsymbol{\eta})}{\nu} + \mathbf{1}^t \times k(\boldsymbol{y}, \nu)\right\}, \end{aligned} \tag{4.8}$$

where $\mathbf{1}$ is a unit vector of dimension of either N or n, depending on the context.

Given the expression (4.7) and (4.8), we need to sample from the posterior distribution

$$\begin{aligned} P(\Theta|\boldsymbol{y}) \propto& P(\Theta) \times L(\Theta) \\ \propto& P(\alpha) \times P(\boldsymbol{c}|\alpha) \times P(\boldsymbol{\beta}) \times P(\boldsymbol{\phi}|\boldsymbol{c}) \times P(\nu|\alpha_\nu, \beta_\nu) \times \\ & P(\beta_\nu) \times \exp\left\{\frac{\boldsymbol{y}^t \times \boldsymbol{\eta} - \mathbf{1}^t \times q(\boldsymbol{\eta})}{\nu} + \mathbf{1}^t \times k(\boldsymbol{y}, \nu)\right\}. \end{aligned} \tag{4.9}$$

4.2.4 Choosing Prior Distributions

For simplicity, we assume the parameters of prior distribution of both $\boldsymbol{\beta}$ and $\boldsymbol{b}$ are known. More specifically, we set $\boldsymbol{\mu_b} = \mathbf{0}$, and also we set the covariance matrices $\boldsymbol{\Sigma_\beta}$ and $\boldsymbol{\Sigma_b}$ to be diagonal matrices with very large variances. This will allow us to explore the large support of both parameters. So, for $\boldsymbol{\beta}$, we have

$$P(\boldsymbol{\beta}|\boldsymbol{\mu_\beta}, \boldsymbol{\Sigma_\beta}) \propto |\boldsymbol{\Sigma_\beta}|^{-\frac{1}{2}} \times \exp\Big\{ -\frac{1}{2}(\boldsymbol{\beta} - \boldsymbol{\mu_\beta})^t \boldsymbol{\Sigma_\beta}^{-1}(\boldsymbol{\beta} - \boldsymbol{\mu_\beta})\Big\}. \tag{4.10}$$

The prior of $\boldsymbol{b}$ has also multivariate normal distribution with mean $\boldsymbol{\mu_b}$ and covariance matrix $\boldsymbol{\Sigma_b}$.

It is common to use the gamma distribution for the prior of the concentration parameter α of the Dirichlet Process. For simplicity, we assume that the parameters of this distribution are known, i.e., $\alpha \propto \text{Gamma}(a_\alpha, b_\alpha)$, where a_α, b_α are fixed. For our simulation study (next chapter) we set both parameters to 1. This would draw, on average, a concentration parameter that is close to 1, resulting in smaller number of mixture components. In practice, however, one may try building the model with different values of a_α and b_α and then choose the values that produce the most plausible number of components.

For the prior and hyperprior parameters of dispersion parameter ν in Eq.(4.5), we set $\alpha_\nu = 2$, $\alpha_{\beta_\nu} = 0.2$ and $\beta_{\beta_\nu} = 5$. This corresponds to a weakly informative prior and has been used by other authors [100].

4.3 Sampling from Posterior Distributions

In this section, we describe how to sample parameters from the posterior distribution outlined in (4.9).

4.3.1 Sampling concentration parameter of Dirichlet Process

The number of distinct values of random effects, which in the paragraph surrounding Eq.(4.5) we denote by r, is random and has its own distribution, derived early on by Antoniak [104]. The posterior distribution of α depends on its prior distribution and the likelihood but only through the number of distinct values of $\boldsymbol{b}_i$ random effects (r of them). We use the method of West [105] to sample α, which is a mixture of two Gamma distributions as follows

$$\begin{aligned} P(\alpha|x,r) \sim & \pi_x \times \text{Gamma}(a_\alpha + r, b_\alpha - \log(x)) \\ & + (1-\pi_x) \times \text{Gamma}(a_\alpha + r - 1, b_\alpha - \log(x)), \end{aligned} \tag{4.11}$$

where x is an auxiliary variable with values between 0 and 1. The mixture weights are determined from the following expression

$$\frac{\pi_x}{1-\pi_x} = \frac{a_\alpha + r - 1}{N(b_\alpha - \log(x))}. \tag{4.12}$$

All the remaining parameters are fixed.

4.3.2 Sampling allocation variables

We sample allocation variables using the posterior distribution $\text{P}(\boldsymbol{c}|\boldsymbol{y}) \propto P(\boldsymbol{c}|\alpha) \times L(\Theta)$. The allocation variables $c_1, \ldots, c_N$ are not independent, and each has to be sampled conditionally on other allocation parameters.

Without loss of generality, we may assume that the allocation vector $\boldsymbol{c}$ contains values between 1 and r, where r is the number of distinct mixture components, each component having at least one profile allocated to it. To sample allocation variables, we follow Algorithm 8 of Neal [73]. Profiles are allocated to mixture components one at a time. After each allocation, the number of mixture components may remain the same, it may increase by one or it may decrease by one. Let $\boldsymbol{c}_{-1}$ be the vector of all component allocations as in $\boldsymbol{c}$ but excluding the component i. Given the current r distinct mixture components, let r_i denote the number of distinct components when unit i is removed from its currently allocated component. If profile i was the only profile allocated to its mixture component, then after removing the i^{th} profile from it, the mixture component would be empty. We remove empty components and update the allocation vector $\boldsymbol{c}$ so that other profiles (excluding profile i) are correctly allocated to the remaining components (no gaps in label values).

According to [73], the current component may be allocated to one of the remaining r_i components or to one of the m new components, where m is an auxiliary variable (we set it to 3 in our simulations, as suggested by [73]). The probability of it being allocated to an existing component depends on the number of profiles currently allocated to it and parameters ($\boldsymbol{\phi}_{c_i}$) of that component. For new components, the parameters are drawn from the prior distribution. The posterior distribution, according to [73] may be expressed as

$$P(c_i = c|\boldsymbol{c}_{-i}, \boldsymbol{y}_i, \boldsymbol{\phi}) = \begin{cases} \kappa \frac{n_{-i,c}}{N-1+\alpha} L(\boldsymbol{y}_i|\boldsymbol{\phi}_c), & \text{for } 1 \leq c \leq r_i, \\ \\ \kappa \frac{\alpha/m}{N-1+\alpha} L(\boldsymbol{y}_i|\boldsymbol{\phi}_c), & \text{for } r_i \leq c \leq r_i + m. \end{cases} \tag{4.13}$$

In the above expression, κ is the normalizing constant, $n_{-i,c}$ is the number of profiles allocated to the component c excluding the current profile, and $\mathrm{L}(\boldsymbol{y}_i|\boldsymbol{\phi}_c)$ is the contribution of the likelihood by the profile i.

4.3.3 Sampling random effects parameters

From (4.9), we have

$$\begin{aligned} P(\boldsymbol{\phi}|\boldsymbol{y}) &\propto P(\boldsymbol{\phi}|\boldsymbol{c}) \times L(\Theta) \\ &\propto \prod_{i=1}^{r} P(\phi_r|\boldsymbol{c}) \times L(\Theta). \end{aligned} \tag{4.14}$$

Given that we select multivariate normal distribution as the base distribution of our Dirichlet Process, i..e, $\boldsymbol{\phi}_c \propto \mathrm{MVN}(\boldsymbol{\mu_b}, \boldsymbol{\Sigma_b})$, the posterior distribution of $\boldsymbol{\phi}_v, 1 \leq v \leq r$ is given by

$$
\begin{aligned}
P(\boldsymbol{\phi}_v|\boldsymbol{y}) \propto & P(\boldsymbol{\phi}_v) \times \prod_{u:c_u=v} \exp\left\{\frac{\boldsymbol{y}_u^t\boldsymbol{\eta}_u - \mathbf{1}^t q(\boldsymbol{\eta}_u)}{\nu} + k(\boldsymbol{y}_u,\nu)\right\} \\
\propto & \exp\left\{-\frac{1}{2}(\boldsymbol{\phi}_v - \boldsymbol{\mu}_b)^t\boldsymbol{\Sigma}_b^{-1}(\boldsymbol{\phi}_v - \boldsymbol{\mu}_b)\right\} \times \\
& \prod_{u:c_u=v} \exp\left\{\frac{\boldsymbol{y}_u^t\boldsymbol{\eta}_u - \mathbf{1}^t q(\boldsymbol{\eta_u})}{\nu} + k(\boldsymbol{y}_u,\nu)\right\} \\
\propto & \exp\left\{-\frac{1}{2}(\boldsymbol{\phi}_v - \boldsymbol{\mu}_b)^t\boldsymbol{\Sigma}_b^{-1}(\boldsymbol{\phi}_v - \boldsymbol{\mu}_b)\right\} \times \\
& \exp\left\{\frac{1}{\nu}\Big[\sum_{u:c_u=v} \boldsymbol{y}_u^t\boldsymbol{\eta}_u - \mathbf{1}^t \times q(\boldsymbol{\eta}_u)\Big] + \sum_{u:c_u=v} \mathbf{1}^t \times k(\boldsymbol{y}_u,\nu)\right\},
\end{aligned}
\tag{4.15}
$$

where the index u goes over all profiles which are allocated to the mixture component with label v.

To sample from the above posterior distribution we use the Metropolis-Hastings algorithm. As a proposal distribution, we use the multivariate normal distribution based on a second-order approximation using the Newton-Raphson procedure with one step only.

Before deriving the expressions for the mean and variance of the proposal distribution, we introduce the following notation. Let $\boldsymbol{q}'_{ij} \equiv \boldsymbol{q}'(\eta_{ij}) = \frac{\partial \boldsymbol{q}(\eta_{ij})}{\partial \eta_{ij}}$ and let $\boldsymbol{q}''_{ij} \equiv \boldsymbol{q}''(\eta_{ij}) = \frac{\partial \boldsymbol{q}'(\eta_{ij})}{\partial \eta_{ij}}$. Then let $\boldsymbol{q}'_i = (\boldsymbol{q}'_{i1}, \boldsymbol{q}'_{i2}, \ldots, \boldsymbol{q}'_{iN})^t$ and let $\boldsymbol{q}' = \Big((\boldsymbol{q}'_1)^t, (\boldsymbol{q}'_2)^t, \ldots, (\boldsymbol{q}'_N)^t\Big)^t$. Similarly, let $\boldsymbol{q}''_i = (\boldsymbol{q}''_{i1}, \boldsymbol{q}''_{i2}, \ldots, \boldsymbol{q}''_{iN})^t$ and let $\boldsymbol{q}'' = \Big((\boldsymbol{q}''_1)^t, (\boldsymbol{q}''_2)^t, \ldots, (\boldsymbol{q}''_N)^t\Big)^t$. Finally, let $\boldsymbol{q}'(\boldsymbol{\phi})$ and $\boldsymbol{q}''(\boldsymbol{\phi})$ be vectors evaluated for a particular value of $\boldsymbol{\phi}$ parameter.

Assuming that the current value of the parameter $\boldsymbol{\phi}_i$ at iteration $m-1$ is $\boldsymbol{\phi}_i^{(m-1)}$, the proposal distribution for sampling $\boldsymbol{\phi}_i$ at the next iteration will be

multivariate normal distribution MVN$(\boldsymbol{\mu}_{p_i}, \omega\boldsymbol{\Sigma}_{p_i}), i = 1, 2, \ldots, r$, where

$$\boldsymbol{\Sigma}_{p_i} = \left[\frac{1}{\nu^2}\boldsymbol{Z}_i^t\boldsymbol{q}''(\boldsymbol{\phi}_i^{(m-1)})\boldsymbol{Z}_i + \boldsymbol{\Sigma}_b^{-1}\right]^{-1} \tag{4.16}$$

and

$$\boldsymbol{\mu}_{p_i} = \boldsymbol{\phi}_i^{(m-1)} + \boldsymbol{\Sigma}_{p_i}\left[\frac{1}{\nu}\boldsymbol{Z}_i^t(\boldsymbol{y} - \boldsymbol{q}'(\boldsymbol{\phi}_i^{(m-1)})) - \boldsymbol{\Sigma}_b^{-1}\left(\boldsymbol{\phi}_i^{(m-1)} - \boldsymbol{\mu}_b\right)\right], \tag{4.17}$$

and ω is a multiplication factor that allows us to control the acceptance rate of the sampler. This is often the ratio between the number of times certain parameter has been updated and the total number of iterations; however, in our case the number of mixture components (each of which is associated with a unique random effect parameter) changes and counting the number of times a random parameter is updated would overestimate the acceptance rate. Instead, the calculate the acceptance rate as the average rate (over the number of iterations) of average number of times random effects have been updated (per iteration). The value of ω is set so that this value is between 30% and 40%.

In expressions (4.16) and (4.17), $\boldsymbol{Z}_i$ is a matrix containing subset of covariates of all profiles which were allocated to component i. This matrix is a proper subset of the original $\boldsymbol{Z}_i$ matrix specified on the full model. The size of vectors $\boldsymbol{q}'$ and $\boldsymbol{q}''$ equals the number of profiles allocated to the current mixture component.

4.3.4 Sampling fixed effects parameters

To sample fixed effects parameters $\boldsymbol{\beta}$, we follow the same steps as in the previous section. The expressions are almost the same, except that the indexes are not

restricted in Eq.(4.15) to only those profiles allocated to a particular component. Note that in the previous step, we have to sample multiple random effect parameters (one for each mixture component), while in this section we have to sample only one fixed effect parameter.

Given that the prior of $\boldsymbol{\beta}$ is multivariate normal with mean $\boldsymbol{\mu}_\beta$ and covariance matrix $\boldsymbol{\Sigma}_\beta$, the posterior distribution of $\boldsymbol{\beta}$ is

$$\begin{aligned} P(\boldsymbol{\beta}|\boldsymbol{y}) \propto & \exp\left\{-\frac{1}{2}(\boldsymbol{\beta}-\boldsymbol{\mu}_\beta)^t\boldsymbol{\Sigma}_\beta(\boldsymbol{\beta}-\boldsymbol{\mu}_\beta)\right\}\times \\ & \exp\left\{\frac{1}{\nu}\Big(\boldsymbol{y}^t\boldsymbol{\eta}-\mathbf{1}^t\times q(\boldsymbol{\eta})\Big)+\mathbf{1}^t\times k(\boldsymbol{\eta},\nu)\right\}. \end{aligned} \tag{4.18}$$

Similar to sampling random effects parameters, the proposal distribution to the above posterior distribution is multivariate normal with MVN($\boldsymbol{\mu_\gamma}, \omega\boldsymbol{\Sigma_\gamma}$), where

$$\boldsymbol{\Sigma_\gamma} = \left[\frac{1}{\nu^2}\boldsymbol{X}^t\boldsymbol{q}''(\boldsymbol{\beta}_i^{(m-1)})\boldsymbol{X}+\boldsymbol{\Sigma}_\beta^{-1}\right]^{-1} \tag{4.19}$$

and

$$\boldsymbol{\mu_\gamma} = \boldsymbol{\beta}_i^{(m-1)} + \boldsymbol{\Sigma}_\gamma\left[\frac{1}{\nu}\boldsymbol{X}^t(\boldsymbol{y}-\boldsymbol{q}'(\boldsymbol{\beta}_i^{(m-1)})) - \boldsymbol{\Sigma}_\beta^{-1}\Big(\boldsymbol{\beta}_i^{(m-1)}-\boldsymbol{\mu}_\beta\Big)\right], \tag{4.20}$$

where ω is a multiplication factor that allows us to control the acceptance rate of the sampler. Unlike in the random effects parameters case (as outlined in the previous section), ω here is calculated as the ratio of number of times a fixed effect parameter has been changed over the course of all iterations.

4.3.5 Sampling dispersion parameters

The posterior distribution of both dispersion parameter and its hyper-parameters can be shown to have gamma distribution. For the posterior of ν, we have

$$\nu^{-1} \propto \text{Gamma}\Big(\frac{\alpha_\nu + n}{2}, \frac{\beta_\nu^{-1} + (\boldsymbol{y} - \boldsymbol{\eta})^t(\boldsymbol{y} - \boldsymbol{\eta})}{2}\Big), \tag{4.21}$$

where N is the total number of observations, and $\boldsymbol{y}$ is the vector of all responses in the data set, and $\boldsymbol{\eta}$ is the vector of all linear predictors. The parameter β_ν^{-1} is updated according to the following gamma distribution

$$\beta_\nu^{-1} \propto \text{Gamma}\Big(\alpha_{\beta_\nu} + \frac{\alpha}{2}, \beta_{\beta_\nu} + \frac{\nu^{-1}}{2}\Big). \tag{4.22}$$

4.3.6 Summary of steps in GLMM-DP

The following algorithm summarizes the main steps in GLMM-DP method.

Data: Response vector $\boldsymbol{y}$, covariate matrices $\boldsymbol{X}$ and $\boldsymbol{Z}$

Result: For each MCMC sample: component alloctions $\boldsymbol{c} = \{c_1, c_n, \ldots, c_N\}$, estimates of $\boldsymbol{\beta}$ and $\boldsymbol{b}_i, i = 1, 2, \ldots, N$, estimate of concentration parameter α, acceptance rate for fixed and random parameters in Metropolis-Hastings sampler

Initialization:

Assign each profile to its own cluster, and set $\hat{\boldsymbol{\beta}}^0$ and $\hat{\boldsymbol{b}}_i^0$ to a value from their respective prior distributions;

STEP 1: Allocate profiles to clusters using Eq.(4.13);

STEP 2: Update component parameters using Eq.(4.15);

for *each mixture component* **do**

1. Derive proposal distribution using Eq.(4.16) and Eq.(4.17);
2. Sample new random effect parameter using Metropolis-Hastings algorithm;
3. Update the acceptance rate for random effects parameters;

end

STEP 3: Update fixed effect parameters as per Section 4.3.4 ;

STEP 4: Update the acceptance rate fixed effects parameters;

STEP 5: Update concentration parameter as per Eq.(4.11);

STEP 6: Update dispersion hyper-parameter as per Eq.(4.22), and its parameter as per Eq.(4.21);

Algorithm 3: Processing steps of the GLMM-DP method

The GLMM-DP method produces parameter estimates at each iteration. The number of parameters may differ from one iteration to the next, as the number of

mixture components changes. The final estimates are produced in the post-processing phase - this is described in the next section.

There are other steps performed in the algorithm that are considered too low-level and are not mentioned above, such as adding new mixture component when a profile is allocated to a new component, and/or removing an existing components when the last profile allocated to it is removed.

4.4 Label Switching

Label switching is a well-known problem in Bayesian inference in mixture models. It occurs due to the non-identifiability of mixture components.

4.4.1 Introduction

Let $\boldsymbol{y} = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_N\}$ be a data set consisting of N profiles, each profile being modeled as a mixture distribution

$$P(\boldsymbol{y}_i|\Theta) = \pi_1 P(\boldsymbol{y}_i|\boldsymbol{\theta}_1) + \cdots + \pi_K P(\boldsymbol{y}_i|\boldsymbol{\theta}_K).$$

Here $\Theta = \{\pi_1, \pi_2, \ldots, \pi_K, \boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_K\}$. The likelihood is given as

$$L(\Theta|\boldsymbol{y}) = \prod_{i=1}^{N} \Big[\pi_1 P(\boldsymbol{y}_i|\boldsymbol{\theta}_1) + \cdots + \pi_K P(\boldsymbol{y}_i|\boldsymbol{\theta}_K)\Big].$$

Let $\boldsymbol{V}_K$ be the set of all permutations on $(1, 2, \ldots, K)$. Then for $\nu_i \in \boldsymbol{V}_K$, let $\nu(\Theta) = (\pi_{\nu(1)}, \pi_{\nu(2)}, \ldots, \pi_{\nu(K)}, \boldsymbol{\theta}_{\nu(1)}, \boldsymbol{\theta}_{\nu(2)}, \ldots, \boldsymbol{\theta}_{\nu(K)})$. The label switching problem

occurs because the likelihood

$$L(\nu(\Theta)|\boldsymbol{y}) = \prod_{i=1}^{N} \left[\pi_{\nu(1)} P(\boldsymbol{y}_i|\boldsymbol{\theta}_{\nu(1)}) + \cdots + \pi_{\nu(K)} P(\boldsymbol{y}_i|\boldsymbol{\theta}_{\nu(K)})\right]$$

is the same for any permutation $\nu \in \boldsymbol{V}_K$. This is due to the fact that the likelihood function is invariant to the order of components. However, the order of the components may change during the sampling process, but the parameter estimation using the MCMC relies on the order of components being consistent across all iterations. For example, assuming that we have K components in our model, to estimate the parameter of the k^{th} component, we would average the parameters of all k^{th} components, across all iterations. However, what was the k^{th} component in one iteration may become the m^{th} component in the next iterations, $k \neq m$. Therefore, averaging parameter estimates across all iterations may not produce valid results, because the posterior surface may consist of up to $K!$ different modes, each corresponding to a different permutation $\nu_k \in \boldsymbol{V}_K$, unless this symmetry is broken by the parameter prior.

The label switching problem may not be an issue if we are primarily interested in estimating population or unit level parameters. Population level parameters do not depend on mixture components and can be estimated by averaging their estimates over all iterations, while unit level parameters may depend on mixture component but can be estimated easily since their value can be uniquely identified in each iteration. However, estimating component level parameters requires that the label switching issue be resolved.

Label switching problem has been extensively studied in finite mixture models, and various solutions have been proposed. These solutions could be divided into three different types [106]. The first type imposes identifiability constraints on the prior and that way breaks the symmetry in the posterior [7]. For example, given $\Theta = (\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_K)$, an identifiability constraint may be just

$$\boldsymbol{\theta}_1 < \boldsymbol{\theta}_2 < \cdots < \boldsymbol{\theta}_K,$$

where inequality operator is appropriately defined on vectors.
This leads to the correct marginals (as $N \to \infty$), but for finite N, it may over-estimate difference between parameters [107]. Another type of label switching solutions, called the deterministic relabeling algorithms, treat two permutations as one if the characteristic of interest has similar values under the two permutations. Stephens [108] uses the matrix allocation probabilities of the observations as the characteristic of interest, while [109] uses closeness of allocation vectors $\boldsymbol{c}_i$. Finally, the last class of label switching solutions places probabilities over permutations of mixture components.

The above solutions apply to mixture models with finite number of components. In infinite mixture models, Yan He [12] proposes a solution based on hierarchical clustering of longitudinal profiles. Our implementation is based on a similar approach proposed in [110]. We describe it in the next section.

4.4.2 Finding a Reference Clustering

In this section, we describe the process of finding the clustering of profiles that is best representative of the posterior sampling.

Each MCMC iteration produces a cluster assignment vector $\boldsymbol{c} = \{c_1, c_2, \ldots, c_N\}$, where $c_i = k$ if the i^{th} profile is assigned to the mixture component k. This vector induces a partitioning of profiles. We denote the partitioning of profiles at iteration g by $\boldsymbol{P}_g = \{\boldsymbol{p}_1, \boldsymbol{p}_2, \cdots, \boldsymbol{p}_K\}$, where $\boldsymbol{p}_k = \{j : c_j = k\}$. Here $\boldsymbol{p}_k$ contains indices of all profiles which were allocated to cluster k.

Not only can the component assignment change from one iteration to the next, but the number of mixture components may also change. So we cannot compare cluster assignments across iterations. The best we can do is to build the similarity matrix between all profiles [111]. This is a matrix $\boldsymbol{M}$ of size $N \times N$, where $ij^{(th)}$ element specifies the percent of iterations in which profiles i and j were allocated to the same component, i.e.,

$$m_{ij} = \frac{\text{\# of samples after burn-in for which } c_i = c_j}{\text{\# of samples after burn-in}}$$

Then based on the similarity matrix, we use the partitioning around medoids (PAM) method [4] to cluster profiles. The PAM method requires a number of clusters to be provided in advance, and it requires a dissimilarity matrix, which in our cases is $1 - \boldsymbol{M}$. We may run the method N times, and for each N, we cluster profiles and choose the clustering with the highest average silhouette width. A silhouette width of a profile i is defined as

$$s(i) = \frac{\min_{\boldsymbol{C}} d(i, C) - d(i)}{\max\big[d(i), \min_{\boldsymbol{C}} d(i, C)\big]},$$

where the index $\boldsymbol{C}$ goes over all profile clusters, $d(i, C)$ is the average dissimilarity between profile i and all other profiles in cluster C, $d(i)$ is the average dissimilarity

between profile i and all other profiles assigned to the same cluster as profile i.

4.4.3 Estimating Component Parameters

Let $\boldsymbol{P}_r$ be the reference partition produced by the algorithm described in the previous section. The reference partition $\boldsymbol{P}_r$ may be different for all partitions $\boldsymbol{P}_i, 1 \leq i \leq M$, where M is the total number of MCMC iterations. Even if $\boldsymbol{P}_r$ matches one of the existing partitions, chances are that there may be too few such partitions in order to make valid estimates based only on them. This is especially true for large datasets or data sets that come from subpopulations that are not well separated, a pattern that many real-world data sets exhibit. The method described in this section is based on [112].

The GLMM-DP method assigns a random effect $\boldsymbol{b}_i$ to profile i, which in turn is assigned to a particular cluster that is a part of the partition. Let the reference partition $\boldsymbol{P}_r$ consist of k clusters, i.e., $\boldsymbol{P}_r = (\boldsymbol{p}_{r_1}, \boldsymbol{p}_{r_2}, \ldots, \boldsymbol{p}_{r_k})$. Then parameter estimate for cluster j at iteration i, $\boldsymbol{b}_j^{(i)}, 1 \leq j \leq k, 1 \leq i \leq M$ is the average of random effects parameters $\boldsymbol{b}_u$ of all profiles which were allocated to a cluster for which $c_u \in \boldsymbol{p}_{r_j}$, i.e.

$$\hat{\boldsymbol{b}}_j^{(i)} = \frac{1}{n_{r_j}} \sum_{c_u \in \boldsymbol{p}_{r_j}} \boldsymbol{b}_u,$$

where n_{r_j} is the number of profiles in j^{th} partition of the reference partition $\boldsymbol{P}_r$.

Basically, the estimate of $\boldsymbol{b}_i$ at the j^{th} cluster is the average over all $\boldsymbol{b}_j$ that were assigned in the same cluster in the reference partition but may have been allocated to different clusters in the partition of the current iterations.

Based on the above estimates, we may estimate the density of component parameters. Flat density functions would indicate inconsistent profile clustering across different iterations, and would require that the label switching solution be re-examined.

The following algorithm summarizes the above steps:

Data: MCMC output from the processing phase

Result: Estimates of $\boldsymbol{\beta}$ and $\boldsymbol{b}_i, i = 1, 2, \ldots, N$, the optimnal clustering of profiles ($\boldsymbol{P}$)

STEP 1: Derive the similarity matrix as described in Section 4.4.2;

STEP 2: Derive the final clustering of profiles ($\boldsymbol{P}$) as described in Section 4.4.2;

$K \leftarrow$ number of clusters in $\boldsymbol{P}$

STEP 3: Derive estimates of $\boldsymbol{b}_i, i = 1, 2, \ldots, K$ as described in Section 4.4.3;

STEP 4: Derive fixed parameter estimates by taking mean of $\hat{\boldsymbol{\beta}}^{(m)}$ for each MCMC sample m;

Algorithm 4: Post-processing steps in the GLMM-DP method

4.5 Summary

In this chapter, we have introduced the GLMM-DP method which facilitates clustering of longitudinal profiles in generalized linear mixed models and estimating model parameters at the same time. We have fully defined the model, including the (type of) prior of its parameters and have derived posterior distributions of all model

parameters - the actual values of prior parameters are data specific and would need to be specified by a data analyst. We have also described a label switching solution in order to be able to draw inferences at the component level. Similar methods have been proposed recently, but only in longitudinal studies in which the response variable is continuous and the underlying model is a Dirichlet Process mixture of linear mixed models. The GLMM-DP method is the first method of its kind that extends the scope of the response variable to any type with a distribution that is a member of the exponential family.

There are many ways to cluster longitudinal data. For example, one may cluster profiles by grouping response values, separately or together with the values of explanatory values. One could cluster profiles based on the similarity of their patterns of change over time. The GLMM-DP method provides a mechanism to estimate model parameters and cluster profiles based on similarity of model parameters, without imposing a number of clusters in advance.

It is important to put this method in the right context. Clustering data in which observations are not independent is a challenging task. Clustering profiles is even more challenging because profiles may evolve over time in different ways. Attempting to solve this problem while relaxing constraints such as the number of clusters or distribution of parameters, makes the clustering even more challenging. It is easy to come up with a scenario in which this method would work very well (well-separated clusters of model parameters), and other scenarios in which it may not perform as well.

Therefore, the user would always need to verify the result of clustering, and

may have to override the results produced by this method. The user may use the GLMM-DP method to identify homogeneous groups in the data, but sometimes the method may produce grouping that the user is either not interested in, or that don't quite make sense. For example, two or more clusters may represent a subpopulation that does not differ in a significant way, or clusters may contain very few profiles that the user may verify to be outliers (or perhaps of no interest). In such cases, the user could re-run the method, with different values of the concentration parameter of the Dirichlet Process prior (forcing the number of clusters to be smaller/larger (but still unknown)), the user may remove profiles before re-running the method, or the user may discard some clusters. Whatever the case, the user may be more likely to override results produced by the GLMM-DP method than with other methods. The main reason for this is that the GLMM-DP method tries to solve a very challenging problem, the general solution to which may always require at least some human input.

The proposed method is not meant to replace existing methods. Rather, it may be considered as an additional tool in data analyst's arsenal that may be used in combination with other tools and methods to get insight into the data that one may not be able to obtain using any other single method out there.

Chapter 5

Simulation Study

In this chapter, we evaluate the performance of the proposed GLMM-DP method using simulated data. We perform simulations on models with two different types of responses: one with continuous and another one with count response.

We organize this chapter as follows. First, we describe the process of simulating data sets. Then, we briefly walk through the process of verifying convergence of posterior distributions on a single model. Finally, we generate 100 replicates of data sets for each type of the model and under different values of input parameters, and summarize the performance characteristics of the GLMM-DP method.

5.1 Simulating Data Sets

We simulate a data set with $N = 100$ units, with n observations taken on each unit (n can be one of the following three values: 5, 10 or 20). We assume that all observations are taken at a fixed set of time points $t = 1, 2, \ldots, n$ (i.e., we have a balanced design). The distribution of the response variable Y is either normal or Poisson distribution, making our link function either the identity function (in case of normal) or a log function (in case of Poisson distribution). The linear predictor consists of three fixed

effects and one random effect parameter:

$$\eta_{ij} = X_{ij1}\beta_1 + X_{ij2}\beta_2 + X_{ij3}\beta_3 + b_i. \tag{5.1}$$

In the above equation, X_{ijk} is the value of the k^{th} predictor observed at time j on unit i. We simulate the values of predictors so that predictors of successive observations on the same individual are highly correlated. Values of the first and second predictors at time j are linear combinations of their values at a previous time ($t = j - 1$) and some random component. More specifically, we set X_{ij1} to $X_{i(j-1)1} * 0.78 + N(0, 0.1)$, and similarly, we set X_{ij2} to $X_{i(j-1)2} * (-0.78) + N(0, 0.1)$. Values of the first and second predictors at time zero are drawn randomly from the normal distribution with means 0.1 and 0.9, respectively, and a standard error of 0.5. Finally, the last predictor X_{ij3} represents the time at which the observations were taken, $X_{ij3} = j$. We standardize X_{ij3} so that its mean is 0 and standard deviation is 1.

For simplicity, we take the random effect b_i to be an intercept only. The only constraints we have to be aware of, for identifiability purposes, is to ensure that the intersection of column space of fixed effect covariate matrix $\boldsymbol{X}$ and random effect covariate matrix $\boldsymbol{Z}$ is non-empty. This is guaranteed with the above model by including an intercept in $\boldsymbol{Z}$ and not including it in $\boldsymbol{X}$. The random effects matrix $\boldsymbol{Z}$ has only one column - with all its entries set to one.

We simulate a dataset that consists of two clusters. Each cluster is of the same size. The first half of the units are assigned to the first cluster and the second half of units are assigned to the second cluster. The random effects parameters are generated from the normal distribution, with the mean in the first cluster being

equal to one of the following values: $-0.5, 0.5, 0.75, 1.15, 1.65, 2.2$, and a standard deviation of 0.1. The mean of the random effects in the second cluster is set to 1.15, and its values are generated from the normal distribution with the same standard deviation. Therefore, each unit is initialized with different (but very similar) random effect parameter, while the GLMM-DP method assumes that units in the same cluster have the same random effect parameter. Hence, we refer to the true value of a parameter through its mean (such as μ_b) and its estimate as $\hat{b}$.

The fixed effect parameters do not vary in our simulation study and are fixed at $\boldsymbol{\beta} = (0.8, 0.6, 0.3)$.

Given the values of the linear predictors generated according to the above scheme, we simulate values of the responses either from the normal distribution with mean $\boldsymbol{X\beta} + \boldsymbol{b}$ and a standard deviation of 0.1, or from the Poisson distribution with the mean $\exp(\boldsymbol{X\beta} + \boldsymbol{b})$.

5.2 Simulation on a Single Data Set

In the previous section, we provided detailed description on how we simulate a data set. Before creating multiple replicates of data sets under different scenarios, which we do in the next section, in this section we walk through the process of verifying the convergence of posterior distributions of model parameters. Once we establish that the posterior distributions converge to stationary distributions, we derive the point estimates of model parameters and their respective credible intervals. We approximate the posterior distribution of model parameters using 20,000 Markov chain Monte Carlo iterations.

To test if the posterior distribution converges to the stationary distribution, we use the Geweke's convergence diagnostic [66], as implemented in coda package [113]. This diagnostic compares the mean of the parameter in the first part of the chain (10% of iterations, by default) with the mean of the same parameter using the last part of the chain (50% of iterations, by default). If the distribution converges to the stationary distribution, the two means should be very close, and the absolute difference between them should be asymptotically normal. That allows us to use the standard Z-score statistics to test the hypothesis that the two means are indeed the same. We conduct the test at 5% level of significance.

Table 5.1: Geweke's statistic for fixed and random effect parameters for the model with count responses

Parameter	Geweke's statistic
β_1	0.6599
β_2	-0.3405
β_3	-0.4670
b_1	0.3721
b_2	-0.9883

Table 5.1 shows values of the the Geweke's statistic when the first part of the chain is set to 0.2 or 20% of the total number of iterations. We see that all values are well within $[-1.96, 1.96]$. Therefore, the Geweke's diagnostic indicates that using the first 20% of the chain as a burn-in period is reasonable.

Figure 5.1 shows that the target state space of the fixed effect parameters is explored well (we have good mixing) and that the densities of the parameters are

fairly peaked, though the density of β_3 does not have a bell shape as nice as those of β_1 and β_2.

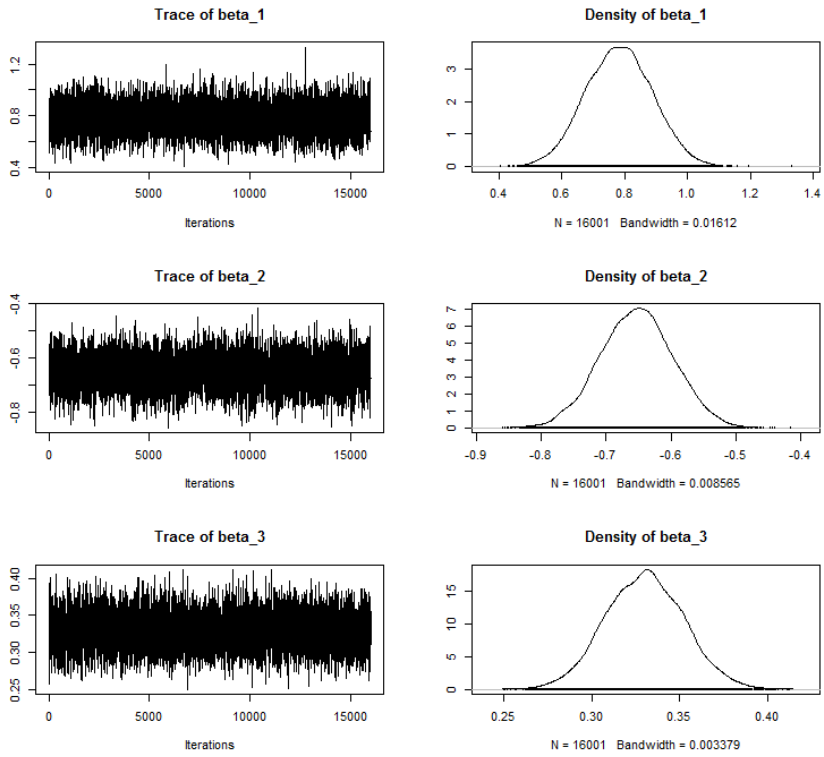

Figure 5.1: Trace and density plot of fixed effect parameters in the model with count response

Figure 5.2 shows that the target state space of the random effect parameters is explored well and that the distribution of the parameters is not too flat. This indicates that out label switching solution is performing well.

Figures 5.3 and 5.4 show the autocorrelation between successive samples of fixed and random effects parameters, respectively. We see that in both cases the autocorrelation

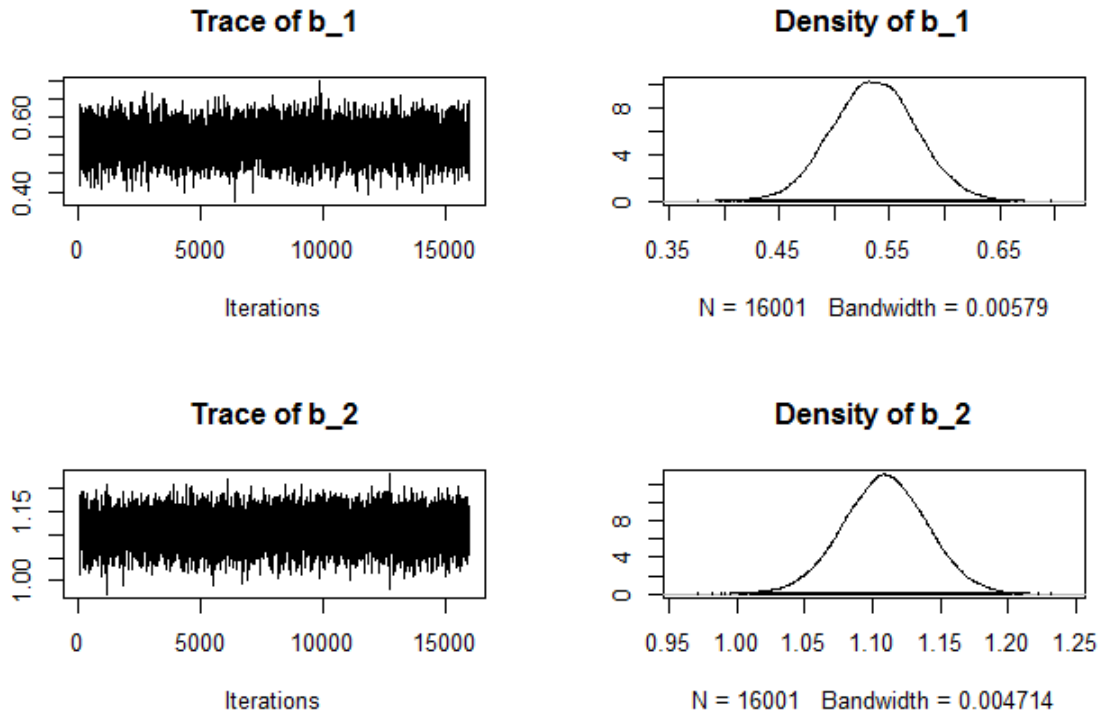

Figure 5.2: Trace and density plot of the fixed effect parameters

drops fairly quickly, indicating that the trace plots in Figure 5.1 (for fixed effects parameters) and Figure 5.2 (for random effects parameters) are fairly good diagnostics of convergence. Strictly speaking, the plot in Figure 5.4 is not an autocorrelation plot between successive random effects parameters, as they are generated during the MCMC procedure. Instead, it is the autocorrelation plot of random effects parameters as produced by the label switching procedure. It is impossible to trace a random effect parameter through the MCMC procedure, and the autocorrelation between these "calculated" random effects seems to be the best we can have. We continue referring to them as autocorrelation plots of random effects parameters in the rest of the thesis, but it is important to keep in mind that they are not the true autocorrelation plots of random effects parameters as produced by the MCMC.

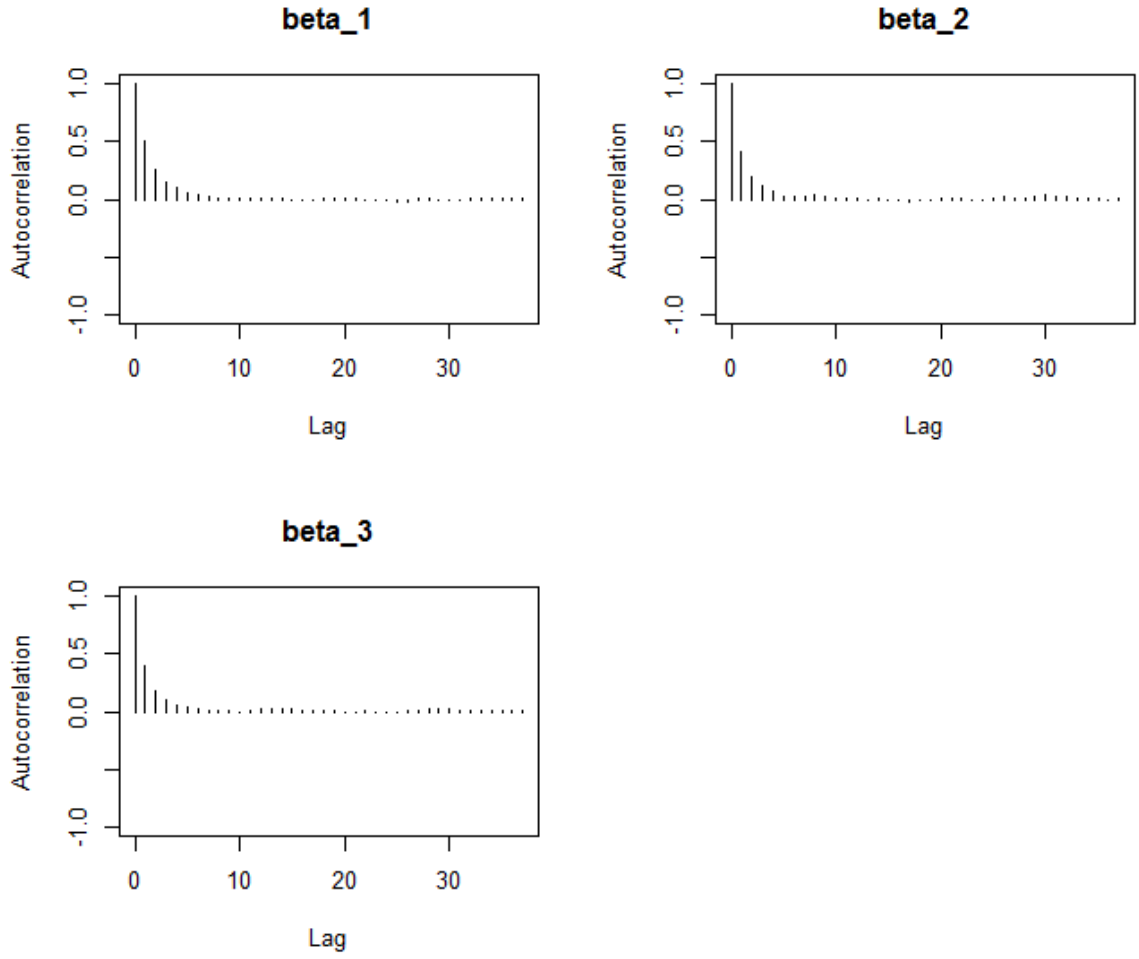

Figure 5.3: Autocorrelation between successive fixed effects parameters of the model with count response

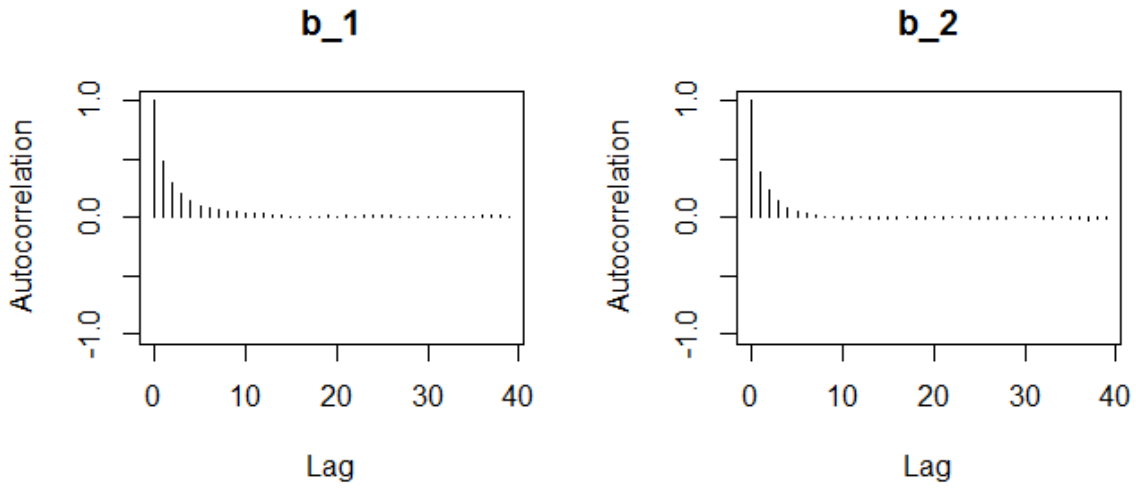

Figure 5.4: Autocorrelation between successive random effects parameters of the model with count response

The above diagnostic results indicate that the posterior distributions of our model parameters converge to the stationary distributions, and that we may proceed with parameter estimates. We get the point estimates of the fixed effects parameters as $\hat{\boldsymbol{\beta}} = (\hat{\beta}_1, \hat{\beta}_2, \hat{\beta}_3) = (0.6710, -0.5772, 0.2887)$, with the highest posterior density interval $(0.4820, 0.8461)$ for β_1, $(-0.4800, -0.6703)$ for β_2, and $(0.2474, 0.3318)$ for β_3. The true values of the fixed effects parameters are $\boldsymbol{\beta} = (0.8, -0.6, 0.3)$

We get the point estimate of the random effect for the first cluster as 0.4994, with its 95% highest posterior density (HPD) interval being $(0.4215, 0.5720)$. The point estimate of the random effect for the second cluster is 1.1733 and its 95% HPD interval is $(1.1164, 1.1.2276)$. The true value of the random effects for the first cluster is 0.5 and for the second cluster 1.15. These results show that the point estimates of both fixed and random effect parameters are within the 95% HPD interval.

The above shows that to approximate the posterior distributions of the model

parameters, it suffices to generate 20,000 MCMC samples. After discarding the first 4,000 samples, considered to be part of the burn-in period, we use the remaining 16,000 samples for parameter estimation. We use these numbers in the following section, in which we generate data sets with different model parameters and perform their estimations at the end of each simulation, without explicitly checking convergence of their posterior distributions. We do so because we have shown in this section that 20,000 samples approximate posterior distributions fairly well.

5.3 Simulation Results

We repeat the above experiment 100 times under three different conditions. In the first condition, we vary the mean of the random effect parameter of one of the two clusters (with the mean of the random effect parameter of the other cluster being the same in all settings). We choose 6 different values for this mean: -0.5, 0.5, 0.75, 1.15, 1.65 and 2.2. In the second condition, we vary the number of observations per individual. We choose three cases: 5, 10 and 20 observations per individual. Finally, in the third condition, we change the type of the response variable. We test the model with both continuous and count response. The primary focus of the thesis is the count response and the continuous response is used mostly as a reference case, especially given that the clusters in continuous response are well separated when the difference between the means of their random effects parameters is greater than 0.5, while the clusters in count response are not as well separated.

Similar to other studies [114], we evaluate the performance of the GLMM-DP method using several metrics. This includes the number of times in which the correct number of clusters was recovered, the percent of profiles that were classified

correctly, and the mean squared error (MSE) for random effects parameters. For each run, we evaluate these metrics and provide their point estimates with their standard errors. For MSE of random effects, we take the square root of the difference between the true and the posterior mean of random effect parameter. Note that in frequentist statistics this would make no sense (since random effects are predicted not estimated), however, in Bayesian statistics, the two parameter types are treated the same way.

5.3.1 Simulation Results with Continuous Response

Table 5.2 shows the number of times (out of 100) in which two clusters were recovered from the data. It is clear from the table that the GLMM-DP method successfully recovers the true number of mixture components (or clusters) in almost all cases, except in the case when both clusters are initialized with the same random effect parameter ($\mu_{b_1} = \mu_{b_2}$). We observe that our proposed method produces better results for larger sample sizes, as expected.

Table 5.2: Data with continuous response: number of times in which two clusters were recovered from the data ($\mu_{b_1} = 1.15$)

μ_{b_2}	$n_i = 5$	$n_i = 10$	$n_i = 20$
-0.5	99	100	100
0.5	100	100	100
0.75	98	100	100
1.15	81	82	95
1.65	100	100	100
2.2	100	100	100

The current version of the GLMM-DP method can infer no less than two clusters in the data. This is due to the label switching solution, as outlined in Section 4.4. More specifically, this is due to the fact that the PAM method [4] produces at least two clusters. Even with this fact, we see in Table 5.2, that there are many runs in which more than 2 clusters were inferred (although better results were obtained with larger number of observations per individual). This is also due to the PAM method: the similarity matrix is very dense and the method identifies too many clusters. However, a visual inspection of the results can detect this anomaly, since the values of the estimated random effects parameters in each cluster are very similar. For example, one of the runs (with $n_i = 20$) produces three clusters with the following estimates of random effects parameters: $1.15015322, 1.150151145, 1.15015109$.

Table 5.3 shows the percent of correctly classified profiles for each value of the random effects parameter (that varies in the simulation), and for each number of observations per individual. Note that these results are calculated only for those cases in which the correct number of mixture components were estimated (as shown in the previous section). It is clear from the table that the model fully recovers profiles in all conditions except when both clusters are given the same mean of the random effects. Again, this is due to the label switching solution, as described in the previous section.

Table 5.4 shows the mean squared errors (MSEs) for random effects parameters when the response variable is continuous. The MSE is calculated as the square root of the difference between the true value and the posterior mean of random effects parameters. In Bayesian statistics, fixed and random effects parameters are treated the same way, while in frequentist statistics fixed effects are estimated and the

Table 5.3: Data with continuous response: classification accuracy of profiles

μ_{b_2}	$n_i = 5$	$n_i = 10$	$n_i = 20$
-0.5	100.00	100.00	100.00
0.5	100.00	100.00	100.00
0.75	100.00	100.00	100.00
1.15	59.93	62.41	63.31
1.65	100.00	100.00	100.00
2.2	100.00	100.00	100.00

random effects are predicted. We observe that the smallest overall MSE is obtained

Table 5.4: Data with continuous response: MSE of random effects parameters

μ_{b_2}	$n_i = 5$	$n_i = 10$	$n_i = 20$
-0.5	0.0056	0.0042	0.0031
0.5	0.0058	0.0042	0.0029
0.75	0.0222	0.0040	0.0032
1.15	0.0036	0.0029	0.0022
1.65	0.0066	0.0047	0.0031
2.2	0.0054	0.0038	0.0031

when the data set is simulated with two clusters and each cluster is given the same random effects mean. We also observe that the MSE decreases as more observations are recoded on each individual.

Table 5.5 shows the average value of point estimates of fixed effect and random effect parameters over 100 runs. The first column shows the true value of one random effect parameter (this is constant for all runs). The second column shows the

Table 5.5: Data with continuous response: estimates of fixed and random effects parameters. (simulation standard errors of random effects parameters are shown in parentheses)

μ_{b_1}	μ_{b_2}	$\hat{\beta}_1$	$\hat{\beta}_2$	$\hat{\beta}_3$	$\hat{b}_1$	$\hat{b}_2$
			$n_i = 5$			
1.15	-0.5	0.7992	-0.5998	0.2994	1.1498(0.0061)	-0.4996(0.0065)
1.15	0.5	0.8025	-0.6003	0.3003	1.1483(0.0059)	0.4999(0.0069)
1.15	0.75	0.8015	-0.6002	0.2999	1.1283(0.0095)	0.7716(0.0083)
1.15	1.15	0.7985	-0.5994	0.2995	1.1495(0.0043)	1.1495(0.0043)
1.15	1.65	0.7997	-0.5998	0.2997	1.1520(0.0068)	1.6474(0.0074)
1.15	2.2	0.7983	-0.6009	0.3005	1.1494(0.0061)	2.1995(0.0064)
			$n_i = 10$			
1.15	-0.5	0.8000	-0.6016	0.3006	1.150(0.0048)	-0.5002(0.0048)
1.15	0.5	0.8004	-0.5998	0.3002	1.1513(0.0045)	0.5000(0.0049)
1.15	0.75	0.7995	-0.599	0.3002	1.1496(0.0038)	0.7507(0.0035)
1.15	1.15	0.8006	-0.5995	0.2998	1.1499(0.0027)	1.1499(0.0027)
1.15	1.65	0.8000	-0.5997	0.2998	1.1501(0.0052)	1.6493(0.0051)
1.15	2.2	0.8003	-0.5999	0.3002	1.1507(0.0044)	2.2002(0.0041)
			$n_i = 20$			
1.15	-0.5	0.8026	-0.5994	0.3001	1.1495(0.0033)	-0.5005(0.0038)
1.15	0.5	0.8004	-0.5998	0.3000	1.1510(0.0032)	0.4998(0.0033)
1.15	0.75	0.7995	-0.599	0.3002	1.1496(0.0038)	0.7507(0.0035)
1.15	1.15	0.8006	-0.5995	0.2998	1.1499(0.0027)	1.1499(0.0027)
1.15	1.65	0.7974	-0.5999	0.3003	1.1501(0.0033)	1.6498(0.0036)
1.15	2.2	0.8003	-0.5999	0.3002	1.1496(0.0035)	2.1996(0.0035)

second random effect parameter, varying from -0.5 to 2.2. The next three columns show the estimates of fixed effect parameters. The true values of these parameters are set as follows: $\beta_1 = 0.8$, $\beta_2 = -0.6$, $\beta_3 = 0.3$. The last two columns show the estimates of the first two columns, with simulation standard errors shown in parentheses.

Table 5.5 shows that estimates of both fixed and random effects parameters are very close to their true values. We provide standard deviations only for estimates of random effects parameters since, in general, one would expect their precision to be smaller. The reason for this is that fixed effects parameters are estimated using all available data, while random effects parameters are estimated using only data (profiles) that is allocated to a particular cluster. However, the results seem to indicate that the precision of random effects parameters is indeed very high.

5.3.2 Simulation Results with Count Response

In this section, we present the results of the GLMM-DP method when applied on longitudinal data in which the response variable is of count type. We compare the results in this section with those obtained in the previous section.

The GLMM-DP method produces exactly the same results for the true number of clusters when the response variable is count as it does when the response variable is continuous, as shown in Table 5.6.

Table 5.7 shows the percent of correctly classified profiles when the response variable represents counts. The results are obtained by considering only those runs in which two clusters were recovered (even in the case where both clusters have

Table 5.6: Data with count response: number of times in which two clusters were recovered from the data

μ_{b_2}	$n_i = 5$	$n_i = 10$	$n_i = 20$
-0.5	99	100	100
0.5	100	100	100
0.75	98	100	100
1.15	81	82	95
1.65	100	100	100
2.2	100	100	100

the same random effects parameter and the method should have recovered a single cluster). First, we see that the method cannot cluster profiles as successfully as it did

Table 5.7: Data with count response: classification accuracy of profiles

μ_{b_2}	$n_i = 5$	$n_i = 10$	$n_i = 20$
-0.5	98.87	99.93	100.00
0.5	87.93	93.50	98.93
0.75	80.28	84.57	92.80
1.15	74.09	76.13	77.67
1.65	89.13	94.80	99.20
2.2	99.90	100.00	100.00

in the case when the response variable was continuous. This is expected. However, the results are still very good. In many cases, the method manages to cluster profiles correctly with more than 90% accuracy rate. Second, we observe that the lowest accuracy rate is obtained when the two clusters are initialized with the same random effects parameter. This is so for the same reason as explained in the previous section. Finally, we see that the overall accuracy rate improves as the number of observations

per individual increases, and as two two clusters get more separated.

Table 5.8 shows the mean squared error (MSE) for each condition (specific value of random effects parameter and number of observations per individual).

Table 5.8: Data with count response: MSE of random effects parameters

μ_{b_2}	$n_i = 5$	$n_i = 10$	$n_i = 20$
-0.5	0.0622	0.038	0.0257
0.5	0.0941	0.0466	0.0210
0.75	0.1228	0.0693	0.0363
1.15	0.0392	0.0302	0.0140
1.65	0.0721	0.0341	0.0167
2.2	0.0273	0.0156	0.0126

We observe that the MSE decreases as the number of observations per individual increases (as expected). Also, the MSE increases as the distance between centers of the two clusters decreases. For example, the largest MSE is obtained when the random effect parameter is 0.75. We also observe, that the MSE is quite bigger than the corresponding MSE with continuous response (by approximately 10 times).

Table 5.9 shows the point estimates of both fixed and random effects parameters. We see that the number of observations taken on an individual does not affect the estimates of fixed effect parameters as it affects random effect parameters. We also see that the estimates of random effects parameters are better (in terms of precision and bias) when $n_i = 10$ than those when $n_i = 5$, and also are better when $n_i = 20$ than those when $n_i = 10$.

Table 5.9: Data with count response: estimates of fixed and random effects parameters. (simulation standard errors of random effects parameters are shown in parentheses)

μ_{b_1}	μ_{b_2}	$\hat{\beta}_1$	$\hat{\beta}_2$	$\hat{\beta}_3$	$\hat{b}_1$	$\hat{b}_2$
			$n_i = 5$			
1.15	-0.5	0.8047	-0.6031	0.3029	1.1279(0.0439)	-0.5278(0.0834)
1.15	0.5	0.8031	-0.5972	0.3016	0.9878(0.1767)	0.6321(0.1979)
1.15	0.75	0.8035	-0.5892	0.3020	0.9851(0.0835)	0.9215(0.1577)
1.15	1.15	0.8015	-0.6058	0.3021	1.1457(0.0315)	1.1241(0.0759)
1.15	1.65	0.8041	-0.6015	0.3013	1.2570(0.1259)	1.5384(0.1459)
1.15	2.2	0.8080	-0.5987	0.3019	1.1427(0.0369)	2.1923(0.0230)
			$n_i = 10$			
1.15	-0.5	0.806	-0.5984	0.3022	1.1506(0.0284)	-0.5143(0.0529)
1.15	0.5	0.8022	-0.6020	0.2996	1.1007(0.1049)	0.5493(0.1114)
1.15	0.75	0.7910	-0.6006	0.3002	1.0494(0.1015)	0.8418(0.1166)
1.15	1.15	0.8088	-0.6022	0.2972	1.1493(0.0213)	1.1307(0.0693)
1.15	1.65	0.8068	-0.6006	0.3000	1.1917(0.1037)	1.6006(0.1038)
1.15	2.2	0.7966	-0.5992	0.3005	1.1442(0.0209)	2.1988(0.0140)
			$n_i = 20$			
1.15	-0.5	0.8027	-0.6009	0.3029	1.1512(0.0212)	-0.5064(0.0351)
1.15	0.5	0.7947	-0.6012	0.3018	1.1354(0.0605)	0.5095(0.0658)
1.15	0.75	0.807	-0.6042	0.3010	1.0959(0.0923)	0.8048(0.0993)
1.15	1.15	0.805	-0.6065	0.3008	1.1467(0.0124)	1.1488(0.0231)
1.15	1.65	0.8026	-0.6011	0.2991	1.1537(0.0204)	1.6432(0.0145)
1.15	2.2	0.7966	-0.5992	0.3005	1.1490(0.0177)	2.1994(0.0110)

Table 5.9 also shows that the standard errors of random effects parameters are the smallest in the case when the clusters perfectly overlap. This is in line with the observations on accuracy and the MSE discussed in the previous section.

The estimates of both fixed and random effects parameters are very similar to those obtained in normal cases (previous section). The precision of estimates of random effects parameters are significantly better in normal case (by an order of magnitude of 10) than they are in the Poisson case. This could be due to the sampling method (being approximate in the Poisson case and exact in the normal case).

The results clearly indicate that the GLMM-DP method manages to recover two clusters in the data (in all but the case when two clusters perfectly overlap), and assigns profiles to the correct clusters with high probability, while simultaneously estimating both fixed and random effects parameters.

5.4 Summary

In this chapter, we have evaluated the performance of the GLMM-DP method using simulated data. We have performed the simulation with different conditions. This includes two different types of response variables (continuous and count), different numbers of observations recorded on each individual, and different (mean) values of random effects parameters, which control how separated two clusters are. We have evaluated the GLMM-DP method using several evaluation criteria: number of clusters correctly recovered from the data, classification accuracy of profiles, the

mean squared error of the random effects parameter (that varied in simulations) and using point estimates of all parameters.

In all scenarios (except when the two clusters perfectly overlap), and in terms of all evaluation criteria mentioned above, the GLMM-DP method performs very well. As expected, the results are better on data with continuous response than on data with count response. This is because sampling parameters using Metropolis-Hastings algorithm, which uses an approximation of the posterior distribution as the proposal distribution, can never be as good as directly sampling parameters from the posterior distribution when it has a known form, as is the case when the response is continuous.

The GLMM-DP method is not able to recognize when there is only one cluster in the data. This is a limitation of the PAM method, which is used in label switching solution and which can produce no less than two clusters. This problem could be further investigated. However, given that the estimates of cluster parameters are very similar, as shown in the case of a continuous response, the results could be visually inspected and easily overridden by a data analyst.

Chapter 6

Analysis of Public Health Data

In this chapter, we apply the GLMM-DP method on a dataset consisting of asthma patients, as collected in the Canadian Community Health Survey in 2013 [115]. The dataset is grouped by health regions. As expected, the GLMM-DP method produces results that are similar to those obtained by a frequentist method [116] based on the maximum likelihood estimation technique. In addition to being able to estimate the model parameters, the GLMM-DP method also identifies clusters of health regions that are more homogeneous in nature and on which other frequentist methods produce results that are different from those results obtained on the full dataset.

6.1 Canadian Community Health Survey

The Canadian Community Health Survey (CCHS) [115] is a large cross-sectional survey that is jointly designed by the Canadian Institute for Health Information, Statistics Canada and Health Canada, and is run by Statistics Canada. The main objective of the survey is to collect health-related information on Canadians and provide data that may be used for research, health surveillance (such as disease

monitoring), and utilization of health care programs and services.

The CCHS target population is the Canadian population who are 12 years of age or older. The target population excludes the institutionalized individuals, persons living on Aboriginal settlements or reserves, full-time members of the Canadian Forces, and two regions in Quebec (Nunavik and Terres-Cries-de-la-Baie-James), and therefore, the results of this or any other method, cannot be used to make inferences on this excluded population.

The objective of the survey is to provide data at a community level, where a community or health region is a geographic area that is defined by provincial ministries of health. A health region most often consists of several neighboring census subdivisions that are the responsibility of the same regional health authority. A census subdivision is an "area that is a municipality or an area that is deemed to be equivalent to a municipality for statistical reporting purposes" [117]. In 2015, Canada was partitioned into 112 different health regions. However, the number of health regions, as well as their boundaries, may change from one year to another, as may be required per provincial jurisdictions.

The CCHS survey uses a two-stage sampling design, with the first stage being at the province level, and the second stage being at the health region level. Generally, models for complex survey data are considered with survey weights; however, for the purpose of this thesis to illustrate the application of the GLMM-DP method, we do not consider survey weights.

In this chapter we consider a subset of CCHS dataset obtained from 2013.

The dataset includes only asthma patients. Our objective is to identify subgroups of asthma patients who exhibit similar behaviors with respect to the number of times they visit a doctor's office, and to explain what predictors affect the number of visits and how.

6.2 Model Description

We analyze the aforementioned survey data using the proposed GLMM-DP method, as described in previous chapters. The results of the GLMM-DP method are compared with those obtained by ordinary maximum likelihood (ML) method [116].

We consider the number of consultation with a medical doctor (CHPGMDC) as our response variable. There are 97 health regions in the data, where the number of respondents per health region ranges from 14 to 1025, with an average of 44 respondents and a standard deviation of 19. Observations in different health regions are considered independent; however, observations within the same health regions we treat as dependent. Therefore, we are dealing with clustered data, where health regions are treated as clusters.

We consider the following explanatory variables: sex of the respondent (DHH_SEX), type of smoker (SMK_202), an indicator of whether a respondent has had any asthma symptoms (CCC_035) within the last 12 months of the survey, and the Body Mass Index or BMI class level (HWTGISW). All variables are categorical, and are coded as follows (variable names in capital letters are the original variable names, while the names of variables used in the model are provided in brackets):

- DHH_SEX (sex) = 1 for male and 0 for female;

- SMOKE1 (smoke_daily) = 1 for a daily smoker and 0 otherwise, while SMOKE2 (smoke_occas) = 1 for an occasional smoker and 0 otherwise;
- CCC_035 (symptoms) = 1 if an asthma patient has had asthma symptoms within the last 12 months of the survey and 0 otherwise;
- HWTGISW1 (bmi_low) = 1 if a patient is underweight (based on their BMI class) and 0 otherwise, HWTGISW2 (bmi_high) = 1 if the patient is overweight and 0 otherwise, and HWTGISW3 (bmi_high) = 1 if the patient is obese and 0 otherwise;

All records in which one of the above predictors does not contain the actual value have been removed. We define an actual value as a variable value that is different from one of the following: 'Not Applicable', 'Refusal', 'Not Stated' or 'Don't know'. A total of 86 records were removed due to response variable (CHPGMDC) not having an actual value. The CCC_035 predictor did not have an actual value in 18 records, while a total of 31 records were removed due to missing data in SMK_202 predictors. Additionally, a total of 941 records were removed due to BMI class predictor (HWTGISW) not having the actual value: 333 of these records had a missing value, while the question about the BMI class level did not apply to 607 asthma patients.

There is a potential, as with all real data, that the data is not missing completely at random (MCAR) [118]. This is especially true given that the answers are self-reported. For example, refusing to give one's BMI class level may not be treated the same way as when the given question is not applicable. However, for illustrative purposes, and with the objective of comparing the GLMM-DP method with the existing methods, we treat all missing data as MCAR.

Our choice of coding scheme makes a female patient, who does not smoke, has a normal BMI class level and who has not had any asthma symptoms within the last 12 months of the survey, a baseline or a reference case.

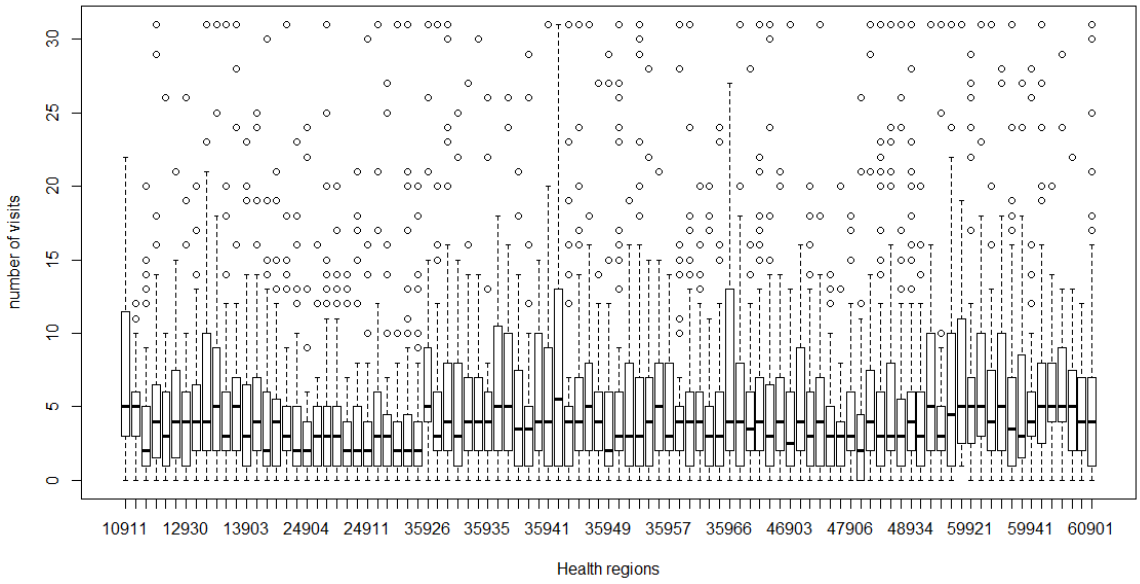

Figure 6.1: Box plot of the number of doctor visits per health region

Figure 6.1 shows the distribution of the number of visits to a doctor's office by asthma patients across different health regions. It can be seen visually that there are groups of health regions that have very similar average number of visits. The presence of these groups are an indication that the generalized linear mixed model may be suitable to model this data.

A common approach to building a GLMM model in frequentist statistics would

include fixed effect parameters that describe the change in the mean response at the population level, and a subset of these parameters as random effects that would describe the deviation of the mean response at a unit or health region level. Then the predicted value of the random effect of a particular health region would be used to estimate the number of visits to a doctor's office for that particular health region.

In Bayesian statistics, all unknown parameters are described with a probability distribution, and both fixed and random effect parameters are treated the same way. Therefore, we model the average number of visits per health region directly (as opposed to it being a deviation from the population average). In order to avoid identifiability issues, as explained in Section 4.2.2, we define our model so that the column space of $\boldsymbol{X}$ and the column space of $\boldsymbol{Z}$ are disjoint. One way of ensuring that this is the case is to include the intercept only in random effects parameters.

We use the GLMM-DP method to group the health regions with a similar relationship between the average number of visits to a doctor's office and the predictors described above, as defined by the parameters of the underling generalized linear mixed model. The average number of visits is taken for a reference case, which in our model is a female patient, who does not smoke, has a normal BMI class level and who has not had any asthma symptoms within the last 12 months of the survey. The average number of visits in a health region is represented by a (random) intercept, and the GLMM-DP method carries out the clustering based on these values. In addition to clustering, the method also estimates the parameters of the model for each cluster. Grouping health regions allows us to borrow strength across different units and improve the precision of the estimates.

Let y_{ij} be the number of visits to a doctor's office by the j^{th} asthma patient in the i^{th} health region, $i = 1, 2, \ldots, 97$ and $j = 1, 2, \ldots, n_i$. As indicated in the previous section, n_i ranges from 14 to 102. The conditional mean of y_{ij} depends on both fixed and random effects parameters via the following log-link function and linear predictor:

$$\log(E(y_{ij}|\boldsymbol{\beta}, \boldsymbol{b})) = \eta_{ij} = \boldsymbol{X}_{ij}\boldsymbol{\beta} + \boldsymbol{Z}_{ij}\boldsymbol{b}_i, \tag{6.1}$$

where $\boldsymbol{\beta}$ are fixed effects parameters and $\boldsymbol{b}_i$ are random effect parameters associated with the i^{th} health region, and $\boldsymbol{X}_{ij}$ and $\boldsymbol{Z}_{ij}$ are vectors of explanatory variables associated with the fixed and random effects, respectively. For each observation y_{ij}, we collect multiple predictors for fixed effects, and include only one predictor for random effect parameters. Therefore, $Z_{ij} = 1$ for all i, j. This is often referred to as the random intercept model.

To impose the grouping over health regions, we place a Dirichlet Process prior on b_i, i.e.,

$$\begin{aligned} b_i|G &\sim G, \\ G|\alpha, G_0 &\sim \mathrm{DP}(\alpha, G_0). \end{aligned} \tag{6.2}$$

To complete the model, we place a multivariate normal prior on the fixed effect parameters given by $MVN(\boldsymbol{\mu}_{\boldsymbol{\beta}}, \boldsymbol{\Sigma}_{\boldsymbol{\beta}})$ and a gamma prior for the concentration parameters with hyper-parameters a_α and b_α, i.e., $\alpha|a_\alpha, b_\alpha \sim \mathrm{Gamma}(a_\alpha, b_\alpha)$. The full specification is similar to that of Eq.(4.3) except that this model does not include the dispersion parameter ν since our response variable is modeled using the Poisson distribution.

6.3 Estimation and Convergence

In this section, we describe how we choose the priors of our model, and investigate the convergence of posterior distributions of the model parameters.

6.3.1 Choosing prior parameters

Setting prior parameters is a difficult task, and needs to be approached with care. This is especially true for the concentration parameter α of the Dirichlet Process prior, which allows for great flexibility in building a model with different number of mixture components. As per Hastie [110], we built the model with different values of hyper-parameters of the prior for the concentration parameters (a_α and b_α), and each time we got the model with the same average number of mixture components, which is 3. Therefore, we set the hyper-parameter values as: $a_\alpha = 2$ and $b_\alpha = 1$.

We expect our parameter estimates not to be very large (since they represent additive effects on the log scale). Therefore, we set the mean of the prior of fixed effects parameters $\boldsymbol{\beta}$ to have mean 1 and diagonal covariance matrix with large variances (all equal to 100). As for random effects parameters, we set the base distribution of the Dirichlet Process to be (univariate) normal with mean 1 and variance of 100. Setting the variance to a large value allows one to explore a large space of the posterior distribution while still being weakly-informative.

6.3.2 Parameter estimation

Given the data $\boldsymbol{y} = (\boldsymbol{y}_1^t, \boldsymbol{y}_2^t, \ldots, \boldsymbol{y}_{97}^t)^t$, where $\boldsymbol{y}_i = (y_{i1}, y_{i2}, \ldots, y_{in_i})^t$ represents a vector of counts (doctor visits) for the i^{th} health region, the posterior distribution of

the parameters $\Theta = \{\alpha, \boldsymbol{\beta}, b_1, \ldots, b_{97}\}$ is given by

$$\begin{aligned} P(\Theta|\boldsymbol{y}) \propto & P(\Theta) \times L(\Theta) \\ \propto & P(\Theta) \times \prod_{i=1}^{97} \exp\Big(\frac{\boldsymbol{y}_i\boldsymbol{\eta}_i - q(\boldsymbol{\eta}_i)}{\nu} + k(\boldsymbol{y}_i, \nu)\Big) \\ \propto & P(\Theta) \times \exp\Big(\boldsymbol{y}^T \times \boldsymbol{\eta} - \mathbf{1}^T \times q(\boldsymbol{\eta}) + \mathbf{1}^t \times k(\boldsymbol{y})\Big), \end{aligned} \tag{6.3}$$

where $\boldsymbol{\eta} = (\eta_1, \eta_2, \ldots, \eta_{97})^t$, $q(\eta_i) = \exp(\eta_i)$, $q(\boldsymbol{\eta}) = (q(\eta_1), q(\eta_2), \ldots, q(\eta_{97}))^t$, $k(y_{ij}, \nu) = \log(y_{ij}!)$, $k(\boldsymbol{\eta}, \nu) = (k(\eta_1), k(\eta_2), \ldots, k(\eta_{97}))^t$. Here $P(\Theta)$ is the product of individual priors of parameters, and is given by

$$P(\Theta) = \mathrm{G}(\boldsymbol{b}) \times \mathrm{Gamma}(\alpha|a_\alpha, b_\alpha) \times \mathrm{MVN}(\boldsymbol{\beta}|\boldsymbol{\mu}_{\boldsymbol{\beta}}, \boldsymbol{\Sigma}_{\boldsymbol{\beta}}), \tag{6.4}$$

where b is Dirichlet Process distributed. The inference proceeds as described in the Chapter 4.

6.3.3 Checking convergence of posterior distribution

It is a well-established fact that it is difficult to prove that the posterior distribution converges to the stationary distribution. As a result, we often look for signs that it does not converge. First, to determine what percent of the chain to use as a burn-in, we use the Geweke's diagnostic [66], as described in Chapter 4. It compares the mean of the first part of the chain (10% by default) to the mean of the last part of the chain (50% by default). If the posterior distribution has converged to the stationary distribution, then the difference between these two means would be asymptotically normal, and checking the convergence of the posterior distribution reduces to testing if means of the two intervals of the chain are equal. Table 6.1 shows the values of the Geweke's statistic for both random and fixed effects parameters.

All values of the Geweke's statistics in Table 6.1 were obtained using 20% of the samples as the first part of the chain, and the last 50% of samples as the second part of the chain. Since all values are within $[-1.96, 1.96]$ range, we discard the first 20% of the samples as part of the burn-in interval. Given that we have run our simulations for $50,000$ iterations, we drop the first $10,000$ samples and use the remaining $40,000$ for estimation.

Table 6.1: Geweke's statistic for fixed and random effects parameters in the CCHS model

Parameter	Geweke's statistic
symptoms	0.9284
sex	1.6387
smoke_daily	-0.1732
smoke_occas	-0.5337
bmi_low	-0.0767
bmi_high	0.0462
bmi_obese	-0.2848
b_1	0.05268
b_2	-0.5436
b_3	-0.3467

Figure 6.2 shows the autocorrelation plot between successive samples of random effects parameters in their posterior distribution.

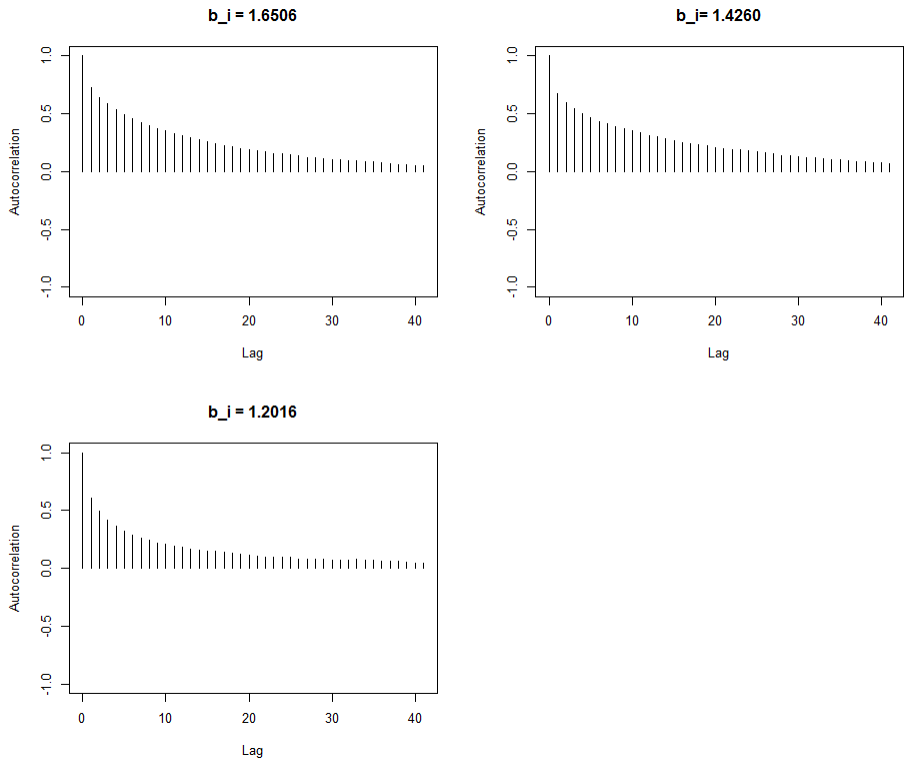

Figure 6.2: Autocorrelation plots for random effect parameters in the CCHS model

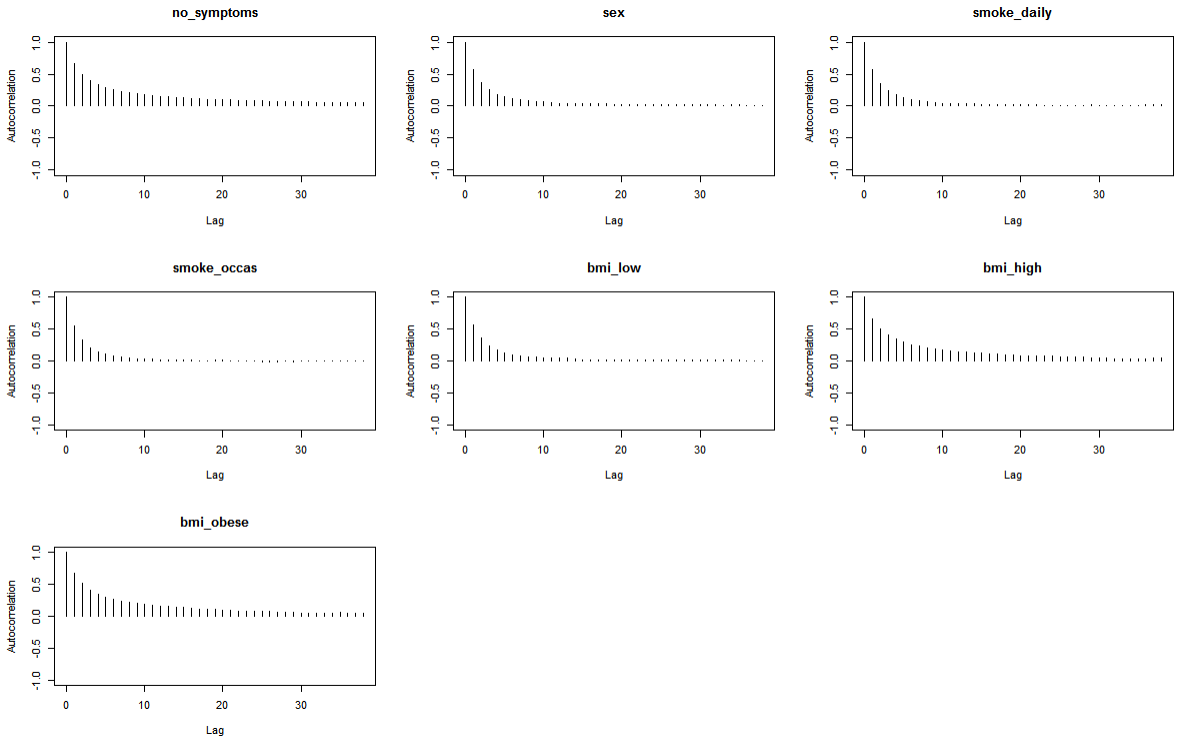

Figure 6.3: Autocorrelation plots for fixed effect parameters in the CCHS model

Figure 6.3 shows the autocorrelation plot between successive samples of fixed effects parameters from their posterior distribution. In both plots (for fixed and random effects parameters) we see that the correlation between successive samples drops fairly quickly, though not as quickly in the case of random effects parameters as fixed effects parameters.

Figure 6.4 shows trace plots and density plots for random effects parameters. We see that all random effects parameters exhibit good mixing and that the estimated

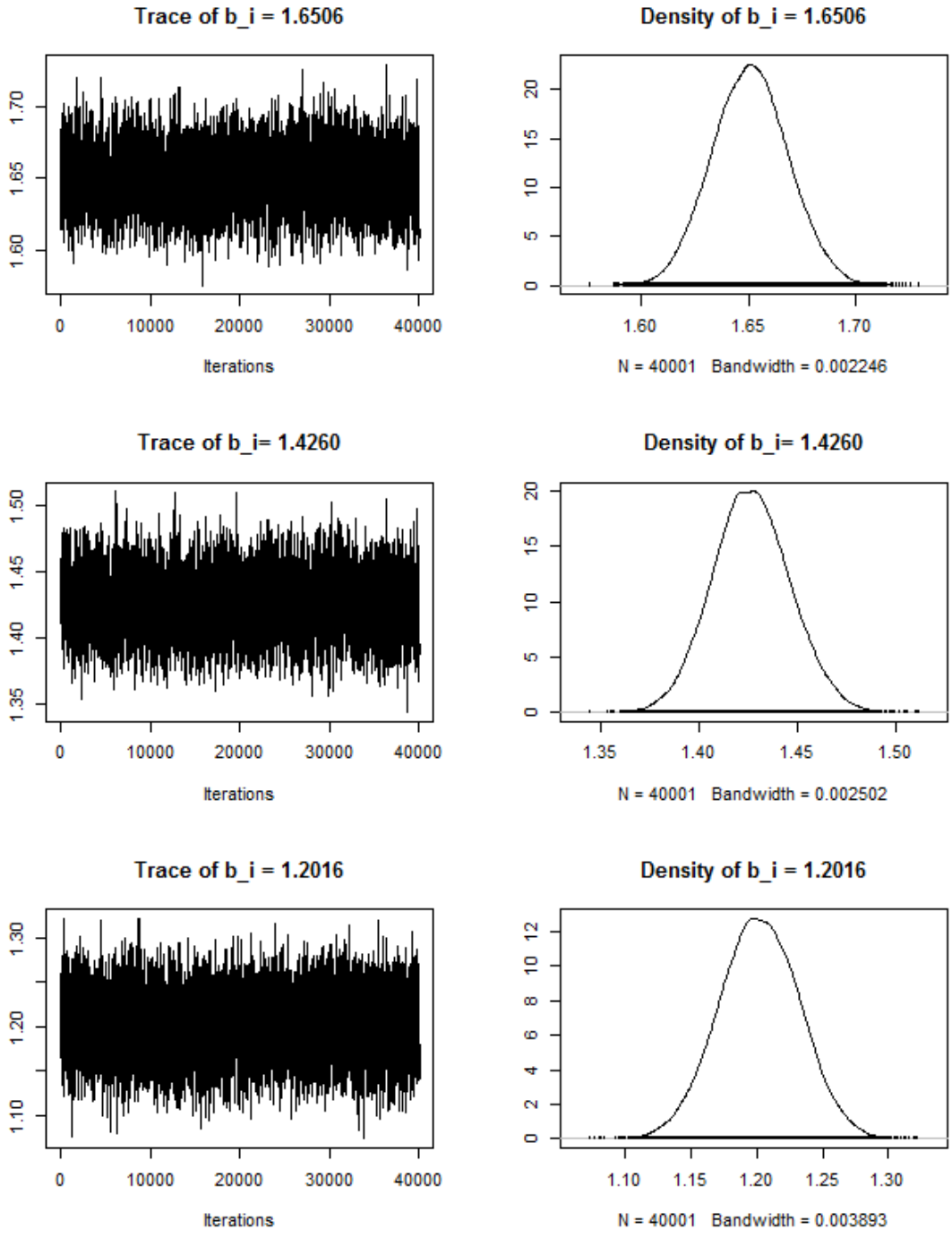

Figure 6.4: Trace and density plots of random effects parameters of clusters in the CCHS model as identified by the GLMM-DP method

density function is not flat, which shows that the label switching solution in Section 4.4 produces a good reference partition. Fixed effects parameters show similar patterns.

6.4 Results

We simulate $50,000$ samples from the posterior distribution of the parameters Θ, as defined in Section 6.3.2. We drop the first $10,000$ samples (as they are considered part of the burn-in period) and use the remaining $40,000$ samples for estimation, as described in Section 6.3.3. The overall acceptance rate for the Metropolis-Hastings algorithm in sampling fixed effects parameters is 37.89% and for random effects parameters it is 31.94%. The overall acceptance rate for the fixed effects parameters, which are changed at most once per iteration, gives us a measure of the number of times the fixed effects parameters were updated, over the total number of iterations. For random effects parameters, we define the overall acceptance rate as the average (over all iterations) of the average rate (per iteration) that random effects parameters have been updated (since we may have different number of mixture components in each iteration).

The GLMM-DP method identifies three groups of health regions. Their sizes are 41, 39, and 17. Table 6.2 shows estimates of random effects parameters for each cluster, as well as their 95% highest posterior density (HPD) intervals.

Table 6.3 shows the estimates of fixed effects parameters, as produced by the GLMM-DP method.

Table 6.2: GLMM-DP method: random effect parameter estimates

cluster (size)	Mean	95%HPD-Low	95%HPD-Upp
cluster 1 (40)	1.6507	1.6167	1.6856
cluster 2 (39)	1.4263	1.3883	1.4654
cluster 3 (17)	1.2018	1.1443	1.2639

Table 6.3: GLMM-DP method: fixed effect parameter estimates

Parameter	Mean	95%HPD-Low	95%HPD-Upp
symptoms	0.2678	0.2407	0.2936
sex	-0.3664	-0.3950	-0.3370
smoker_daily	0.0846	0.0530	0.1167
smoker_occas	0.1425	0.0807	0.2020
bmi_low	0.2172	0.1358	0.2947
bmi_high	0.0882	0.0557	0.1223
bmi_obese	0.2617	0.2295	0.2922

Table 6.4 shows the results obtained from the ordinary ML method (glmmML) [116]. We call this the Full Model.

Table 6.4: Parameter estimates using glmmML method (Full Model)

Parameter	Estimate	Standard Error	z value	$Pr(>\|z\|)$
(Intercept)	1.4983	0.0271	55.241	< 0.0001
symptoms	0.2579	0.0136	18.987	< 0.0001
sex (female)	-0.3595	0.0150	-23.904	< 0.0001
smoke_daily	0.1276	0.0167	7.616	< 0.0001
smoke_occas	0.1390	0.0310	4.481	< 0.0001
bmi_low	0.2330	0.0408	5.714	< 0.0001
bmi_high	0.0791	0.0168	4.701	< 0.0001
bmi_obese	0.2665	0.0156	16.091	< 0.0001

We see that both the GLMM-DP and ML methods produce very similar results. Both methods find all predictors to be statistically significant. The estimates of random effects parameters are in line with the estimate of the intercept produced by the ML method. The weighted average of all random effects parameters is 1.4759, which is a bit lower than 1.4983 (the intercept of the ML method).

Applying the ML method on each cluster obtained using the GLMM-DP method produces the results in Table 6.5. We refer to models built on these clusters as Model 1, Model 2 and Model 3, respectively. All predictors are found to be significant when applied on the Full Model (by both DP-GLMM and ML methods); however in Table 6.5 we see that occasional smokers predictor is not significant in Model 1, and high level of BMI is not significant in Model 3. We see also that daily

Table 6.5: Parameter estimates from ML method for three different clusters

Parameter (cluster)	Estimate	Standard Error	z value	$Pr(>\|z\|)$
cluster 1 ($\hat{\sigma} = 0.0270$)				
(Intercept)	1.5369	0.0422	36.4611	< 0.0001
symptoms	0.2311	0.0202	11.4345	< 0.0001
sex (female)	-0.3676	0.0223	-16.5153	< 0.0001
smoke_daily	0.1125	0.0247	4.5441	< 0.0001
smoke_occas	0.0340	0.0474	0.7165	0.4740
bmi_low	0.1822	0.0608	2.9975	0.0027
bmi_high	0.0771	0.0252	3.0651	0.0022
bmi_obese	0.2585	0.0249	10.3597	< 0.0001
cluster 2 ($\hat{\sigma} = 0.0292$)				
(Intercept)	1.4727	0.0449	32.807	< 0.0001
symptoms	0.2623	0.0210	12.515	< 0.0001
sex (female)	-0.3972	0.0231	-17.212	< 0.0001
smoke_daily	0.1974	0.0256	7.720	< 0.0001
smoke_occas	0.2204	0.0451	4.884	< 0.0001
bmi_low	0.1571	0.0643	2.444	0.0145
bmi_high	0.1004	0.0258	3.894	< 0.0001
bmi_obese	0.3106	0.0253	12.262	< 0.0001
cluster 3 ($\hat{\sigma} = 0.0334$)				
(Intercept)	1.4627	0.0567	25.8031	< 0.0001
symptoms	0.3364	0.0385	8.7711	< 0.0001
sex (female)	-0.1961	0.0439	-4.4662	< 0.0001
smoke_daily	-0.1020	0.0511	-1.9972	0.0458
smoke_occas	0.2536	0.1005	2.5226	0.0117
bmi_low	0.7124	0.01085	6.5682	< 0.0001
bmi_high	0.0221	0.0477	0.4634	0.6430
bmi_obese	0.1560	0.0466	3.3430	0.0008

smokers are barely significant (at 5% confidence level) in Model 3.

Clusters in Table 6.2 are ordered by the estimated values of their random effects (Cluster 1 has the highest and cluster 3 has the lower value), and Table 6.5 shows that this order is maintained in intercepts of the three models, though they do not seem to be as well separated as the cluster effects.

In all three clusters we see that patients with asthma symptoms are going to visit doctor's office more often than patients who have not had any asthma symptoms in 12 months prior to the survey, and this ranges from 26% more visits (for patients in cluster 1 ($0.26 = \exp(0.2311) - 1$) to 40% more visits for patients in cluster 3 ($0.40 = \exp(0.3364) - 1$). Also, male patients are going to make fewer visits than female patients across all clusters, with the biggest difference in cluster 1 (30% fewer visits, $0.30 = 1 - \exp(-0.3676)$) and cluster 2 (32.78% fewer visits, $0.3278 = 1 - \exp(-0.3972)$), while those in cluster 3 make only 17.81% fewer visits.

Both daily and occasional smoking are significant predictors in the full model, and patients who smoke daily or occasionally will visit doctor's office 13.61% and 14.91% more often, respectively. However, in cluster 1 we see that occasional smoking is not significant in explaining the number of doctor's visits. On the other hand, the daily smoking predictor was significant in the Full Model and had a positive effect, while in Model 3 it is barely significant (and has a negative effect on the outcome).

Both the Full Model and all three submodels in Table 6.5 show that the patients with any-but-normal BMI level will visit doctor's office more often than patients with normal BMI level. Patients with low BMI level in cluster 1 and cluster

2 will visit a doctor's office 20% and 17% more often, respectively, than patients with normal BMI level, while patients with the same BMI level in cluster 3 will visit a doctor's office one full visit more often than patients with normal BMI level. Furthermore, the low level of BMI is not a significant predictor of visits in cluster 3. Patients in cluster 1 with obese BMI level will visit a doctor's office roughly the same number of times as patients with the same BMI level for the Full Model (29.50% more visits, $0.2950 = \exp(0.2585) - 1$). Similar results hold for cluster 2 and the Full Model, while those patients in cluster 3, with the same level of BMI will also make more visits but less than those patients in cluster 1 and 2.

Figure 6.5 shows the map of health regions in Ontario, colored according to the cluster to which they belong. Health regions with the smallest average number of visits (by the reference case) are colored in green, those health regions with higher number of visits are colored in blue, and the remaining health regions (the largest group) is colored in red. There does not seem to be an obvious pattern of geographic aspect that may explain how the number of visits to a doctor's office may differ from one region to another. Further study would be required in order to find out what these aspects may be. It could be a particular industry or perhaps some other environmental factors.

6.5 Conclusion and Summary

We have shown in this chapter that the GLMM-DP method produces parameter estimates that are very similar to those obtained by a frequentist maximum likelihood

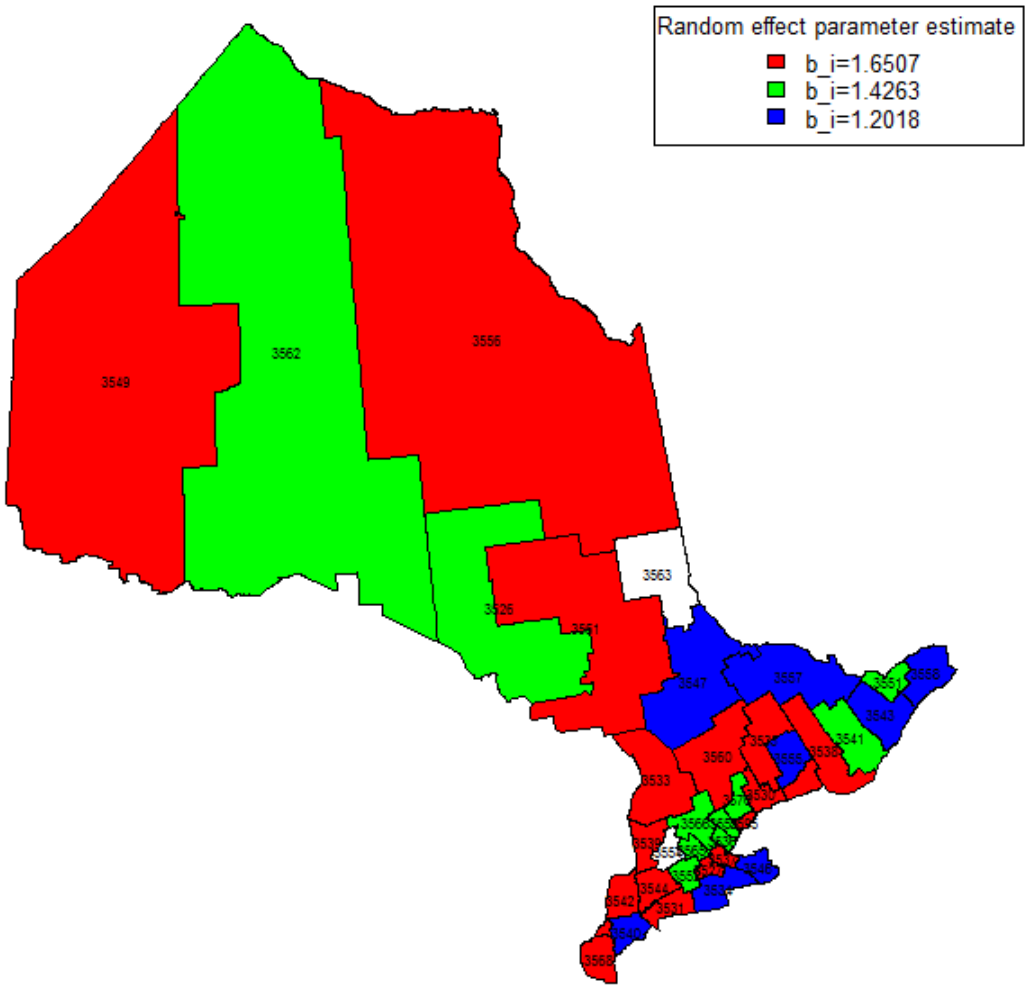

Figure 6.5: Ontario Health Regions, as classified by the GLMM-DP method.

method. However, our method also identifies more homogeneous clusters in the data. When the same maximum likelihood method is applied on these clusters, the parameters of the new models may differ significantly from those obtained on the full dataset. For example, some predictors that were significant in the full model (the model built on a full dataset), such as occasional smoking, may become insignificant in some clusters. Also, the parameters in different clusters may show different trends (across different clusters versus the full dataset). For example, some parameters may have positive effect on the mean response in the full dataset and negative effect on

some clusters (though this effect has very small significance level). In addition to that, parameter estimates on the whole dataset may be different from the parameter estimates on clusters produced by the GLMM-DP method, and parameter estimates may differ in models built on different clusters. These insights cannot be obtained using the existing methods (without considerable effort), which clearly demonstrates the benefit of the GLMM-DP method.

Chapter 7

Clustering GLMM Profiles in Multivariate Settings

In this chapter, we extend the GLMM-DP method to multivariate settings. After a short introduction, we define our model and describe the sampling techniques from posterior distributions of model parameters. We then test the method on a small data set, and conclude the chapter with a small simulation study.

7.1 Introduction

In longitudinal studies, one often observes multiple outcomes on the same unit at each observation time. These are known as multivariate outcomes. Outcomes associated with the same unit may be of different types. For example, one could observe one or more outcomes of count type and one or more outcomes of continuous type. Joint modeling of multiple outcomes are often of direct interest in the analysis. A common modeling approach is based on mixed models, where each outcome is modeled separately using general or generalized linear mixed models, and a common multivariate distribution is assumed on all random effects parameters. These models

are called joint mixed models, and are built on the assumption that outcomes associated with a unit are conditionally independent given the random effects.

Joint mixed models have also been used in clustering longitudinal data. Outcomes may be of the same or different types, leading to joint models at the component level that may be any combination of linear or non-linear types. For example, Dai [119] proposes a method that jointly models Gaussian and beta distributed data. Villarroel [120] proposes a method where each component is non-linear mixed effects model. Qin [121] clusters profiles by grouping parameters of regression models.

Methods proposed by Komarek [100] and Gueorguieva [122] are designed for clustering multivariate outcomes where each outcome has distribution that is a member of an exponential family, and are very similar to the method we propose here. While Gueorguieva [122] estimates parameters using the Monte Carlo EM method, the method proposed by Komarek [100] is fully Bayesian. However, both methods require that the number of clusters be known in advance. The GLMM-DP method does not impose such restrictions on the model.

7.2 Model Description

First, we extend the notation used in the GLMM-DP method to multivariate settings. We assume there are N units in the study. For unit $i, 1 \leq i \leq N$, at each observation time $j, 1 \leq j \leq n_i$, we record M different outcomes $y_{imj}, 1 \leq m \leq M$, and for each outcome y_{imj} we record a vector of covariates $\boldsymbol{X}_{imj}$ associated with fixed effects parameters, and a vector $\boldsymbol{Z}_{imj}$ of covariates associated with random effects parameters. Using the same notation as in Section 1.1, we denote by $\boldsymbol{y}_{im}$ the vector

of all measurements on outcome m for unit i, i.e., $\boldsymbol{y}_{im} = (y_{im1}, y_{im2}, \ldots, y_{imn_i})^t$, and corresponding matrices of covariates by $\boldsymbol{X}_{im}$ and $\boldsymbol{Z}_{im}$. The vector of all observations on outcome m from all units at all times is denoted by $\boldsymbol{y}_m = (\boldsymbol{y}_{1m}^t, \boldsymbol{y}_{im}^t, \ldots, \boldsymbol{y}_{im}^t)^t$, and its corresponding matrices $\boldsymbol{X}_m$ and $\boldsymbol{Z}_m$. Finally, we denote the vector of all responses in a data set by $\boldsymbol{y} = (\boldsymbol{y}_1^t, \boldsymbol{y}_2^t, \ldots, \boldsymbol{y}_M^t)^t$, and matrices of associated covariates by $\boldsymbol{X}$ and $\boldsymbol{Z}$.

We assume that the response vector $\boldsymbol{y}$ consists of M univariate outcomes, and the distribution of each univariate outcome is a member of the exponential family, i.e.,

$$p(y_{imj}|\nu_m, \boldsymbol{\beta}_m, \boldsymbol{b}_{im}) = \exp\left\{\frac{y_{imj}\eta_{imj} - q(\eta_{imj})}{\nu_m} + k(y_{imj}, \nu_m)\right\}, \tag{7.1}$$

where η_{imj} is the linear predictor, defined as

$$\begin{aligned} h_m(E(Y_{imj}|\boldsymbol{\beta_m}, \boldsymbol{b}_{im})) =& \eta_{imj} = \boldsymbol{X}_{imj}^t\boldsymbol{\beta_m} + \boldsymbol{Z}_{imj}^t\boldsymbol{b}_{im}, \\ & i = 1, 2, \cdots, N, j = 1, 2, \cdots, n_i. \end{aligned} \tag{7.2}$$

In Eq.(7.2), $\boldsymbol{\beta}_m$ is the vector of fixed effects parameters for the m^{th} model (associated with m^{th} outcome), $\boldsymbol{b}_{im}$ is the random effects parameter for unit i and outcome m, and $h_m(\cdot)$ is a link function for outcome m as defined in Section 2.2. Hence the response vector may contain both continuous and discrete responses.

We model the dependence of a multivariate response vector $\boldsymbol{Y} = (Y_1, Y_2, \ldots, Y_M)^t$ on a set of predictors by modeling the mean structure of each individual response variable, as in Eq.(7.1), and placing a common prior on a vector of random effects parameters of all individual models. So, for a multivariate model with M outcomes, each outcome is modeled as a generalized linear mixed model with a specific random

effect $b_{ij}, 1 \leq j \leq M$, where the random effect for the multivariate outcome is $\boldsymbol{b}_i = (b_{i1}, b_{i2}, \cdots, b_{iM})$. Dependence between different multivariate outcomes is modeled by placing a prior on $\boldsymbol{b}_i, 1 \leq i \leq N$.

To perform clustering of longitudinal profiles in multivariate settings, we choose the prior of $\boldsymbol{b}_i$ to be a Dirichlet Process $G(\alpha, G_0)$, where G_0 is the base distribution with support covering the space of all $\boldsymbol{b}_i$.

The full model is now defined as

$$\begin{aligned} y_{imj} \mid \boldsymbol{\beta}, \boldsymbol{b}_i, \nu_m &\propto \exp\left\{\frac{y_{imj}\eta_{imj} - q(\eta_{imj})}{\nu_m} + k(y_{imj}, \eta_{imj})\right\}, \\ \boldsymbol{b}_i | G &\propto G, \\ G \mid \alpha, G_0 &\propto G(\alpha, G_0), \\ \boldsymbol{\beta} \mid \boldsymbol{\mu}_{\boldsymbol{\beta}}, \boldsymbol{\Sigma}_{\boldsymbol{\beta}} &\propto \text{MVN}(\boldsymbol{\mu}_{\boldsymbol{\beta}}, \boldsymbol{\Sigma}_{\boldsymbol{\beta}}), \end{aligned} \tag{7.3}$$

where the base distribution (G_0) of the Dirichlet Process is MVN($\boldsymbol{\mu}_{\boldsymbol{b}}, \boldsymbol{\Sigma}_{\boldsymbol{b}}$).
Additionally, we place priors on $\boldsymbol{\Sigma}_b^{-1}$ and ν_m^{-1} as in [100], and add prior on hyper-parameters of the dispersion parameters, as follows:

$$\begin{aligned} {\boldsymbol{\Sigma}_{\boldsymbol{b}}}^{-1} \mid \rho, \boldsymbol{R} &\propto \text{Wishart}(\rho, (\rho\boldsymbol{R})^{-1}), \\ \nu_m^{-1} \mid \zeta_m, \xi_m &\propto \text{Gamma}(\zeta_m/2, \xi_m^{-1}/2), \\ \xi_m^{-1} \mid \phi_\xi, \omega_\xi &\propto \text{Gamma}(\phi_\xi, \omega_\xi). \end{aligned} \tag{7.4}$$

A matrix $\boldsymbol{X}$ of order p, is said to follow the Wishart distribution with ρ degrees of freedom and scale matrix $\boldsymbol{R}$, written as $W_p(\rho, \boldsymbol{R})$, if its density is given by

$$p(\boldsymbol{X} \mid \rho, \boldsymbol{R}) = \frac{|\boldsymbol{X}|^{(\rho-p-1)/2}\text{exp}(-\text{tr}(\boldsymbol{R}^{-1}\boldsymbol{X})/2)}{2^{\frac{\rho p}{2}}|\boldsymbol{R}|^{\rho/2}\Gamma_p(\frac{\rho}{2})}, \tag{7.5}$$

where $\Gamma_p()$ is a multivariate gamma function, and tr() is the trace function.

7.3 Parameter Estimation

The likelihood of the parameters $\Theta = (\alpha, \boldsymbol{\beta}, \boldsymbol{\nu}, \boldsymbol{c})$ is given by

$$
\begin{aligned}
L(\Theta) =& P(\boldsymbol{y} \mid \Theta) \\
=& P(\boldsymbol{y} \mid \alpha, \boldsymbol{\beta}, \boldsymbol{\phi}, \boldsymbol{c}) \\
=& \prod_{i=1}^{N} \prod_{m=1}^{M} P(\boldsymbol{y}_{im} | \boldsymbol{\beta}, \boldsymbol{\phi}_{c_i}) \\
=& \prod_{i=1}^{N} \prod_{m=1}^{M} \prod_{j=1}^{n_i} P(y_{imj} | \boldsymbol{\beta}_i, \boldsymbol{\phi}_{c_i}) \\
=& \prod_{i=1}^{N} \prod_{m=1}^{M} \prod_{j=1}^{n_i} \exp\left\{ \frac{y_{imj}\eta_{imj} - q(\eta_{imj})}{\nu_m} + k(y_{imj}, \nu_m) \right\} \\
=& \prod_{m=1}^{M} \prod_{i=1}^{N} \prod_{j=1}^{n_i} \exp\left\{ \frac{y_{imj}\eta_{imj} - q(\eta_{imj})}{\nu_m} + k(y_{imj}, \nu_m) \right\} \\
=& \prod_{m=1}^{M} \exp\left\{ \frac{\boldsymbol{y_m}^t \times \boldsymbol{\eta}_m - \mathbf{1}^t \times q(\boldsymbol{\eta}_m)}{\nu_m} + \mathbf{1} \times k(\boldsymbol{y}_m, \nu) \right\} \\
=& \exp\left\{ \sum_{m=1}^{M} \frac{\boldsymbol{y}_m^t \times \boldsymbol{\eta}_m - \mathbf{1}^t \times q(\boldsymbol{\eta}_m)}{\nu_m} + \mathbf{1}^t \times k(\boldsymbol{y}_m, \nu_m) \right\}.
\end{aligned}
\tag{7.6}
$$

The posterior distribution of $\Theta = (\alpha, \boldsymbol{\beta}, \boldsymbol{\nu}, \boldsymbol{c}, \boldsymbol{\Sigma}_b, \boldsymbol{\nu}, \boldsymbol{\xi})$ may be obtained as

$$
\begin{aligned}
P(\Theta \mid \boldsymbol{y}) \propto& P(\Theta) \times L(\Theta) \\
\propto& P(\alpha) \times P(\boldsymbol{c} \mid \alpha) \times P(\boldsymbol{\beta}) \times P(\boldsymbol{\phi} \mid \boldsymbol{c}) \times P(\boldsymbol{\Sigma}_b|\cdot) \times P(\boldsymbol{\nu}) \times P(\boldsymbol{\xi}) \\
& \exp\left\{ \sum_{m=1}^{M} \frac{\boldsymbol{y_m}^t \times \boldsymbol{\eta_m} - \mathbf{1}^t \times q(\boldsymbol{\eta_m})}{\nu_m} + \mathbf{1}^t \times k(\boldsymbol{y_m}, \nu_m) \right\}.
\end{aligned}
\tag{7.7}
$$

7.3.1 Sampling from Posterior Distributions

The derivation of the posterior distribution of multivariate model parameters is very similar to that of the univariate case as shown in Section 4.3. We sample the concentration parameter of the Dirichlet Process (α) as in Section 4.3.1 and the inverse of the dispersion parameter ν_m^{-1} for a model with outcome Y_m as in Section 4.3.5. To sample the allocation variables ($\boldsymbol{\phi}_{c_i}$), we follow the same approach as in Section 4.3.2, with the likelihood for a given random effects parameter calculated as the product of all outcome likelihoods. So, the likelihood of profile $\boldsymbol{y}_i$ for fixed $\boldsymbol{b}_i$ is given by

$$L(\boldsymbol{y}_i \mid \boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_M, \boldsymbol{b}_i) = \prod_{m=1}^{M} f(\boldsymbol{y}_{im} \mid \boldsymbol{\beta}_m, \boldsymbol{b}_i) \tag{7.8}$$

The posterior distribution of the precision matrix of the random effects parameters $\boldsymbol{\Sigma}_b$ is given by

$$\begin{aligned}
\boldsymbol{\Sigma}_b^{-1} \propto & \text{Wishart}(\rho, (\rho \boldsymbol{R})^{-1}) \times L(\boldsymbol{y} \mid \boldsymbol{b}_1, \ldots, \boldsymbol{b}_M) \\
\propto & |\boldsymbol{\Sigma}_b^{-1}|^{(\rho-p-1)/2} \exp\left\{-\frac{\text{tr}}{2}(\rho \boldsymbol{R} \boldsymbol{\Sigma}_b^{-1})\right\} |\boldsymbol{\Sigma}_b|^{-M/2} \exp\left\{-\frac{1}{2}\sum_{i=1}^{M}(\boldsymbol{b}_i - \boldsymbol{\mu_b})\boldsymbol{\Sigma}_b^{-1}(\boldsymbol{b}_i - \boldsymbol{\mu}_b)\right\} \\
\propto & |\boldsymbol{\Sigma}_b^{-1}|^{(\rho-p-1+M)/2} \exp\left\{-\frac{\text{tr}}{2}(\rho \boldsymbol{R} \boldsymbol{\Sigma}_b^{-1}) + \sum_{i=1}^{M}(\boldsymbol{b}_i - \boldsymbol{\nu}_b)(\boldsymbol{b}_i - \boldsymbol{\mu}_b)^t \boldsymbol{\Sigma}_b^{-1}\right\} \\
\propto & |\boldsymbol{\Sigma}_b^{-1}|^{(\rho-p-1+M)/2} \exp\left\{-\frac{\text{tr}}{2}\big(\rho \boldsymbol{R} + \sum_{i=1}^{M}(\boldsymbol{b}_i - \boldsymbol{\nu}_b)(\boldsymbol{b}_i - \boldsymbol{\mu}_b)^t\big)\boldsymbol{\Sigma}_b^{-1}\right\} \\
\propto & \text{Wishart}(M + \rho, \big(\rho \boldsymbol{R} + \sum_{i=1}^{M}(\boldsymbol{b}_i - \boldsymbol{\nu}_b)(\boldsymbol{b}_i - \boldsymbol{\mu}_b)^t\big)^{-1}),
\end{aligned} \tag{7.9}$$

which is again the Wishart distribution with updated parameters.

We sample fixed effects parameters from the posterior distributions using the same approach as in Section 4.3.4. Random effects parameters are sampled as in

Section 4.3.3, except that in this case, the covariance matrix of the proposal density (Eq.(4.17)) is a block-diagonal matrix in which each block is a covariance sub-matrix given by Eq.(4.17), derived according to the sub-model of the given outcome.

7.4 Simulation Study

In this section, we first describe how we simulate the data. Then we run our multivariate approach on a single data set and investigate the convergence of posterior distributions. Following that, we run a series of simulations with twelve different settings and describe the performance of the method.

7.4.1 Generating a dataset

We simulate a data set with $N = 50$ individuals, $n_i = 10$ observations per individual, and with a response vector of two outcomes ($M = 2$), i.e., $\boldsymbol{Y} = (Y_1, Y_2)^t$, where Y_1 is of continuous type and Y_2 is a count response. For each outcome, we have associated matrices of fixed and random covariates: $\boldsymbol{X}_1$ and $\boldsymbol{Z}_1$ for $\boldsymbol{y}_1$, and $\boldsymbol{X}_2$ and $\boldsymbol{Z}_2$ for $\boldsymbol{y}_2$. For simplicity, we assume that the two models share the same predictors, i.e., $\boldsymbol{X}_2 = \boldsymbol{X}_1$ and $\boldsymbol{Z}_2 = \boldsymbol{Z}_1$. Each model has its own fixed effects and random effects parameters. Also for simplicity, we assume that the components of the vector of random effects parameters are independent. This allows us to easily simulate the data by first generating the data for the Poisson sub-model (following the steps in Section 5.1) and then augmenting it with the normal sub-model by using the same covariance matrices as in the Poisson model and the given values of the model parameters.

Table 7.1 shows the true values of the parameters used in the simulation.

The above simulation generates data with two clusters, each cluster containing the

Table 7.1: Parameters used to simulate data from a joint model with continuous and count responses

Model	Fixed effects ($\boldsymbol{\beta}$)	Random effects
Poisson	$\boldsymbol{\beta}_1 = (0.8, -0.6, 0.3)$	$\mu_{b_{11}} = -0.5, \mu_{b_{21}} = 1.15$
Normal	$\boldsymbol{\beta}_2 = (-0.5, 0.5, 0.4)$	$\mu_{b_{12}} = 0.5, \mu_{b_{22}} = 1.05$

same number of units. For both outcomes, the first half of the units belongs to one cluster, and the second half of the units belongs to the second cluster. Therefore, we expect the profiles to be clustered into two groups.

7.4.2 Results on a single data set

We run the simulations for 20,000 iterations, and discard the first 2,000 iterations. Table 7.2 indicates that the chain may have converged to a stationary distribution after 2,000 iterations, as the values of the Geweke's statistic for all parameters are within [-1.96, 1.96] range.

Table 7.2: Geweke's statistic for fixed and random effect parameters for a joint mixture model with continuous and count responses

	β_1	β_2	β_3
Normal sub-model	0.9502	-0.4064	-0.9246
Poisson sub-model	0.9395	1.4953	0.6034
	b_1	b_2	
Normal sub-model	-0.9303	-0.7784	
Poisson sub-model	-0.5996	1.5041	

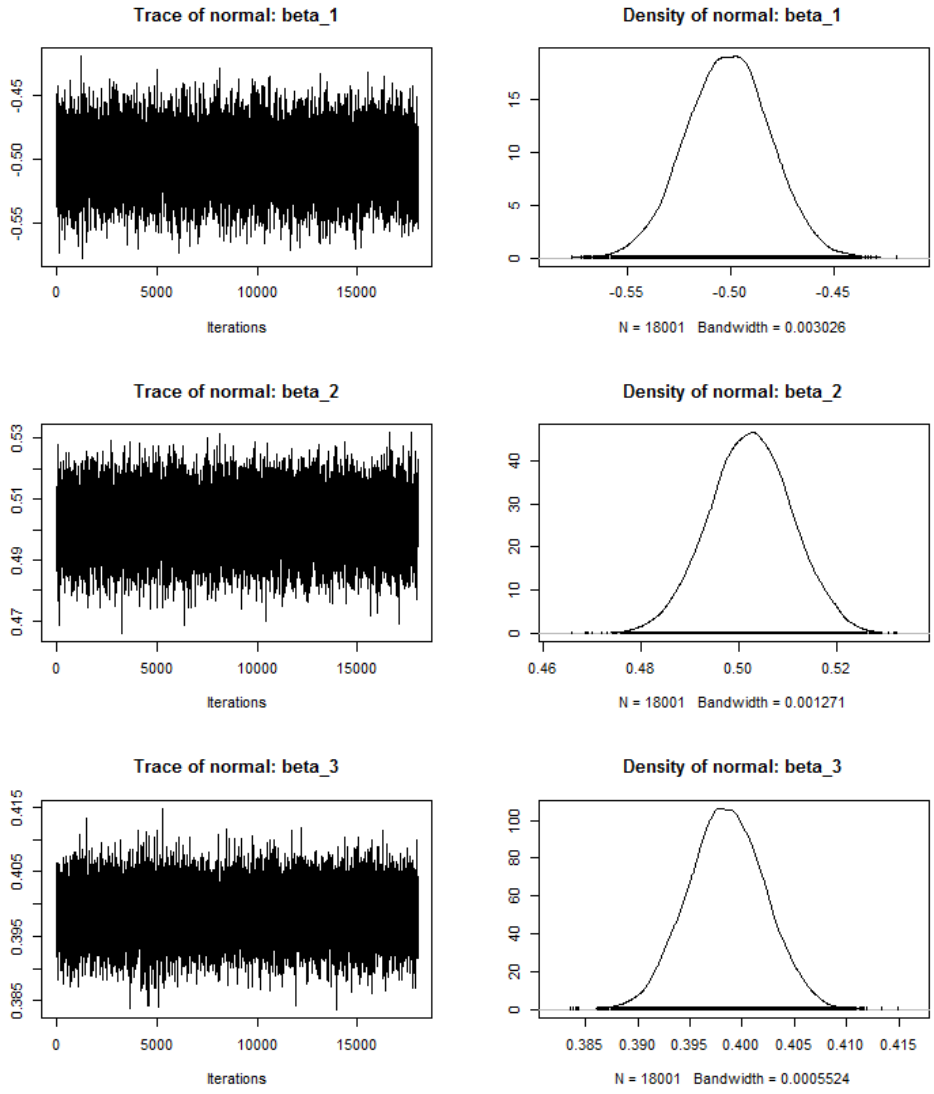

Figure 7.1: Trace and density plots of the fixed effects parameters of normal sub-model in the joint mixture model

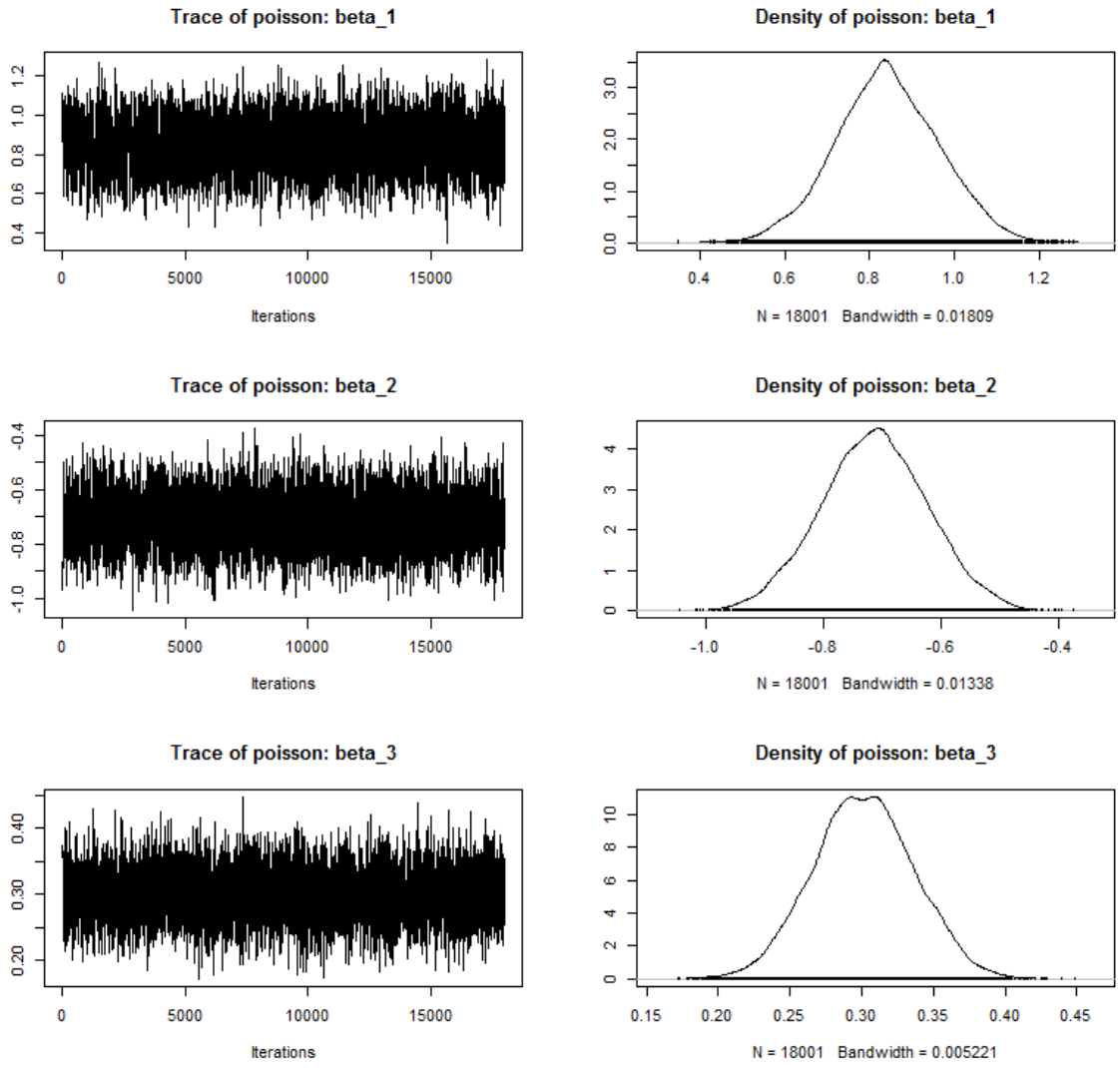

Figure 7.2: Trace and density plot of fixed effects parameters of the Poisson sub-model in the joint mixture model

Figures 7.1, 7.2 and 7.3 show trace plots and density plots of fixed effects parameters in the normal sub-model, fixed effects parameters in the Poisson sub-model

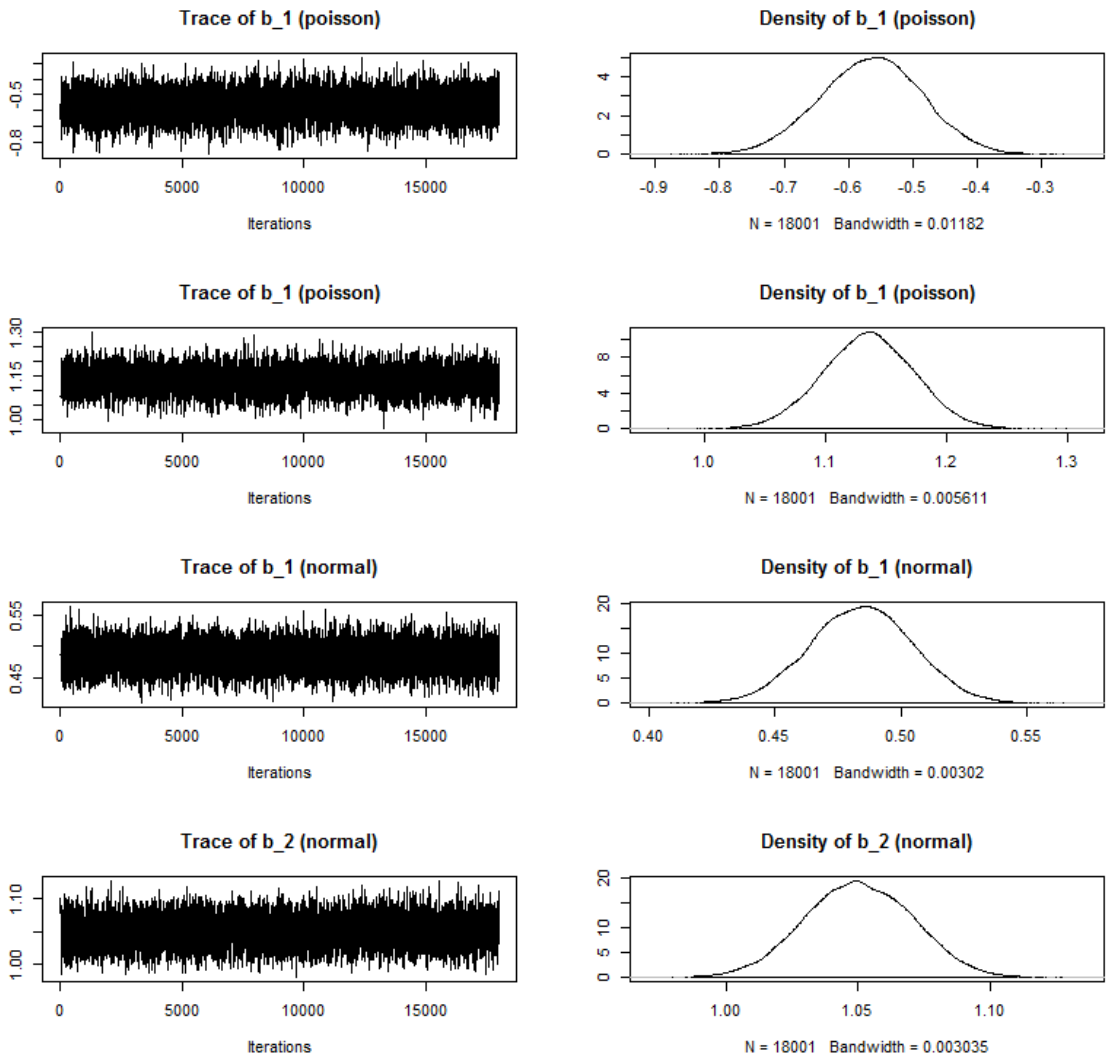

Figure 7.3: Trace and density plot of the random effects parameters in the joint mixture model

and the random effects parameters in the joint model, respectively. All plots indicate that the posterior distribution seems to reach a stationary distribution. The density plots of fixed effects parameters in the Poisson model seem to imply some correlation between consecutive samples (Figure 7.2) however, the autocorrelation plot in Figure 7.6 shows that the autocorrelation drops fairly rapidly.

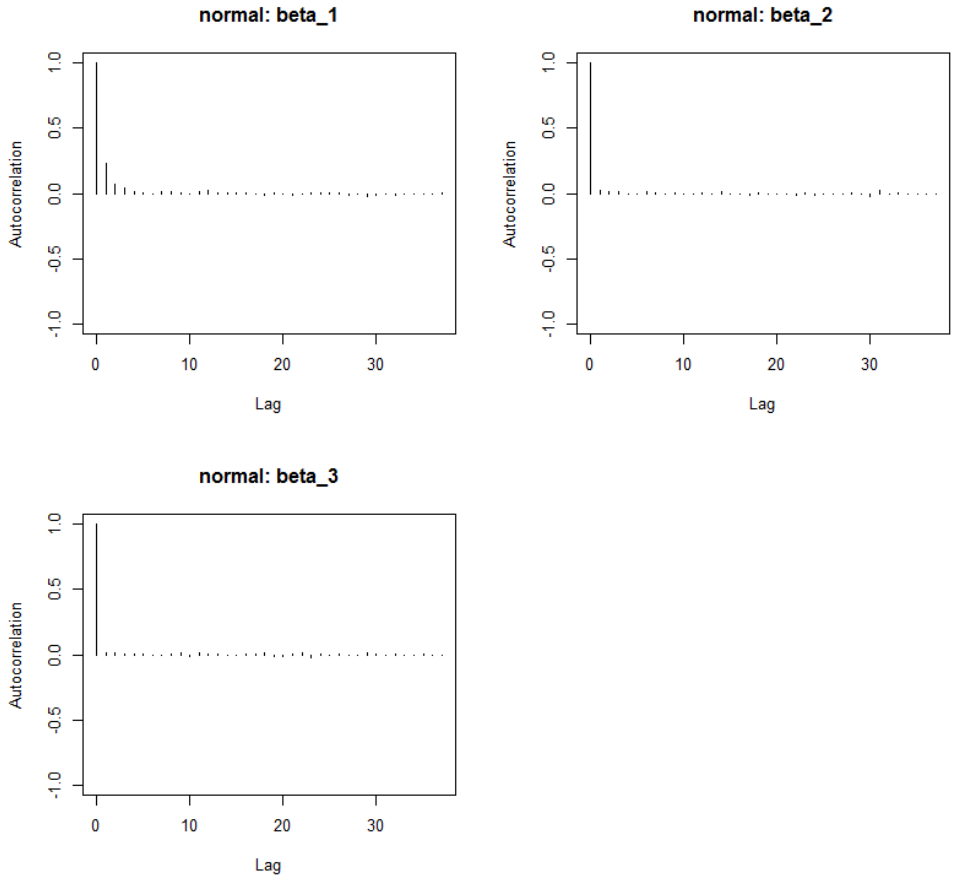

Figure 7.4: Autocorrelation of the fixed effects parameters for the normal sub-model in the joint mixture model

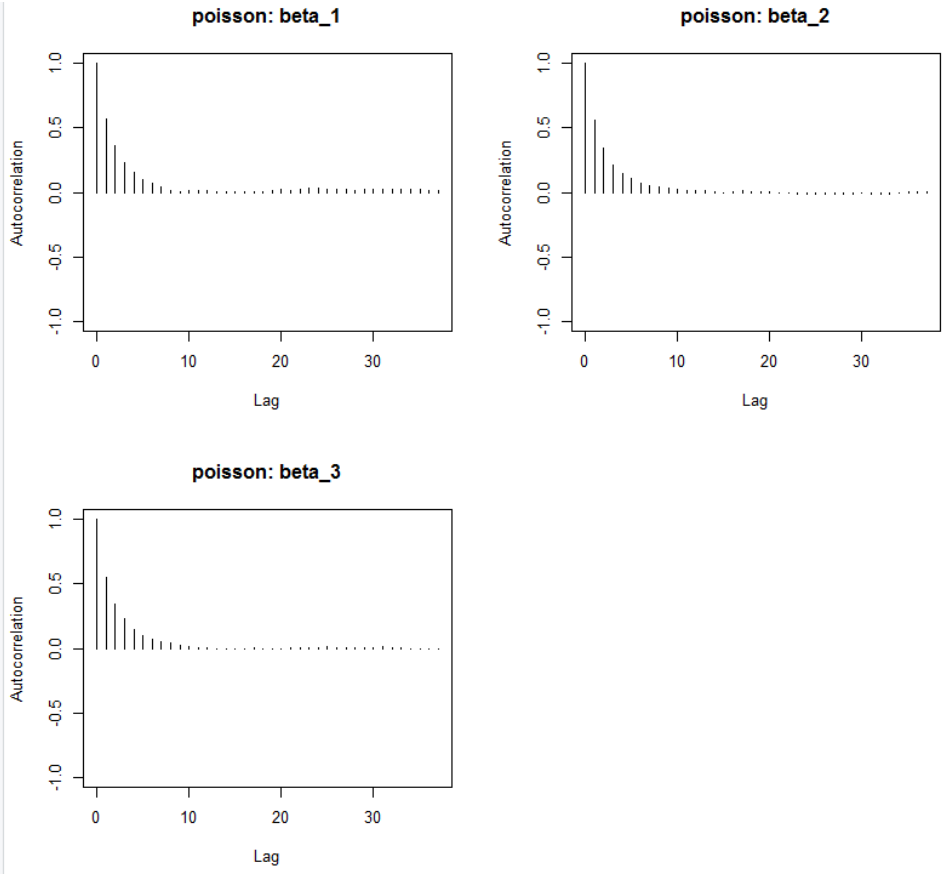

Figure 7.5: Autocorrelation of the fixed effects parameters from Poisson sub-model in the joint mixture model

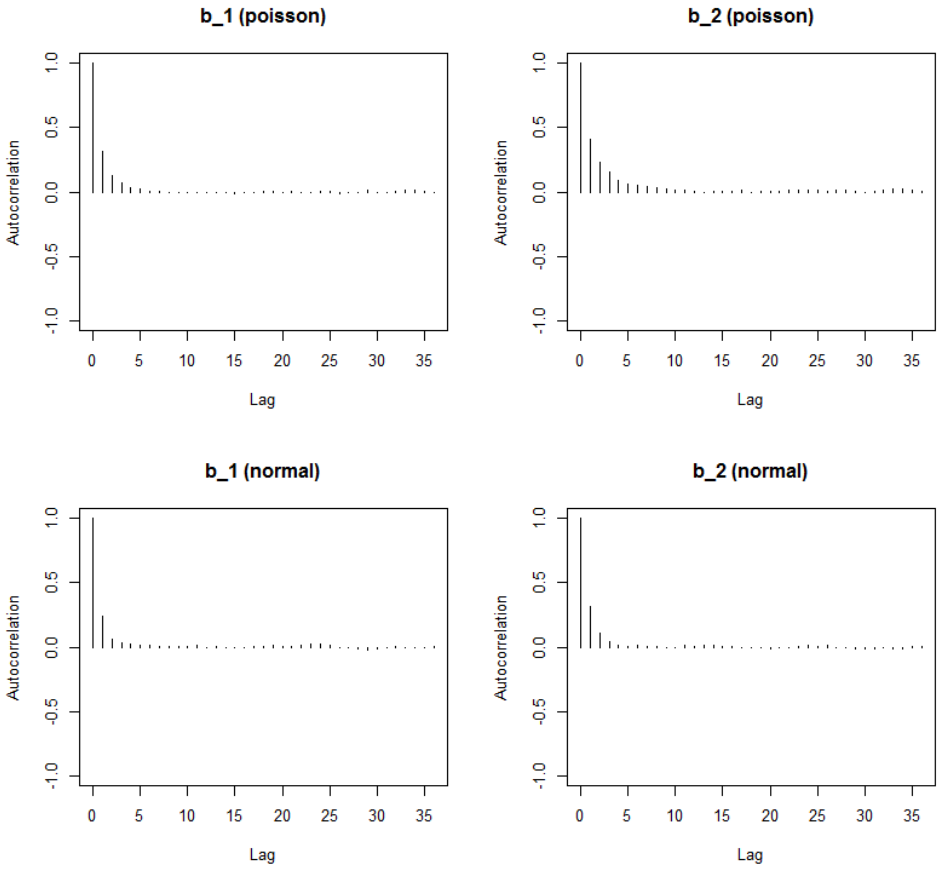

Figure 7.6: Autocorrelation of the random effects parameters in the joint mixture model

Figures 7.4, 7.5 and 7.6 shows the autocorrelation between successive values of fixed effects parameters in the normal sub-model, fixed effects parameters in the Poisson sub-model and random effects parameters in the joint model, respectively. All plots show that the correlation between successive samples drops fairly quickly, which indicates that our sampler explores the parameter space efficiently.

The method produces the point estimates of the fixed effects parameters for the Poisson outcome $(\hat{\beta}_{11}, \hat{\beta}_{12}, \hat{\beta}_{13}) = (0.8417, -0.7136, 0.3012)$, with the following 95% HPD intervals for its components: $(0.6027, 1.0801)$ for β_{11}, $(-0.8953, -0.5421)$ for β_{12}, and $(0.2362, 0.3712)$ for β_{13}. The point estimates of the fixed effects parameters for the normal outcome are $(\hat{\beta}_{21}, \hat{\beta}_{22}, \hat{\beta}_{23}) = (-0.5012, 0.5022, 0.3985)$, with 95% HPD interval $(-0.5415, -0.4623)$ for β_{21}, $(0.4864, 0.5198)$ for β_{22}, and $(0.3912, 0.4056)$ for β_{23}. We see that the 95% HPD intervals contain the true values of fixed effects parameters for both models. The same is true for the random effects parameters, where the point estimates are $(\hat{b}_{11}, \hat{b}_{21}, \hat{b}_{12}, \hat{b}_{22}) = (-0.5652, 1.1355, 0.4842, 1.0502)$ and 95% HPD intervals: $(-0.7245, -0.4168)$ for $\hat{b}_{11}$, $(1.0631, 1.2103)$ for $\hat{b}_{21}$, $(0.4443, 0.5227)$ for $\hat{b}_{12}$, and $(1.0107, 1.0898)$ for $\hat{b}_{22}$.

7.4.3 Simulation Results

We repeat the previous experiment under 12 different scenarios, where for each scenario we generate 100 replicates of data sets. The fixed effects parameters for both sub-models are the same as in the previous section, as are the random effects parameters for the Poisson sub-model. The first random effects parameter for the normal sub-model, $\mu_{b_{12}}$, takes a value of either 0.5 or 1.05, while the second parameter $\mu_{b_{22}}$

takes a value from one of the following values: $-0.5, 0.5, 0.75, 1.15, 1.65,$ and $, 2.2,$ resulting in 12 different conditions.

Tables 7.3 and 7.4 show the point estimates of fixed and random effects parameters, along with their simulation standard errors, for a joint model with both count and continuous outcomes in which the value of random effects parameters of one sub-model do not change, while the values in the other sub-model are set as follows: the value of one random effects parameter is set to $\mu_{b_{12}} = 0.5$, and the value of the other parameter $\mu_{b_{22}}$ changes as described at the beginning of this section. In all cases 95% confidence interval covers the true values of the parameters. We observe that the standard errors are quite larger for parameters in the Poisson sub-model than they are in the normal sub-model . This is because parameters in the normal sub-model are sampled directly from the posterior distribution, while parameters in the Poisson sub-model are sampled from the approximation of the posterior distribution.

Tables 7.5 and 7.6 show the estimates of parameters, and their respective simulation standard errors, when the value of one random effects parameter is $\mu_{b_{12}} = 1.05$ and the value of the second random effects parameter $\mu_{b_{22}}$ changes as described previously. We observe results that are very similar to the previous case: 95% confidence intervals cover the true value of parameters in all cases, and are narrower for parameters in the normal sub-model than they are for the parameters in the Poisson sub-model.

Table 7.3: Multivariate case: estimates of fixed and random effects parameters. Data are grouped into two clusters. Means of random effects for the Poisson sub-model are $(\mu_{b_{11}}, \mu_{b_{21}}) = (-0.5, 1.15)$. Means of random effects of the normal sub-model are $\mu_{b_{12}} = 0.5$ and $\mu_{b_{22}} = \{-0.5, 0.5, 0.75, 1.65, 2.2\}$. (simulation standard errors are shown in parentheses)

Parameter	Poisson model	Normal model
	$(\mu_{b_{12}}, \mu_{b_{22}}) = (0.5, -0.5)$	
$\hat{\beta}_1$	0.8079 (0.1450)	-0.5001 (0.0192)
$\hat{\beta}_2$	-0.5848 (0.0772)	0.5020 (0.0108)
$\hat{\beta}_3$	0.3011 (0.0343)	0.4000 (0.0049)
$\hat{b}_{1\cdot}$	-0.5344 (0.0552)	0.4996 (0.0068)
$\hat{b}_{\cdot 2}$	1.1478 (0.0385)	-0.4736 (0.0413)
	$(\mu_{b_{12}}, \mu_{b_{22}}) = (0.5, 0.5)$	
$\hat{\beta}_1$	0.7848 (0.1290)	-0.5014 (0.019)
$\hat{\beta}_2$	-0.5844 (0.0784)	0.4989 (0.0102)
$\hat{\beta}_3$	0.3003 (0.0315)	0.4005 (0.0054)
$\hat{b}_{1\cdot}$	-0.5088 (0.0749)	0.5037 (0.0053)
$\hat{b}_{\cdot 2}$	1.1493 (0.0318)	0.4966 (0.006)
	$(\mu_{b_{12}}, \mu_{b_{22}}) = (0.5, 0.75)$	
$\hat{\beta}_1$	0.8159 (0.1555)	-0.4986 (0.0211)
$\hat{\beta}_2$	-0.6042 (0.0931)	0.4993 (0.0108)
$\hat{\beta}_3$	0.3051 (0.0366)	0.4000 (0.0048)
$\hat{b}_{1\cdot}$	-0.5089 (0.0813)	0.5004 (0.0072)
$\hat{b}_{\cdot 2}$	1.1397 (0.0389)	0.7500 (0.0063)
	$(\mu_{b_{12}}, \mu_{b_{22}}) = (0.5, 1.15)$	
$\hat{\beta}_1$	0.8289 (0.1272)	-0.4983 (0.0199)
$\hat{\beta}_2$	-0.6024 (0.0781)	0.5005 (0.0103)
$\hat{\beta}_3$	0.2942 (0.0362)	0.4001 (0.0053)
$\hat{b}_{1\cdot}$	-0.5058 (0.0844)	0.4987 (0.0064)
$\hat{b}_{\cdot 2}$	1.1632 (0.0213)	1.1299 (0.0261)

Table 7.4: Multivariate case: estimates of fixed and random effects parameters. This is continuation of Table 7.3

Parameter	Poisson model	Normal model
$(\mu_{b_{12}}, \mu_{b_{22}}) = (0.5, 1.65)$		
$\hat{\beta}_1$	0.7887 (0.1390)	-0.4971 (0.0187)
$\hat{\beta}_2$	-0.605 (0.0888)	0.5003 (0.0096)
$\hat{\beta}_3$	0.3067 (0.0405)	0.4000 (0.0047)
$\hat{b}_{1\cdot}$	-0.5079 (0.0796)	0.5003 (0.0063)
$\hat{b}_{\cdot 2}$	1.1450 (0.0413)	1.6501 (0.0069)
$(\mu_{b_{12}}, \mu_{b_{22}}) = (0.5, 2.2)$		
$\hat{\beta}_1$	0.7957 (0.1243)	-0.4991 (0.0181)
$\hat{\beta}_2$	-0.6022 (0.0845)	0.4977 (0.0116)
$\hat{\beta}_3$	0.3008 (0.0344)	0.4003 (0.0047)
$\hat{b}_{1\cdot}$	-0.5111 (0.0806)	0.5000 (0.0065)
$\hat{b}_{\cdot 2}$	1.1526 (0.0379)	2.1996 (0.0072)

7.5 Conclusion

In this chapter, we have demonstrated how the GLMM-DP method can be easily extended to handle clustering longitudinal or clustered data with multivariate outcomes. We have run simulations in multivariate model with both continuous and count response, for different sets of random effects parameters. The simulation results show that the 95% confidence intervals cover true values of both fixed and random effects parameters in all cases.

A multivariate simulation may be run with many different combinations of values of fixed and random effects parameters. We have demonstrated here a very simple case where the clusters are relatively well-separated and have obtained very good results. These results, however, do not demonstrate that the GLMM-DP

Table 7.5: Multivariate case: estimates of fixed and random effects parameters. Data are grouped into two clusters. Means of random effects for the Poisson sub-model are $(\mu_{b_{11}}, \mu_{b_{21}}) = (-0.5, 1.15)$. Means of random effects of the normal sub-model are $\mu_{b_{12}} = 1.05$ and $\mu_{b_{22}} = \{-0.5, 0.5, 0.75, 1.65, 2.2\}$ (simulation standard errors are shown in parentheses)

Parameter	Poisson model	Normal model
$(\mu_{b_{12}}, \mu_{b_{22}}) = (1.05, -0.5)$		
$\hat{\beta}_1$	0.8113 (0.1364)	-0.4995 (0.0197)
$\hat{\beta}_2$	-0.5941 (0.0844)	0.5009 (0.0105)
$\hat{\beta}_3$	0.3044 (0.0371)	0.4001 (0.0049)
$\hat{b}_{1\cdot}$	-0.5377 (0.0540)	1.0497 (0.0074)
$\hat{b}_{\cdot 2}$	1.1478 (0.0390)	-0.4717 (0.0421)
$(\mu_{b_{12}}, \mu_{b_{22}}) = (1.05, 0.5)$		
$\hat{\beta}_1$	0.8022 (0.1324)	-0.5031 (0.0189)
$\hat{\beta}_2$	-0.6035 (0.0854)	0.4999 (0.0105)
$\hat{\beta}_3$	0.2983 (0.0358)	0.3997 (0.0045)
$\hat{b}_{1\cdot}$	-0.5069 (0.0724)	1.0495 (0.0063)
$\hat{b}_{\cdot 2}$	1.1504 (0.0357)	0.4994 (0.0072)
$(\mu_{b_{12}}, \mu_{b_{22}}) = (1.05, 0.75)$		
$\hat{\beta}_1$	0.7751 (0.1309)	-0.4984 (0.0195)
$\hat{\beta}_2$	-0.606 (0.0782)	0.5002 (0.0102)
$\hat{\beta}_3$	0.3026 (0.0364)	0.4005 (0.0044)
$\hat{b}_{1\cdot}$	-0.5232 (0.0858)	1.0491 (0.0065)
$\hat{b}_{\cdot 2}$	1.1504 (0.0384)	0.7482 (0.0067)
$(\mu_{b_{12}}, \mu_{b_{22}}) = (1.05, 1.15)$		
$\hat{\beta}_1$	0.8141 (0.1328)	-0.5000 (0.0185)
$\hat{\beta}_2$	-0.5877 (0.0825)	0.4991 (0.0110)
$\hat{\beta}_3$	0.3055 (0.0347)	0.4002 (0.0051)
$\hat{b}_{1\cdot}$	-0.5090 (0.0831)	1.0503 (0.0066)
$\hat{b}_{\cdot 2}$	1.1656 (0.0249)	1.1298 (0.0253)

Table 7.6: Multivariate case: estimates of fixed and random effects parameters. This table is continuation of Table 7.5.

Parameter	Poisson model	Normal model
$(\mu_{b_{12}}, \mu_{b_{22}}) = (1.05, 1.65)$		
$\hat{\beta}_1$	0.7777 (0.1213)	-0.4987 (0.0178)
$\hat{\beta}_2$	-0.5923 (0.0845)	0.5002 (0.0107)
$\hat{\beta}_3$	0.2973 (0.0319)	0.4011 (0.0047)
$\hat{b}_{1\cdot}$	-0.5210 (0.0851)	1.0502 (0.0062)
$\hat{b}_{\cdot 2}$	1.1521 (0.0339)	1.6499 (0.0066)
$(\mu_{b_{12}}, \mu_{b_{22}}) = (1.05, 2.2)$		
$\hat{\beta}_1$	0.778 (0.1474)	-0.4983 (0.0175)
$\hat{\beta}_2$	-0.5938 (0.0793)	0.5007 (0.0121)
$\hat{\beta}_3$	0.2996 (0.0361)	0.4 (0.0047)
$\hat{b}_{1\cdot}$	-0.5094 (0.0768)	1.0514 (0.0067)
$\hat{b}_{\cdot 2}$	1.1449 (0.0384)	2.1994 (0.0069)

method would always produce such results in all scenarios. We have seen in Chapter 5 that when there is significant overlap between two clusters, the GLMM-DP method may fail to recover the correct number of clusters, and with it, valid parameter estimates. We expect the extension of the GLMM-DP method in multivariate case to exhibit similar characteristics. However, as pointed out in Chapter 5, these shortcomings may be mitigated easily in some cases.

Chapter 8

Conclusion and Future Research

In this thesis, we have proposed a novel method, Generalized linear mixed models clustering using Dirichlet Process (GLMM-DP) which facilitates simultaneous clustering of longitudinal, or more generally clustered, data and estimation of parameters of the underlying model. The data are clustered based on grouping of random effects parameters in the generalized linear mixed model.

We have tested our method on simulated data sets in which the response variable is either continuous or count, with varying number of observations recorded on an individual and with different values of random effects parameters. Our results show that the method performs well in terms of being able to recover the true number of clusters, in terms of clustering profiles correctly, and in terms of estimating model parameters. Existing methods designed to address the same problem domain (simultaneous clustering of profiles and parameter estimation) are able to handle only continuous outcomes, while the GLMM-DP is the first method of its kind which does not assume that the number of clusters is known in advance and which is able to handle outcomes of any distribution in the exponential family of distributions..

We have also applied the GLMM-DP method on a real data set from a public health survey domain. We have demonstrated that the GLMM-DP method obtains parameter estimates that are very similar to those obtained by a classical maximum likelihood method. However, on top of providing comparable parameter estimates, the GLMM-DP method also identifies three clusters of health regions that, when the maximum likelihood method is applied on these clusters, we are able to obtain an insight from the data that is not otherwise available using other methods.

We have also extended the GLMM-DP method to handle profiles with multiple outcomes, and have evaluated its performance using simulated data.

There are several extensions that could be considered on the GLMM-DP method. First, the focus of our current work has been estimation: we want to simultaneously estimate model parameters and cluster data so that we can better explain, in the case of public health data, what affects doctor's visits and how. The GLMM-DP method could be extended to be better suitable for prediction. For example, instead of developing a probability model for the response variable only, one could extend it to include the probability model for covariates as well. That way, given covariates and response of a new individual, one may be able to identify the cluster to which it belongs, and with it, characteristics of the new unit. Similar work with cross-sectional data was done in [112] and [123]. Note that if the GLMM-DP method clustered profiles based on values of response and covariates, then this extension would be expected to work very well. Instead, we cluster data based on model parameters. However, model parameters are determined by data (response and covariates) and the assumed model, so units that have similar covariates and response would be expected to have also similar parameter values as well.

Second, the method groups profiles based on them sharing the same value of random effects parameter. A more flexible approach would be to cluster profiles based on their 'similarity' of random effects parameters, where profiles allocated to the same cluster would not share the same random effects. The GLMM-DP method can be easily extended to accommodate this requirement: instead of placing Dirichlet Process prior on random effects parameters, one may place Dirichlet Process prior on the mean of normal distribution from which the random effects are drawn. It would be interesting to see if this would provide different insights with the CCHS survey data.

Third, it is well established that a label switching is a challenging task in mixture models using Bayesian statistics. We have used the approach proposed by Molitor [112] and it seems to perform well. It first identifies the "best" clustering and then estimates parameters based on the chosen clustering. However, when looking for the "best" clustering, it uses only binary similarity matrix, but we know that the profiles are allocated to clusters with certain probabilities. One could investigate if considering more information in identifying the "best" partitioning (such as probabilities) would improve the estimates of component-based parameters, especially when the clusters are not well separated.

Fourth, sampling in high-dimensional spaces is a difficult task, and significant improvements to the GLMM-DP method might be possible by using more advanced sampling algorithms for both fixed and random effects parameters. For random effects parameters, one may consider the slice sampling [77], or a variation of it.

Finally, implementation the GLMM-DP method in C or C++ [124] would not only improve the current performance but also allow one to test the method on even more complex models and larger data sets.

List of References

[1] N. Balakrishnan and V. B. Nevzorov, *A Primer on Statistical Distributions*. Wiley-Interscience, 2003.

[2] G. M. Fitzmaurice, N. M. v, and J. H. Ware, *Applied Longitudinal Analysis*, vol. 998. John Wiley & Sons, 2012.

[3] J. B. MacQueen, "Some methods for classification and analysis of multivariate observations," *Proceedings of 5th Berkeley Simposium on Mathematical Staistics and Probability. University of California Press*, pp. 281–297, 1967.

[4] L. Kaufman and P. Rousseeuw, *Clustering by Means of Medoids*. North-Holland, 1987.

[5] C. Genolini, X. Alacoque, M. Sentenac, and C. Arnaud, "kml and kml3d: R packages to cluster longitudinal data," *Journal of Statistical Software*, vol. 65, 2015.

[6] C. Genolini, R. Ecochard, M. Benghezal, T. Driss, S. Andrieu, and F. Subtil, "kmlshape: An efficient method to cluster longitudinal data (time-series) according to their shapes," *PLOS ONE*, vol. 11, pp. 1–24, 06 2016.

[7] G. McLachlan and D. Peel, *Finite Mixture Models*. Wiley-Interscience, 1st eition ed., 2000.

[8] M. Pourahmadi, "Joint mean-covariance models with applications to longitudinal data: unconstrained parameterisation," *Biometrika*, vol. 86, pp. 677–690, Sept. 1999.

[9] P. D. McNicholas and T. B. Murphy, "Model-based clusterig of longitudinal data," *The Canadian Journal of Statistics*, vol. 38, no. 1, pp. 153–168, 2010.

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

[11] J. D. Banfield and A. E. Raftery, "Model-based gaussian and non-gaussian clustering," *Biometrics*, vol. 49, pp. 803–821, 1993.

[12] Y. He, *Bayesian Cluster Analysis with Longitudinal Data*. PhD thesis, UC Irvine, 2014.

[13] M. Kyung, "Dirichlet process mixtures of linear mixed regressions," *Communications for Statistical Applications and Methods*, vol. 22, no. 6, pp. 625–637, 2015.

[14] J. Sun, J. Herazo-Maya, N. Kaminski, H. Zhao, and J. Warren, "A dirichlet process mixture model for clustering longitudinal gene expression data," *Statistics in Medicine*, 09 2016.

[15] Y. W. Teh, "Dirichlet process," in *Encyclopedia of Machine Learning*, pp. 280–287, Springer, 2011.

[16] R. C. Littell, J. Pendergast, and R. Natarajan, "Modelling covariance structure in the analysis of repeated measures data," *Statistics in Medicine*, vol. 19, no. 13, pp. 793–819, 2000.

[17] H. D. Patterson and R. Thompson, "Recovery of inter-block information when block sizes are unequal," *Biometrika*, vol. 58, no. 3, pp. 545–554, 1971.

[18] G. Casella and R. L. Berger, *Statistical Inference*, vol. 2. Duxbury Pacific Grove, CA, 2002.

[19] J. A. Nelder and R. W. M. Wedderburn, "Generalized linear models," *Journal of the Royal Statistical Society, Series A*, vol. 135, no. 3, pp. 370–384, 1973.

[20] R. W. M. Wedderburn, "Quasi-likelihood functions, generalized linear models, and the gauss-newton method," *Biometrika*, vol. 61, pp. 439–447, Dec. 1974.

[21] S. R. Searle, G. Casella, and C. E. McCulloch, *Variance Components*, vol. 391. John Wiley & Sons, 2009.

[22] C. E. McCulloch, S. R. Searle, and J. M. Neuhaus, *Generalized Linear and Mixed Models*. Wiley Series in Probability and Statistics, New Jersey: Wiley, second ed., 2008.

[23] D. A. Harville, "Maximum likelihood approaches to variance component estimation and to related problems," *Journal of the American Statistical Association*, vol. 72, no. 358, pp. 320–338, 1977.

[24] V. Geert and M. Geert, *Linear Mixed Models for Longitudinal Data*. Springer, 2000.

[25] N. Lange and L. Ryan, "Assessing normality in random effects models," *The Annals of Statistics*, pp. 624–642, 1989.

[26] H. O. Hartley and J. N. K. Rao, "Maximum-likelihood estimation for the mixed analysis of variance model," *Biometrika*, vol. 54, no. 1-2, pp. 93–108, 1967.

[27] J. Jiang, "Reml estimation: Asymptotic behavior and related topics," *The Annals of Statistics*, vol. 24, no. 1, pp. 255–286, 1996.

[28] W. A. T. Jr, "The problem of negative estimates of variance components," *The Annals of Mathematical Statistics*, pp. 273–289, 1962.

[29] E. A. C. Crouch and D. Spiegelman, "The evaluation of integrals of the form+- f (t) exp (- t 2) dt: Application to logistic-normal models," *Journal of the American Statistical Association*, vol. 85, no. 410, pp. 464–469, 1990.

[30] C. E. McCulloch, "Maximum likelihood algorithms for generalized linear mixed models," *Journal of the American Statistical Association*, vol. 92, no. 437, pp. 162–170, 1997.

[31] 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*, pp. 1–38, 1977.

[32] P. X.-K. Song, Y. Fan, and J. D. Kalbfleisch, "Maximization by parts in likelihood inference," *Journal of the American Statistical Association*, vol. 100, no. 472, pp. 1145–1158, 2005.

[33] N. E. Breslow and D. G. Clayton, "Approximate inference in generalized linear mixed models," *Journal of the American Statistical Association*, vol. 88, no. 421, pp. 9–25, 1993.

[34] P. S. Laplace, "Memoir on the probability of the causes of events," *Statistical Science*, vol. 1, pp. 364–378, 08 1986.

[35] A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin, *Bayesian Data Analysis*. CRC press, 2013.

[36] H. Jeffreys, *Theory of Probability*. Oxford, England: Oxford, third ed., 1961.

[37] G. E. P. Box and G. C. Tiao, *Bayesian Inference in Statistical Analysis*. Wiley-Interscience, 1 ed., 1992.

[38] M.-H. Chen and J. G. Ibrahim, "Power prior distributions for regression models," *Statistical Science*, vol. 15, pp. 46–60, 02 2000.

[39] R. Willink and I. Lira, "A united interpretation of different uncertainty intervals," *Measurement*, vol. 38, no. 1, pp. 61 – 66, 2005.

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

[41] N. Choudhuri, S. Ghosal, and A. Roy, "Bayesian methods for function estimation," *Handbook of statistics*, vol. 25, pp. 373–414, 2005.

[42] Z. Ghahramani, "Bayesian non-parametrics and the probabilistic approach to modelling," *Philosophical Transactions A, The Royal Society Publishing*, vol. 371, no. 1984, p. 20110553, 2013.

[43] F. Caron and E. B. Fox, "Bayesian nonparametric models of sparse and exchangeable random graphs," in *NIPS Workshop on Frontiers in Network Analysis*, Citeseer, 2014.

[44] C. Pearson, "Contributions to the mathematical theory of evolution. ii. skew variation in homogeneous material," *Philosophical Transactions of the Royal Society of London*, vol. 186, no. Part I, pp. 343–424, 1895.

[45] T. Leonard, "A bayesian method for histograms," *Biometrika*, vol. 60, no. 2, pp. 297–308, 1973.

[46] A. W. van der Vaart, *Asymptotic statistics*. Cambridge Series in Statistical and Probabilistic Mathematics, Cambridge University Press, 1998.

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

[48] T. Tao, *An Introduction to Measure Theory*, vol. 126. American Mathematical Society, 2011.

[49] C. E. Rasmussen and C. K. I. Williams, "Gaussian processes for machine learning," *The MIT Press, Cambridge, MA, USA*, vol. 38, pp. 715–719, 2006.

[50] E. B. Sudderth, *Graphical Models for Visual Object Recognition and Tracking*. PhD thesis, Massachusetts Institute of Technology, 2006.

[51] J. Sethuraman, "A constructive definition of dirichlet priors," *Statistica Sinica*, pp. 639–650, 1994.

[52] D. Blackwell and J. B. MacQueen, "Ferguson distributions via polya urn schemes," *The Annals of Statistics*, p. 3, 1973.

[53] J. Pitman, "Combinatorial stochastic processes. lectures from the 32nd summer school on probability theory held in saint-flour, july 7–24, 2002. with a foreword by jean picard," *Lecture Notes in Mathematics*, vol. 1875, 1875.

[54] A. Y. Lo, "On a class of bayesian nonparametric estimates: I. density estimates," *The Annals of Statistics*, pp. 351–357, 1984.

[55] C. P. Robert and G. Casella, *Monte Carlo Statistical Methods*. Springer, 2005.

[56] D. Gamerman and H. F. Lopes, *Markov Chain Monte Carlo: stochastic simulation for Bayesian inference*. Chapman and Hall/CRC, 2006.

[57] G. Casella and E. I. George, "Explaining the gibbs sampler," *The American Statistician*, vol. 46, no. 3, pp. 167–174, 1992.

[58] J. Liu, *Correlation Structure and Convergence Rate of the Gibbs Sampler*. PhD thesis, University of Chicago, Department of Statistics, 1991.

[59] G. O. Roberts and S. K. Sahu, "Updating schemes, correlation structure, blocking and parameterization for the gibbs sampler," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 59, no. 2, pp. 291–317, 1997.

[60] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *The Journal of Chemical Physics*, vol. 21, no. 6, pp. 1087–1092, 1953.

[61] K. W. Hastings, "Monte carlo sampling methods using markov chains and their applications," *Biometrika*, vol. 57, no. 1, pp. 97–109, 1970.

[62] L. Tierney, "Markov chains for exploring posterior distributions," *The Annals of Statistics*, pp. 1701–1728, 1994.

[63] J. E. Bennett and A. Racine-Poon, "Mcmc for nonlinear hierarchical models," *Markov Chain Monte Carlo Methods in Practice*, pp. 339–357, 1996.

[64] J. Besag, P. Green, D. Higdon, and K. Mengersen, "Bayesian computation and stochastic systems," *Statistical Science*, pp. 3–41, 1995.

[65] P. Müller, "Metropolis based posterior integration schemes," in *Numerical Recipes in Fortran (2nd Edition)*, Citeseer, 1994.

[66] J. Geweke, "Evaluating the accuracy of sampling-based approaches to the calculation of posterior moments. bayesian statistics 4. edited by: Bernardo jm, berger j, dawid ap, smith afm. 1992."

[67] S. Chib and E. Greenberg, "Bayes inference in regression models with arma (p, q) errors," *Journal of Econometrics*, vol. 64, no. 1-2, pp. 183–206, 1994.

[68] A. S. Mahani, A. Hasan, M. Jiang, and M. T. A. Sharabiani, "sns: Stochastic newton sampler. r package version 0.9," 2014.

[69] M. D. Escobar, "Estimating normal means with a dirichlet process prior," *Journal of the American Statistical Association*, vol. 89, no. 425, pp. 268–277, 1994.

[70] M. D. Escobar and M. West, "Bayesian density estimation and inference using mixtures," *Journal of the American Statistical Association*, vol. 90, no. 430, pp. 577–588, 1995.

[71] C. A. Bush and S. N. MacEachern, "A semiparametric bayesian model for randomised block designs," *Biometrika*, vol. 83, no. 2, pp. 275–285, 1996.

[72] M. West, P. Müller, and M. D. Escobar, "Hierarchical priors and mixture models with applications in regression and density estimation," *Aspects of Uncertainty: A Tribute to D. V. Lindley*, pp. 363–386, 1994.

[73] R. M. Neal, "Markov chain sampling methods for dirichlet process mixture models," *Journal of Computational and Graphical Statistics*, vol. 9, no. 2, pp. 249–265, 2000.

[74] S. Jain and R. Neal, "A split-merge markov chain monte carlo procedure for the dirichlet process mixture model," *Journal of Computational and Graphical Statistics*, vol. 13, pp. 158–182, 2000.

[75] S. Jain and R. M. Neal, "Splitting and merging components of a nonconjugate dirichlet process mixture model," *Bayesian Analysis*, vol. 2, pp. 445–472, 09 2007.

[76] S. G. Walker, "Sampling the dirichlet mixture model with slices," *Communications in Statistics - Simulation and Computation*, vol. 36, no. 1, pp. 45–54, 2007.

[77] M. Kalli, J. E. Griffin, and S. G. Walker, "Slice sampling mixture models," *Statistics and Computing*, vol. 21, no. 1, pp. 93–105, 2011.

[78] O. Papaspiliopoulos and G. Roberts, "Retrospective markov chain monte carlo methods for dirichlet process hierarchical models," *Biometrika*, vol. 95, no. 1, pp. 169–186, 2008.

[79] H. Ishwaran and M. Zarepour, "Markov chain monte carlo in approximate dirichlet and beta two-parameter process hierarchical models," *Biometrika*, vol. 87, no. 2, pp. 371–390, 2000.

[80] H. Ishwaran and L. F. James, "Gibbs sampling methods for stick-breaking priors," *Journal of the American Statistical Association*, vol. 96, no. 453, pp. 161–173, 2001.

[81] B. S. Everitt, S. Landau, M. Leese, and D. Stahl, *Cluster Analysis*. John Wiley & Sons, Inc., 2011.

[82] D. Xu and Y. Tian, "A comprehensive survey of clustering algorithms," *Annals of Data Science*, vol. 2, pp. 165–193, Jun 2015.

[83] C. Rae, *Statistics in market research*. Oxford University Press, 2004.

[84] G. J. Babu and E. D. Feigelson, *Statistical Challenges in Modern Astronomy II*. Springer Science & Business Media, 2012.

[85] R. Xu and D. C. Wunsch, "Clustering algorithms in biomedical research: a review," *IEEE Reviews in Biomedical Engineering*, vol. 3, pp. 120–154, 2010.

[86] A. Alsayat and H. El-Sayed, "Social media analysis using optimized k-means clustering," in *Software Engineering Research, Management and Applications (SERA)*, pp. 61–66, IEEE, 2016.

[87] C. Fraley and A. E. Raftery, "How many clusters? which clustering method? answers via model-based cluster analysis," *The computer journal*, vol. 41, no. 8, pp. 578–588, 1998.

[88] N. A. Heard, C. C. Holmes, and D. A. Stephens, "A quantitative study of gene regulation involved in the immune response of anopheline mosquitoes: An application of bayesian hierarchical clustering of curves," *Journal of the American Statistical Association*, vol. 101, no. 473, pp. 18–29, 2006.

[89] P. D. McNicholas, K. R. Jampani, and S. Subedi, *longclust: Model-Based Clustering and Classification for Longitudinal Data*, 2018. R package version 1.2.2.

[90] M. Shaikh, P. McNicholas, and A. F. Desmond, "A pseudo-em algorithm for clustering incomplete longitudinal data.," *The International Journal of Biostatistics*, vol. 6, no. 1, 2010.

[91] J. L. Andrews and P. D. McNicholas, "Model-based clustering, classification, and discriminant analysis via mixtures of multivariate t-distributions," *Statistics and Computing*, vol. 22, pp. 1021–1029, Sept. 2012.

[92] P. D. McNicholas and S. Subedi, "Clustering gene expression time course data using mixtures of multivariate t-distributions," *Journal of Statistical Planning and Inference*, vol. 142, no. 5, pp. 1114–1127, 2012.

[93] A. Ciampi, H. Campbell, A. Dyachenko, B. Rich, J. McCusker, and M. G. Cole, "Model-based clustering of longitudinal data: Application to modeling disease course and gene expression trajectories," *Communications in Statistics - Simulation and Computation*, vol. 41, no. 7, pp. 992–1005, 2012.

[94] S. J. Gaffney, "Curve clustering with random effects regression mixtures," in *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*, 2003.

[95] G. Celeux, C. Lavergne, and O. C. Martin, "Mixture of linear mixed models for clustering gene expression profiles from repeated microarray experiments," *Statistical Modelling*, vol. 5, pp. 243–267, Nov. 2005.

[96] R. D. la Cruz-Mesa, F. A. Quintana, and G. Marshall, "Model-based clustering for longitudinal data," *Computational Statistics & Data Analysis*, vol. 52, no. 3, pp. 1441–1457, 2008.

[97] G. Schwarz, "Estimating the dimension of a model," *The Annals of Statistics*, vol. 6, no. 2, pp. 461–164, 1978.

[98] H. Akaike, "Information theory and an extension of the maximum likelihood principle," in *Second International Symposium on Information Theory* (B. N. Petrov and F. Csaki, eds.), (Budapest), pp. 267–281, Akadémiai Kiado, 1973.

[99] F. Heinzl and G. Tutz, "Clustering in linear mixed models with approximate dirichlet process mixtures using em algorithm," 2013.

[100] A. Komarek and L. Komárková, "Clustering for multivariate continuous and discrete longitudinal data," *The Annals of Applied Statistics*, pp. 177–200, 2013.

[101] I. G. G. Kreft, I. Kreft, and J. de Leeuw, *Introducing multilevel modeling*. Sage, 1998.

[102] R. A. Christensen, *Plane Answers to Complex Questions: The Theory of Linear Models*. Berlin, Heidelberg: Springer-Verlag, 1987.

[103] M. Davidian and D. M. Giltinan, "Nonlinear models for repeated measurement data: an overview and update," *Journal of Agricultural, Biological and Environmental Statistics*, vol. 8, 2003.

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

[105] M. West, *Hyperparameter Estimation in Dirichlet Process Mixture Models*. Duke University ISDS Discussion Paper# 92-A03, 1992.

[106] M. Sperrin, T. Jaki, and E. Wit, "Probabilistic relabelling strategies for the label switching problem in bayesian mixture models," *Statistics and Computing*, vol. 20, no. 3, pp. 357–366, 2010.

[107] J. Geweke, "Interpretation and inference in mixture models: Simple mcmc works," *Computational Statistics & Data Analysis*, vol. 51, no. 7, pp. 3529–3550, 2007.

[108] M. Stephens, "Dealing with label-switching in mixture models," *Journal of the Royan Statistical Society*, no. 62, pp. 795–8009, 2000.

[109] A. Nobile and A. T. Fearnside, "Bayesian finite mixtures with an unknown number of components: the allocation sampler," *Statistics and Computing*, vol. 17, no. 2, pp. 147–162, 2007.

[110] D. I. Hastie, S. Liverani, and S. Richardson, "Sampling from dirichlet process mixture models with unknown concentration parameter: mixing issues in large data implementations," *Statistics and computing*, vol. 25, no. 5, pp. 1023–1037, 2015.

[111] M. Medvedovic and S. Sivaganesan, "Bayesian infinite mixture model based clustering of gene expression profiles," *Bioinformatics*, vol. 18, no. 9, pp. 1194–1206, 2002.

[112] J. Molitor, M. Papathomas, M. Jerrett, and S. Richardson, "Bayesian profile regression with an application to the national survey of children's health," *Biostatistics*, vol. 11, no. 3, pp. 484–498, 2010.

[113] M. Plummer, N. Best, K. Cowles, and K. Vines, "Coda: Convergence diagnosis and output analysis for mcmc," *R News*, vol. 6, no. 1, pp. 7–11, 2006.

[114] D. I. Hastie, S. Liverani, L. Azizi, S. Richardson, and I. Stcker, "A semi-parametric approach to estimate risk functions associated with multi-dimensional exposure profiles: application to smoking and lung cancer," *BMC Medical Research Methodology*, vol. 13, no. 1, p. 1, 2013.

[115] S. Canada, "Canadian community health survey," 2013.

[116] G. Bronstrom, "glmmml: Generalized linear models with clustering. r package version 1.0," 2013.

[117] S. Canada, "Census subdivision (csd)." http://www12.statcan.gc.ca/census-recensement/2011/ref/dict/geo012-eng.cfm, Sept. 2015.

[118] R. J. A. Little and D. B. Rubin, *Statistical Analysis with Missing Data*, vol. 333. John Wiley & Sons, 2014.

[119] D. X., E. T., Y.-H. O., and L. H., "A joint finite mixture model for clustering genes from independent gaussian and beta distributed data," *BMC Bioinformatics*, vol. 10, May 2009.

[120] L. Villarroel, G. Marshall, and A. Baron, "Cluster analysis using multivariate mixed effects models," *Statistics in Medicine*, vol. 28, pp. 2552–65, 09 2009.

[121] L.-X. Qin and S. G. Self, "The clustering of regression models method with applications in gene expression data," *Biometrics*, vol. 62, no. 2, pp. 526–533, 2006.

[122] R. Gueorguieva, "A multivariate generalized linear mixed model for joint modeling of clustered outcomes in the exponential family," *Statistical Modelling*, vol. 1, pp. 177–193, 10 2001.

[123] D. I. Hastie, S. Liverani, L. Azizi, S. Richardson, and I. Stcker, "A semi-parametric approach to estimate risk functions associated with multi-dimensional exposure profiles: application to smoking and lung cancer," *BMC Medical Research Methodology*, vol. 13, p. 129, Oct 2013.

[124] B. Stroustrup, *The C++ Programming Language*. Addison-Wesley Professional, 4th ed., 2013.

[125] D. Markwick and G. J. Ross, *dirichletprocess: Build Dirichlet Process Objects for Bayesian Modelling*, 2018. R package version 0.2.0.

Appendix A

R Code for GLMM-DP

This Appendix contains the implementation code of the GLMM-DP method. The code was initially based on the dirichletprocess package [125]. It still has similar structure (breakdown of the process into functions), and borrows few functions from it: the function to update concentration parameter, the function to update cluster labels (as new clusters are added or existing clusters removed), and the high-level (utility) function that brings all the functions together. The core statistical methods have been completely rewritten.

The following code is the code that was used in multivariate case. The code for analysis of models with univariate response is not included - the attached code can be easily modified to handle univariate case.

A.1 Simulating data

```
#####################################################################
# Simulate longitudinal data with a response of Poisson distribution.
#####################################################################
get_poisson_data <- function(units=60, obs.per.unit=3,
                             beta=c(-1.5, 0.5, -1), bi=list(c(0.2)))
{
     #total number of observations
```

```
N <- units * obs.per.unit

#times of ovservations: 1, 2, 3, ..., obs.per.unit
times = seq(obs.per.unit)

#standardize the observation times
times = (times - mean(times))/sd(times)

#times vector (duplicated ny number of observations per unit)
all.times <- rep(times, units)

#value of the first predictor at time = 0
x1.base = rnorm(units, mean = 0.1, sd=0.5)

#value of the second predictor at time = 0
x2.base = rnorm(units, mean = 0.9, sd=0.5)

#vector of the first predictor for all units
first = NULL
#vector of the second predictor of all units
second = NULL

for(i in 1:length(x1.base)){

    #current value of the first predictor
    x1.current = x1.base[i]
    #current value of the second predictor
    x2.current = x2.base[i]

    for(j in 1:obs.per.unit){

        #next value of the first predictor
        x1.next = x1.current*0.78 + rnorm(1, 0, 0.1)
        #next value of the second predictor
        x2.next = x2.current*(-0.78) + rnorm(1, 0, 0.1)

        #add next value of the current predictors
        first = rbind(first, x1.next)
        second = rbind(second, x2.next)

        #remember current values of the predictors
        x1.current = x1.next
        x2.current = x2.next
    }
}
```

```
#the complete matrix of fixed effects covariates
x.matrix <- cbind(first,second, all.times)

#re-shuffle the rows of matrix X (the reason for this is that
#one random effect is added to the first half (or first portion)
#of the data, and the second random effect is added to the second
#half) - reshufling the rows prevents the non-identifiability problem.
kk = sample(1:units)
x.mat.temp = NULL
for(i in 1:length(kk)){
    x.mat.temp = rbind(x.mat.temp, x.matrix[ ((kk[i] - 1)*obs.per.unit +
                      1):(kk[i]*obs.per.unit),])
}
x.matrix=x.mat.temp

#number of clusters (is the same as the number of random effects)
clusters.count = length(bi)

#number of units per cluster
clusters.size = units / clusters.count

#random effects (one per unit)
all.rand.effects = NULL
for(i in 1:clusters.count){
    random.effects = rnorm(clusters.size, mean = bi[[i]], sd=0.01)
    all.rand.effects = c(all.rand.effects,random.effects)
}

#random effects (one per observation)
all.rand.effects = rep(all.rand.effects, each=obs.per.unit)

#this is Z*b_i (with Z being a vector of all 1's)
z.matrix <- matrix(all.rand.effects, ncol=1)#cbind(uu0, uu1)

#linear predictor
eta <- c(x.matrix %*% beta) + z.matrix

#average of the Poisson model
mu <- exp(eta)

#response
y <- rpois(N, mu)

#response in a matrix form
```

```
     y = matrix(y, ncol=1)

     return (list(Y=y, X=x.matrix, Z=matrix(1, ncol=1, nrow=N),
     t=all.times, t0=times))
}

#####################################################################
# Expands the data to include normal submodel.  The two models share
# the same fixed effects covariates (but have different parameters).
#####################################################################
add_normal_response <- function(X, Y, Z, obs.per.unit = 3,
                           beta=c(-1.5, 0.5, -1), bi=list(c(0.2)) ){

     #number of clusters
     clusters.count = length(bi)

     #number of units per cluster
     clusters.size = nrow(X) / (clusters.count * obs.per.unit)

     #random effects (one per unit)
     all.rand.effects = NULL
     for(i in 1:clusters.count){
          random.effects = rnorm(clusters.size, mean = bi[[i]], sd=0.01)
          all.rand.effects = c(all.rand.effects,random.effects)
     }

     #random effects (one per observation)
     all.rand.effects = rep(all.rand.effects, each=obs.per.unit)

     #this is Z*b_i (with Z being a vector with all 1's)
     z.matrix <- matrix(all.rand.effects, ncol=1)#cbind(uu0, uu1)

     #linear predictor of the normal model
     eta <- X %*% beta + z.matrix

     #response of the normal model
     Y2 = rnorm(nrow(X), mean = eta, sd = 0.1)

     #response in a matrix form
     Y2 = matrix(Y2, ncol=1)

     return (list(Y = list(Y, Y2), #response of both models
     X = list(X, X),  #the two models share the same matrix X
     Z = list(Z, Z))) #the two models share the same matrix Z
}
```

A.2 Approximating posterior distributions

```
library(tools)
library(coda)
library(magic)

#####################################################################
# Draws n samples from the prior of random effects distribution.
#####################################################################
PriorDraw <- function(mdobj, n=1){
     #the list to hold the result
     theta <- list()

     #draw a sample from the prior distribution of fixed effects params
     betas = mvtnorm::rmvnorm(n, mean = mdobj$biPriors$mu,
     sigma = mdobj$biPriors$sigma)

     #format output as 3-dimensional array
     theta[[1]]=array(t(betas), dim =c(1,length(mdobj$biPriors$mu), n))

     return(theta)
}

#####################################################################
# Calculates the likelihood given the observations from the normal
# model.
#####################################################################
Likelihood.normal <- function(mdobj, Y, X, Z, beta, bi){
     if(is.null(dim(bi))){
          eta1 = X%*%beta
          eta2 = Z%*%bi
          if(ncol(eta2)>ncol(eta1)){
               eta1 = matrix(rep(eta1,each=ncol(eta2)),
               ncol=ncol(eta2), byrow=TRUE)
          }
          eta = eta1 + eta2
     }else{
          if(is.null(beta) || is.null(X)){
               eta1 = 0
          }else{
               eta1 = X%*%beta
          }
          eta2 = Z%*%bi
          eta = eta1 + eta2
     }
     kk = log(2*mdobj$phi * pi)/2 - Y*Y/(2*mdobj$phi)
```

```
     temp = (t(Y)%*%eta - colSums(eta*eta/2))/mdobj$phi + sum(kk)

     return (as.numeric(exp(temp)))
}

#####################################################################
# Calculates the likelihood of the Poisson model.
#####################################################################
Likelihood.poisson <- function(mdobj, Y, X, Z, beta, bi){
     if(is.null(dim(bi))){
          eta1 = X%*%beta
          eta2 = Z%*%bi
          if(ncol(eta2)>ncol(eta1)){
               eta1 = matrix(rep(eta1,each=ncol(eta2)),
               ncol=ncol(eta2), byrow=TRUE)
          }
          eta = eta1 + eta2
     }else{
          if(is.null(beta) || is.null(X)){
               eta1 = 0
          }else{
               eta1 = X%*%beta
          }
          eta2 = Z%*%bi#[[1]][,,]
          eta = eta1 + eta2
     }
     temp = t(Y)%*%eta - colSums(exp(eta)) - sum(lfactorial(Y))

     return (as.numeric(exp(temp)))
}

#####################################################################
#Calculates the likelihood of the Bernoulli model. (not tested)
#####################################################################
Likelihood.bernoulli <- function(mdobj, Y, X, Z, beta, bi){
     if(is.null(dim(bi)) ){
          if(length(bi)==1){
               eta = X%*%beta + Z%*%bi#[[1]][,,] #+ Z*bi
          }else{
               eta1 = X%*%beta
               eta2 = Z%*%bi#[[1]][,,]
               eta = eta2
               for(i in 1:ncol(eta)){
                    eta[,i] = eta[,i] + eta1
               }
```

```
          }
     }
     temp = t(Y)%*%eta - colSums(log(1+exp(eta)))
     return (as.numeric(exp(temp)))
}

#######################################################################
# Calculates the log-likelihood given the data from the Bernoulli
# model. (not tested)
#######################################################################
logLikelihood.bernoulli <- function(mdobj, Y, X, Z, beta, bi){
     #linear predictor
     eta = X%*%beta + Z%*%bi
     #log-likelihood
     temp = Y*eta - log(1 + exp(eta))
     return (as.numeric(temp))
}

#######################################################################
# Calculates the likelihood given the data from the bivariate model.
# This method is restricted only to the Poisson and Normal response.
#######################################################################
Likelihood <- function(mdobj, Y, X, Z, beta, bi){
     if(is.null(dim(bi[[1]][,,]))){
          #the likelihood of the poisson model
          like1 = Likelihood.poisson(mdobj = mdobj, Y = Y[[1]],
           X = X[[1]], Z = Z[[1]], beta[[1]], bi[[1]][,,][1])
          #the likelihood of the normal model
          like2 = Likelihood.normal(mdobj = mdobj, Y = Y[[2]],
           X = X[[2]], Z = Z[[2]], beta[[2]], bi[[1]][,,][2])
     }else{
          #the likelihood of the poisson model
          like1 = Likelihood.poisson(mdobj = mdobj, Y = Y[[1]],
           X = X[[1]], Z = Z[[1]], beta[[1]], bi[[1]][,,][1,])
          #the likelihood of the normal model
          like2 = Likelihood.normal(mdobj = mdobj, Y = Y[[2]],
           X = X[[2]], Z = Z[[2]], beta[[2]], bi[[1]][,,][2,])
     }
     #return the total likelihood
     return (as.numeric(exp(log(like1) + log(like2))))
}

#######################################################################
# Calculate the log-likelihood assuming the Normal model.
#######################################################################
```

```
logLikelihood.normal <- function(mdobj, Y, X, Z, beta, bi){

     if(is.null(X) || is.null(beta)){
          eta =  Z%*%matrix(bi, ncol=1)
     }else{
          eta = X%*%beta + Z%*%matrix(bi, ncol=1)
     }

     kk = log(2*mdobj$phi * pi)/2 - Y*Y/(2*mdobj$phi)
     temp = (Y*eta - (eta*eta/2))/mdobj$phi + kk

     return (as.numeric(temp))
}

#####################################################################
# Calculate the log-likelihood assuming the Poisson model.
#####################################################################
logLikelihood.poisson <- function(mdobj, Y, X, Z, beta, bi){

     if(is.null(X) || is.null(beta)){
          eta = Z%*% matrix(bi, ncol=1)
     }else{
          eta = X%*%beta + Z%*% matrix(bi, ncol=1)
     }

     temp = Y*eta - exp(eta) - lfactorial(Y)
     return (as.numeric(temp))
}

#####################################################################
# Calculates the log-likelihood given the data from the bivariate model.
# This method is restricted only to the Poisson and Normal response.
#####################################################################
logLikelihood <- function(mdobj, Y, X, Z, beta, bi){
     #the log-likelihood of the poisson model
     like1 = logLikelihood.poisson(mdobj = mdobj, Y = Y[[1]],
      X = X[[1]], Z = Z[[1]], beta[[1]], bi[[1]][,,])
     #the log-likelihood of the Normal model
     like2 = logLikelihood.normal(mdobj = mdobj, Y = Y[[2]],
      X = X[[2]], Z = Z[[2]], beta[[2]], bi[[1]][,,])
     #return the total likelihood
     return (as.numeric(like1+ like2))
}

#####################################################################
```

```
# Calculate the prior density of the random effects.
#######################################################################
PriorDensity <- function(mdobj, theta){
     priorParameters <- mdobj$biPriors
     thetaDensity <- mvtnorm::dmvnorm(theta[[1]][,,],mean =
      priorParameters$mu, sigma = priorParameters$sigma)
     return(as.numeric(thetaDensity))
}

#######################################################################
# Wrapper method to update the concentration parameter of
# the Dirichlet Process.
#######################################################################
UpdateAlpha <- function(dpobj) {

     newAlpha <- update_concentration(dpobj$alpha, dpobj$units,
                 dpobj$numberClusters, dpobj$alphaPriorParameters)
     dpobj$alpha <- newAlpha

     return(dpobj)
}

#######################################################################
# Update the concentration parameter of the Dirichlet Process.
#######################################################################
update_concentration<-function(oldParam,n,nParams,priorParameters){

     x <- rbeta(1, oldParam + 1, n)

     pi1 <- priorParameters[1] + nParams - 1
     pi2 <- n * (priorParameters[2] - log(x))
     pi1 <- pi1/(pi1 + pi2)

     if (runif(1) < pi1) {
          g1 <- rgamma(1, priorParameters[1] + nParams,
                         priorParameters[2] - log(x))
          new_alpha <- g1
     } else {
          g2 <- rgamma(1, priorParameters[1] + nParams - 1,
                         priorParameters[2]-log(x))
          new_alpha <- g2
     }

     new_alpha
     return(new_alpha)
```

```
}

#####################################################################
# Create a Dirichlet Process object.
#####################################################################
DirichletProcessCreate <- function(X, Y, Z, mdObject, units, obs,
                                   alphaPriorParameters = c(1, 1)) {

     dpObj <- list(Y=Y, X = X, Z = Z,
                   mixingDistribution = mdObject,
                   units = units, obs = obs,
                   alphaPriorParameters = alphaPriorParameters,
                   alpha = rgamma(1, alphaPriorParameters[1],
                                     alphaPriorParameters[2])
               )

     class(dpObj) <- append(class(dpObj), c("dirichletprocess",
                                            class(mdObject)[-1]))

     return(dpObj)
}

#####################################################################
# Initialize the GLMM-DP object.
#####################################################################
Initialise <- function(dpObj, m=3) {

     #assign all units to the same cluster
     dpObj$clusterLabels <- rep(1, dpObj$units)
     #set the number of clusters variable
     dpObj$numberClusters <- 1
     #initialize the number of units per cluster
     dpObj$pointsPerCluster <- dpObj$units

     #initialize the fixed effects to the mean of their priors
     if(!is.null(dpObj$X)){
          dpObj$beta = list(dpObj$mixingDistribution$betaPriors$mu,
          dpObj$mixingDistribution$betaPriors$mu)
     }
     #initialize random effects to prior values
     dpObj$clusterParameters <- PriorDraw(dpObj$mixingDistribution, 1)

     #this is the h parameters (Neal's Algorithm 8)
     dpObj$m <- m
```

```
     return(dpObj)
}

######################################################################
# Assign units to clusters, expanding/shrinking number of
# clusters, as required.
######################################################################
ClusterComponentUpdate <- function(dpObj) {

     N <- dpObj$units
     alpha <- dpObj$alpha
     beta.current <- dpObj$beta

     clusterLabels <- dpObj$clusterLabels
     clusterParams <- dpObj$clusterParameters
     numLabels <- dpObj$numberClusters

     mdObj <- dpObj$mixingDistribution
     m <- dpObj$m

     pointsPerCluster <- dpObj$pointsPerCluster

     aux <- vector("list", length(clusterParams))
     #a list to hold (multivariate) responses for a unit
     Y.list = list(NULL, NULL)
     #a list to hold (multivariate) covariates (of fixed effects)
     X.list = list(NULL, NULL)
     #a list to hold (multivariate) covariates (of random effects)
     Z.list = list(NULL, NULL)

     for (i in seq_len(N)) {

          #a vector of probabilities of assigning a unit to clusters
          probs <- numeric(numLabels + 1)

          #current assignment of units to clusters
          currentLabel <- clusterLabels[i]

          #number of units per cluster, excluding the current (i^th) unit
          pointsPerCluster[currentLabel] <- pointsPerCluster[currentLabel] - 1

          if (pointsPerCluster[currentLabel] == 0) {
               priorDraws <- PriorDraw(mdObj, m - 1)

               for (j in seq_along(priorDraws)) {
```

```
          aux[[j]]<-array(c(clusterParams[[j]][,,currentLabel],
          priorDraws[[j]]),
          dim = c(dim(priorDraws[[j]])[1:2], m))
     }
} else {
     aux <- PriorDraw(mdObj, m)
}

#indices of data of the current unit
sub.index = seq((i - 1) * n + 1, i*n)

#response of the first variable of the current unit
Y.list[[1]] = dpObj$Y[[1]][sub.index,,drop=FALSE]
#response of the second variable of the current unit
Y.list[[2]] = dpObj$Y[[2]][sub.index,,drop=FALSE]

#covariates of fixed effects of the first variable
#of the current unit
X.list[[1]] = dpObj$X[[1]][sub.index, , drop=FALSE]
#covariates of fixed effects of the second variable
#of the current unit
X.list[[2]] = dpObj$X[[2]][sub.index, , drop=FALSE]

#covariates of random effects of the first variable
#of the current unit
Z.list[[1]] = dpObj$Z[[1]][sub.index, , drop=FALSE]
#covariates of random effects of the first variable
#of the current unit
Z.list[[2]] = dpObj$Z[[2]][sub.index, , drop=FALSE]

#probabilities of assigning the current unit to one
# of the existing clusters
probs[1:numLabels] <- pointsPerCluster * Likelihood(mdObj,
 Y=Y.list, bi=clusterParams,  X = X.list,
 Z = Z.list, beta = beta.current)

#probabilities of assigning the current unit to a new cluster
probs[(numLabels + 1):(numLabels + m)] <- (alpha/m) *
 Likelihood(mdObj, Y=Y.list, bi=aux,
 X = X.list, Z = Z.list, beta = beta.current)
#assign probability of 0 to those cases with NaN probability
if (any(is.nan(probs))) {
     probs[is.nan(probs)] <- 0
}
```

```
          #assign probability of 0 to those cases with NA probability
          if (anyNA(probs)) {
               probs[is.na(probs)] <- 0
          }

         #set probabilities when the probability is +/- infinity
          if (any(is.infinite(probs))) {
               probs[is.infinite(probs)] <- 1
               probs[-is.infinite(probs)] <- 0
          }

          #set all probabilities to 1 is they are all zero
          if (all(probs == 0)) {
               probs <- rep_len(1, length(probs))
          }

          #sample a new cluster assignment of the current unit
          newLabel <- sample.int(numLabels + m, 1, prob = probs)

          #update the state variables given assignment of the current
          # unit to a new cluster
          dpObj$pointsPerCluster <- pointsPerCluster
          dpObj<-ClusterLabelChange(dpObj,i,newLabel,currentLabel,aux)
          pointsPerCluster <- dpObj$pointsPerCluster
          clusterLabels <- dpObj$clusterLabels
          clusterParams <- dpObj$clusterParameters
          numLabels <- dpObj$numberClusters

     }

     dpObj$pointsPerCluster <- pointsPerCluster
     dpObj$clusterLabels <- clusterLabels
     dpObj$clusterParameters <- clusterParams
     dpObj$numberClusters <- numLabels
     return(dpObj)
}

######################################################################
# Update cluster labels as per new label of the unit i, with its
# parameters contained in aux.
######################################################################
ClusterLabelChange <- function(dpObj, i, newLabel, currentLabel, aux) {

     pointsPerCluster <- dpObj$pointsPerCluster
     clusterLabels <- dpObj$clusterLabels
```

```
clusterParams <- dpObj$clusterParameters
numLabels <- dpObj$numberClusters

if (newLabel <= numLabels) {
     pointsPerCluster[newLabel] <- pointsPerCluster[newLabel] + 1
     clusterLabels[i] <- newLabel

     if (pointsPerCluster[currentLabel] == 0) {
          numLabels <- numLabels - 1
          pointsPerCluster <- pointsPerCluster[-currentLabel]
          clusterParams <- lapply(clusterParams, function(x) x[, ,
                              -currentLabel,  drop = FALSE])
          inds <- clusterLabels > currentLabel
          clusterLabels[inds] <- clusterLabels[inds] - 1
     }
} else {

     if (pointsPerCluster[currentLabel] == 0) {

          for (j in seq_along(clusterParams)) {
               clusterParams[[j]][, , currentLabel] <- aux[[j]][, ,
                                          newLabel - numLabels]
          }
          pointsPerCluster[currentLabel] <-
           pointsPerCluster[currentLabel] + 1

     } else {
          clusterLabels[i] <- numLabels + 1
          pointsPerCluster <- c(pointsPerCluster, 1)

          for (j in seq_along(clusterParams)) {
               clusterParams[[j]] <- array(c(clusterParams[[j]],
                             aux[[j]][, , newLabel -numLabels]),
               dim = c(dim(clusterParams[[j]])[1:2],
               dim(clusterParams[[j]])[3] +1))
          }
          numLabels <- numLabels + 1
     }
}

dpObj$pointsPerCluster <- pointsPerCluster
dpObj$clusterLabels <- clusterLabels
dpObj$clusterParameters <- clusterParams
dpObj$numberClusters <- numLabels
```

```
     return(dpObj)
}

#####################################################################
# Calculates the mean and variance of a Gaussian approximation of a
# bivariate model with poisson and normal response.  The method also
# calculates the log-likelihood of the bivariate model.
#####################################################################
getGaussianApprox.poisson_normal <- function(current, eta, Y, Z,
                                             prior.mu, prior.sigma,
                                             phi1=1, phi2)
{

     #Poisson model: diagonal matrix with variances on the diagonal
     temp1 = exp(eta[[1]])
     V1 = diag(temp1[,1])
     if(length(eta[[1]])==1){
          V1 = diag(temp1[,1], ncol=1)
     }
     #the mean of the Poisson model
     lambda1 = temp1

     #Normal model: diagonal matrix with variances on the diagonal
     V2 = diag(phi2, nrow = length(eta[[2]]))
     #the mean of the normal model
     lambda2 = eta[[2]]


     block1 = t(Z[[1]])%*%V1%*%Z[[1]]/phi1^2
     block2 = t(Z[[2]])%*%V2%*%Z[[2]]/phi2^2

     #variance
     prop.var = chol2inv(
          chol(
               (adiag(block1, block2) + prior.sigma)
          )
     )

     block.mean1 = t(Z[[1]])%*%(Y[[1]] - lambda1)/phi1
     block.mean2 = t(Z[[2]])%*%(Y[[2]] - lambda2)/phi2

     block.mean = rbind(block.mean1, block.mean2)

     prop.mean = current + prop.var %*% (block.mean -
     prior.sigma%*%(current - prior.mu))
```

```
     #the log-likelihood of the Poisson model
     f1.val = ((t(Y[[1]])%*%eta[[1]] - sum(temp1))/phi1)[1,1]

     #the log-likelihood of the Normal model
     f2.val = (t(Y[[2]])%*%eta[[2]] - sum(eta[[2]]^2/2))/phi2 +
     sum(log(2*pi*phi2)/2 - Y[[2]]*Y[[2]]/(2*phi2) )

     return (list(mean=prop.mean, variance = prop.var, f =
      (f1.val + f2.val)))
}

#####################################################################
# Gaussian approximation of the posterior distribution built using
# the Bernoulli sampling distribution.
#####################################################################
getGaussianApprox.bernoulli<-function(current, eta, X, Y, prior.mu,
                                      prior.sigma, phi=1)
{
     #diagonal matrix with variances on the diagonal
     temp = exp(eta)
     if(length(eta)==1){
          V = diag(temp[,1]/(1+temp[,1])^2, ncol=1)
     }else{
          V = diag(temp[,1]/(1+temp[,1])^2)
     }
     #the mean of the Bernoulli model
     lambda = temp/(1+temp)

     #the variance matrix of the normal approximation
     prop.var = chol2inv(
          chol(
               t(X)%*%V%*%X + prior.sigma
          )
     )

     #the mean of the normal approximation
     prop.mean = current + prop.var %*% (t(X)%*%(Y - lambda) -
     prior.sigma%*%(current - prior.mu))

     #the log-likelihood
     f.val = t(Y)%*%eta - sum(log(1 + temp))

     return (list(mean=prop.mean, variance = prop.var, f = f.val))
}
```

```
#####################################################################
# Gaussian approximation of the posterior distribution built using
# the Poisson sampling distribution.
#####################################################################
getGaussianApprox.poisson <- function(current, eta, X, Y,
                                      prior.mu, prior.sigma,phi=1)
{
     #diagonal matrix with variances on the diagonal
     temp = exp(eta)
     V = diag(temp[,1])
     if(length(eta)==1){
          V = diag(temp[,1])
     }
     #the mean
     lambda = temp

     #the variance of the normal approximation
     prop.var = chol2inv(
          chol(
               (t(X)%*%V%*%X/phi^2 + prior.sigma)
          )
     )

     #the mean of normal approximation of the distribution
     prop.mean = current + prop.var %*% (t(X)%*%(Y - lambda)/phi -
     prior.sigma%*%(current - prior.mu))

     #the log-likelihood of the Poisson model
     f.val = t(Y)%*%eta - sum(temp)

     return (list(mean=prop.mean, variance =  prop.var, f = f.val))
}

#####################################################################
# This method updates the parameters of mixture components.
#####################################################################
ClusterParameterUpdate <- function(dpObj) {

     #number of clusters
     numLabels <- dpObj$numberClusters

     #cluster labels (ordered from 1 to "no of clusters")
     clusterLabels <- dpObj$clusterLabels
```

```
#cluster parameters
clusterParams <- dpObj$clusterParameters

#distribution of the mixture component (metadata)
mdobj <- dpObj$mixingDistribution

#number of observations per unit
n <- dpObj$obs

#current value of fixed effects parameters
beta.current = dpObj$beta

#number of times random effect vector has been updating
#(in Metropolis-Hastings method).
bi.count = 0

#list of multivariate responses (all units) allocated to the
# current cluster
Y.list = NULL

#fixed covariates for all units allocated to the current cluster
X.list = NULL

#random covariates for all units allocated to the current cluster
Z.list = NULL

#list of linked predictors for all units allocated to the
#current cluster
eta = list()

######################################################################
#1. update random effects using Neal's Algorithm 8
######################################################################
for (i in 1:numLabels) {
     #indices of data of all units allocated to the current cluster
     ind1 <- which(clusterLabels == i)
     ind = rep((ind1-1)*n, each=n) + rep(seq(1,n), length(ind1))

     #initialize the data of units allocated to the current cluster
     Y.list[[1]] = dpObj$Y[[1]][ind,,drop=FALSE]
     Y.list[[2]] = dpObj$Y[[2]][ind,,drop=FALSE]
     X.list[[1]] = dpObj$X[[1]][ind,,drop=FALSE]
     X.list[[2]] = dpObj$X[[2]][ind,,drop=FALSE]
     Z.list[[1]] = dpObj$Z[[1]][ind,,drop=FALSE]
     Z.list[[2]] = dpObj$Z[[2]][ind,,drop=FALSE]
```

```
#random effects of units allocated to the current cluster
bi.current = clusterParams[[1]][, , i, drop = FALSE][,,]

#linear predictor for the 1st response
eta[[1]] =  X.list[[1]]%*%beta.current[[1]] +
 Z.list[[1]]*bi.current[1]

#linear predictor for the 2nd response
eta[[2]] =  X.list[[2]]%*%beta.current[[2]] +
 Z.list[[2]]*bi.current[2]

#get normal approximation of the posterior distribution
fit.current = getGaussianApprox.poisson_normal(
                    current = bi.current, eta=eta, Y = Y.list,
                    Z = Z.list,  prior.mu = mdobj$biPriors$mu,
                    prior.sigma = mdobj$biPriors$sigma.inv,
                    phi1=1,phi2 = mdobj$phi)

#scale variance to ensure adequate acceptance ratio
fit.current$variance =  fit.current$variance * 6

#get the proposal
prop_bi = mvtnorm::rmvnorm(1, mean = fit.current$mean,
                    sigma = fit.current$variance)

prop_bi = prop_bi[1,]

#density using the proposal random effect
log.q.prop <- mvtnorm::dmvnorm(prop_bi, fit.current$mean,
                    fit.current$variance, log=TRUE)

#linear predictor of the 1st response using the
#proposal random effect
eta[[1]] =  X.list[[1]]%*%beta.current[[1]] +
 Z.list[[1]]*prop_bi[1]
eta[[2]] =  X.list[[2]]%*%beta.current[[2]] +
 Z.list[[2]]*prop_bi[2]

#normal approximation of the posterior using the
#proposed random effect
gfit.prop <- getGaussianApprox.poisson_normal(
                    current = prop_bi, eta=eta,  Y=Y.list,
                    Z = Z.list, prior.mu = mdobj$biPriors$mu,
                    prior.sigma = mdobj$biPriors$sigma.inv,
```

```
                        phi1=1,phi2 = mdobj$phi)

     #adjust the variance to ensure an adequate acceptance rate (in M-H)
     gfit.prop$variance =  gfit.prop$variance  * 6

     #density using the current random effec
     log.q <- mvtnorm::dmvnorm(bi.current, mean= gfit.prop$mean,
                         sigma=gfit.prop$variance, log=TRUE)

     #log ration (standard Metropolis-Hasting criteria)
     log.ratio <- (gfit.prop$f-fit.current$f) + (log.q-log.q.prop)
     ratio <- min(1, exp(log.ratio))
     if (is.na(ratio) | !length(ratio) ) {
          ratio <- 0
     }
     if (runif(1) < ratio) {
          #accept new random effect proposal value
          clusterParams[[1]][, , i] <- prop_bi
          bi.count = bi.count + 1
     }#else leave current parameters the same
}

#weight it by the number of clusters in the iteration
dpObj$biCount = (bi.count * 1.0)/numLabels

#update the global cluster parameter
dpObj$clusterParameters <- clusterParams

######################################################################
# 2. update the variance (hyperparameter) of random effects - START
######################################################################
cov.temp = mdobj$biPriors$rho * mdobj$biPriors$R
AA = dpObj$clusterParameters[[1]][,,dpObj$clusterLabels]
if(!is.null(dim(AA))){
     for(i in 1:ncol(AA)){
          cov.temp = cov.temp + AA[,i]%*%t(AA[,i])
     }
}else{
     cov.temp = cov.temp + sum(AA*AA)
}

conv.mat = rWishart(1, N + mdobj$biPriors$rho, solve(cov.temp))
dpObj$mdpobj$biPriors$sigma.inv = conv.mat[,,1]
dpObj$mdpobj$biPriors$sigma = solve(conv.mat[,,1])
# update the variance (hyperparameter) of random effects - START
```

```
#####################################################################
# 3. Update fixed effects parameters (corresponding to poisson
#submodel).   Use the Metropolis-Hastings method.
#####################################################################
#fixed effects parameters for the first (Poisson) submodel
beta_poisson <- dpObj$beta[[1]]

#this expression is Z_i^t * b_i
          z.eta =  dpObj$Z[[1]]*rep(dpObj$clusterParameters[[1]][,,]
          [1,,drop=FALSE][,dpObj$clusterLabels], each=n)

#the linear predictor of the submodel
eta = dpObj$X[[1]]%*%beta_poisson + z.eta

#get the normal approximation of the distribution
fit.current=getGaussianApprox.poisson(current=beta_poisson,eta=eta,
X = dpObj$X[[1]], Y = dpObj$Y[[1]],
prior.mu = mdobj$betaPriors$mu,
prior.sigma =
mdobj$betaPriors$sigma.inv_poisson)

#adjust variance in order to get an acceptable acceptance rate
fit.current$variance = fit.current$variance * 3

#get the proposal fixed effect parameter
prop_beta = mvtnorm::rmvnorm(1, mean = fit.current$mean,
sigma = fit.current$variance)

#density using the proposed value of the fixed effect parameter
log.q.prop <- mvtnorm::dmvnorm(prop_beta, mean=fit.current$mean,
sigma = fit.current$variance, log=TRUE)

#linear predictor using the proposed value of the fixed effect prameter
eta = dpObj$X[[1]]%*%t(prop_beta) + z.eta

#get the normal approximation using the updated linear predictor (above)
gfit.prop <- getGaussianApprox.poisson(current = t(prop_beta),eta=eta,
               X=dpObj$X[[1]], Y=dpObj$Y[[1]],
               prior.mu = mdobj$betaPriors$mu,
               prior.sigma = mdobj$betaPriors$sigma.inv_poisson)

#update the variance in order to get an acceptable acceptance ratio (M-H)
gfit.prop$variance = gfit.prop$variance * 3
```

```
#new density of the original fixed effect parameter
log.q.old <- mvtnorm::dmvnorm(t(beta_poisson), mean= gfit.prop$mean,
sigma=gfit.prop$variance, log=TRUE)

#log-ratio
log.ratio <- (gfit.prop$f - fit.current$f + log.q.old -log.q.prop)
ratio <- min(1, exp(log.ratio))

if (is.na(ratio) | !length(ratio) ) {
     ratio <- 0
}

#check if the new value should be accepted or not
if (runif(1) < ratio) {
     #accept the new value of fixed effect
     dpObj$beta[[1]] = t(prop_beta)
     dpObj$betaAccepted = 1
} else {
     #keep the current value of fixed effect
     dpObj$betaAccepted = 0
dpObj$beta[[1]] = beta_poisson
}

######################################################################
# 4. Update fixed effects parameters (corresponding to normal
# submodel). Posterior is derived directly (no need to use the M-H)
######################################################################
#variance of the posterior distribution
var.normal = solve(t(dpObj$X[[2]])%*%dpObj$X[[2]]/
               dpObj$mixingDistribution$betaPriors$phi.val +
               dpObj$mixingDistribution$betaPriors$sigma.inv)
#mean of the posterior mean
mean.normal = t(dpObj$X[[2]])%*%(dpObj$Y[[2]] -
               dpObj$Z[[2]]*rep(dpObj$clusterParameters[[1]][,,]
               [2,,drop=FALSE][,dpObj$clusterLabels], each=n))/
               dpObj$mixingDistribution$betaPriors$phi.val
mean.normal = mean.normal +
 dpObj$mixingDistribution$betaPriors$sigma.inv%*%
                              dpObj$mixingDistribution$betaPriors$mu
mean.normal = var.normal %*% mean.normal

#sample new value of fixed effect parameter of the normal submodel
dpObj$beta[[2]] = t(mvtnorm::rmvnorm(1, mean = mean.normal,
                                        sigma = var.normal))
```

```
    beta.current = dpObj$beta[[2]]

    ######################################################################
    # 5. Update the dispersion parameter
    ######################################################################

    #calculate the linear predictor using the (potentially new) value of
    #fixed effect parameters
    z.eta =  dpObj$Z[[2]]*rep(dpObj$clusterParameters[[1]][,,]
     [2,,drop=FALSE][,dpObj$clusterLabels], each=n)
    eta = dpObj$X[[2]]%*%beta.current + z.eta


    #shape of hyperparameter of dispersion parameter
    phi.param2.shape = mdobj$betaPriors$phi.hyper1 +
      mdobj$betaPriors$phi.ksi/2

    #rate of hyperparmeter of dispersion parameter
    phi.param2.rate = mdobj$betaPriors$phi.hyper2 +
     1/(mdobj$betaPriors$phi.val*2)

    #(2x) rate of dispersion parameter
    phi.param2 = rgamma(1,shape=phi.param2.shape,rate=phi.param2.rate)

    #shape of dispersion parameter
    phi.shape = mdobj$betaPriors$phi.ksi + mdobj$betaPriors$phi.nr
    #rate of dispersion parameter
    phi.rate = phi.param2 + t(dpObj$Y[[2]]-eta)%*%(dpObj$Y[[2]] - eta)
    phi.rate = solve(phi.rate)

    #new value of dispersion parameter
    phi = rgamma(1, shape = phi.shape, rate = 1/(2*phi.rate))

    #update beta priors state
    mdobj$mdobj$betaPriors$phi.val = 1/phi
    dpObj$mixingDistribution$betaPriors$phi.val = 1/phi

    return(dpObj)
}

######################################################################
# This method is main method in GLMM-DP model. It runs MCMC simulations
# and wraps various estimates into lists that are suitable for
# further processing.
######################################################################
```

```
Fit <- function(dpObj, its, updatePrior = FALSE, progressBar=TRUE) {

     if (progressBar){
          pb <- txtProgressBar(min=0, max=its, width=50, char="-", style=3)
     }

     #chain to contain concentration parameters of the Dirichlet Process
     alphaChain <- numeric(its)

     #mixing ratio (portion of units allocated to each cluster)
     weightsChain <- vector("list", length = its)

     #chain to contain cluster parameters (potentially different number of
     #parameters in each iteration)
     clusterParametersChain <- vector("list", length = its)

     #chain of cluster labels
     labelsChain <- vector("list", length = its)

     #chain of fixed effects parameters
     betaChain <- vector("list", length = its)

     #total number of times fixed effects parameters have been updated
     #applied only to parameters of the Poisson submodel (in the Normal
     #submodel, parameters are updated in each iteration)
     betaCount = 0

     #total weighted number of times random effects parameters were
     #updated in Metropolis-Hastings sampling
     biCount = 0

     #run MCMC
     for (i in seq_len(its)) {

          alphaChain[i] <- dpObj$alpha
          weightsChain[[i]] <- dpObj$pointsPerCluster / dpObj$units
          clusterParametersChain[[i]] <- dpObj$clusterParameters
          labelsChain[[i]] <- dpObj$clusterLabels
          betaChain[[i]]<-dpObj$beta

          #assign units to clusters
          dpObj <- ClusterComponentUpdate(dpObj)

          #update cluster parameters
          dpObj <- ClusterParameterUpdate(dpObj)
```

```
          #update concentration parameter of the Dirichlet Process
          dpObj <- UpdateAlpha(dpObj)

          if (updatePrior) {
               dpObj$mixingDistribution <-
                PriorParametersUpdate(dpObj$mixingDistribution,
                    dpObj$clusterParameters)
          }
          if (progressBar){
               setTxtProgressBar(pb, i)
          }

          #update betaCount if it was accepted in the last iteration
          betaCount = betaCount + dpObj$betaAccepted
          #update the number of times random effects have been updated
          biCount = biCount +dpObj$biCount
     }

     #update the above properties on the main object
     dpObj$weights <- dpObj$pointsPerCluster / dpObj$units
     dpObj$alphaChain <- alphaChain
     dpObj$weightsChain <- weightsChain
     dpObj$clusterParametersChain <- clusterParametersChain
     dpObj$labelsChain <- labelsChain
     dpObj$betaChain = betaChain
     dpObj$betaCount = betaCount
     dpObj$biCount = biCount

     if (progressBar) {
          close(pb)
     }

     return(dpObj)
}

#####################################################################
# Object providing basic information about a mixing distribution,
# such as the prior of fixed and random effects parameters and the
# dispersion parameter of the normal submodel.
#####################################################################
MixingDistribution<-function(distribution,betaPriors,biPriors,phi){

     mdObj <- list(distribution = distribution,
                    betaPriors = betaPriors,
```

```
                    biPriors = biPriors,
                    phi = phi)

     class(mdObj) <- append(class(mdObj), c(distribution))
     return(mdObj)
}
```

A.3 Label-switching related methods

```
library(cluster)

#####################################################################
# Calculate the similarity matrix of units, starting with MCMC
# iteration start.it and ending with itertion end.it
#####################################################################
get.similarity.matrix <- function(dp, N, start.it, end.it){

     #similatiry matrix (N is total number of units)
     result <- matrix(0, nrow=N, ncol=N)

     for(i in start.it:end.it){
          #similarity matrix for one particular iteration
          tmp = matrix(0, nrow=N, ncol=N)

          #current cluster
          cluster = dp$labelsChain[[i]]
          for(j in 1:N){
               for(k in (j+1):N){
                    #the matrix is symmetric
                    #consider only one half (upper-triangular)
                    if(j > N || k>N){
                         next
                    }
                    if(cluster[j]==cluster[k]){
                         tmp[j,k]=1
                         tmp[k,j] = 1
                    }
               }
          }

          #update the matrix with the "similarity" of current iteration
          result = result + tmp
     }

     #adjust similarity values to fall in the [0,1] range
```

```
     return (result/(end.it - start.it))
}

#####################################################################
# Find the optimal cluster given the dissimilarity matrix.
# The dissimilarity matrix = 1 - similarity matrix
#####################################################################
get.optimal.cluster <- function(maxNClusters, disSimMat){
     clustVec = NULL
     chosenNClusters = NULL
     clustSizes <- NULL
     clustMedoids <- NULL

     # Loop over the possible number of clusters
     avgSilhouetteWidth<--1.0;
     cat(paste("Max no of possible clusters:",maxNClusters,"\n"))
     for(c in 2:maxNClusters){
          cat(paste("Trying",c,"clusters\n"))
          tmpObj<-pam(disSimMat,k=c,diss=T)
          # Check whether the silhouette width from this clustering
          # improves previous best
          if(avgSilhouetteWidth<tmpObj$silinfo$avg.width){
               avgSilhouetteWidth<-tmpObj$silinfo$avg.width
               chosenNClusters<-c
               clustVec<-tmpObj$clustering
               clustSizes<-tmpObj$clusinfo[,1]
               # The id of the objects chosen as the medoids
               clustMedoids<-tmpObj$id.med
          }
     }

     return (list(
          "nClusters"=chosenNClusters,
          "clusObjDisSimMat"=disSimMat,
          "clusterSizes"=clustSizes,
          "clustering"=clustVec,
          "avgSilhouetteWidth"=avgSilhouetteWidth))
}

#####################################################################
# Estimate fixed effects parameters
#####################################################################
estimate.bi <- function(dp, best.cluster, N, start.it, end.it ){

     # dimension of the random effects vector
```

```
     b.dim = dim(dp$clusterParameters[[1]][,,1, drop=FALSE])[2]

     # matrix of estimates of random effects (per MCMC iteration)
     result <- matrix(0, nrow=(end.it - start.it+1),
     ncol= best.cluster$nClusters* b.dim)

     for(i in start.it:end.it){
          current.cluster = dp$labelsChain[[i]]
          for(j in 1:best.cluster$nClusters){

               #find indices parameters for the current (j-th) unit in cluster
               bi.index =seq(from=(b.dim*j-(b.dim-1)), to=(b.dim*j))

               #get the parameters of the current cluster
               bi.parameters = dp$clusterParametersChain[[i]][[1]][,,
                current.cluster]

               if(is.null(dim(bi.parameters))){
                    #parameters in the current cluster that are also
                    #in the best cluster
                    bi.parameters = bi.parameters[best.cluster$clustering ==
                      j, drop=FALSE]

                    #parameters (averaged over size of the best cluster)
                    result[(i-start.it + 1),bi.index] =
                         sum(bi.parameters)/best.cluster$clusterSizes[j]
               }else{
                    bi.parameters = bi.parameters[,best.cluster$clustering ==
                      j, drop=FALSE]
                    result[(i-start.it + 1),bi.index] =
                         rowSums(bi.parameters)/best.cluster$clusterSizes[j]
               }
          }
     }

     return(result)
}

#####################################################################
# This is a utility function.
#####################################################################
get.correct <- function(indices){
     result = sort(table(indices), decreasing = TRUE)
     name = as.integer(names(result)[1])
     return (sum(indices == name))
```

```
}
```

A.4 The main code that starts the estimation

```
source('./script_multivariate.R')
source('./get.mult.data.R')
source('./get.similarity.matrix.R')

#number of MCMC iterations
iterations = 10000

#beta of the first model
beta.poisson = c(0.8, -0.6, 0.3)

#beta of the second model
beta.normal = c(-0.5, 0.5, 0.4)

#random effect of the first cluster in the normal model
bi1 = 0.5
#random effect of the second cluster in the normal model
bi2 = 1.05

#random effect of the poisson model
bi.poisson = list(c(1.15), c(-0.5) )

#random effect of the normal model
bi.normal = list(c(bi1), c(bi2))

#number of units in the study
N = 50

#number of observations per unit
n = 10

#generate data

my.data = get_poisson_data(units = N, obs.per.unit=n,
beta=beta.poisson, bi=bi.poisson)
my.data = add_normal_response(X = my.data$X, Y = my.data$Y, Z = my.data$Z,
obs.per.unit = n, beta =beta.normal, bi = bi.normal )


##########################################
# PRIORS INFORMATION - START
```

```
##########################################
#vector containing priors of fixed effects parameters
betaPriors = NULL

#prior of the mean of random effects parameters
betaPriors$mu_poisson = matrix(c(0, 0, 0), ncol=1)

#prior covariance of fixed effects Poisson parameters
betaPriors$sigma_poisson = diag(100,nrow=3, ncol=3)
betaPriors$sigma.inv_poisson = solve(betaPriors$sigma)

#prior of the mean in Normal model
betaPriors$mu_normal = matrix(c(0, 0, 0), ncol=1)

#prior of the covariance matrix of the Normal model
betaPriors$sigma_normal = diag(100,nrow=3, ncol=3)
betaPriors$sigma.inv_normal = solve(betaPriors$sigma)

#dispersion parameters and hyperparameters
betaPriors$phi.val = 0.1   ##this phi^-1: inverse of variance
betaPriors$phi.ksi = 2
betaPriors$phi.hyper1 = 0.2  #dispersion hyperparam 1
betaPriors$phi.hyper2 = 2.5  #dispersion hyperparam 2
betaPriors$phi.nr = N*n  #number of observations of the rth marker

#vector containing priors of random effects parameters
biPriors = NULL

#prior mean of the random effects parameter
biPriors$mu = c((1.15 + bi1)/2, (bi2 - 0.5)/2)#, -2)

#hyperparameeters of covariance matrix of random effects parameters
biPriors$rho = length(biPriors$mu)
biPriors$R = diag(100, nrow=length(biPriors$mu))
biPriors$R.inv = solve(biPriors$R)

biPriors$sigma = biPriors$R
biPriors$sigma.inv = biPriors$R.inv
##########################################
# PRIORS INFORMATION - END
##########################################


glmmpMd<- MixingDistribution("glmmp", betaPriors = betaPriors,
biPriors = biPriors,  phi=0.1)
```

```
print(Sys.time())
dp <- DirichletProcessCreate(X=my.data$X, Y=my.data$Y,
Z = my.data$Z,  mdObject = glmmpMd, units = N, obs = n)
dp <- Initialise(dp, m = )
print(iterations)
dp <- Fit(dp, its = iterations)

print("beta acceptance rate")
print(dp$betaCount/iterations)

print("bi acceptance rate")
print(dp$biCount/iterations)



##############################################
# POST-PROCESS - START
##############################################

#the first MCMC iterate to use for processing
start.it = iterations/2

#the last MCMC iterate to use for processing
end.it = iterations

#calculate the similarity matrix
similarity.matrix = get.similarity.matrix(dp=dp, N=N,
start.it=start.it, end.it = end.it)

#find the optimal clustering (with max # of clusters being 3)
optimal.cluster = get.optimal.cluster(maxNClusters =  3,
disSimMat =  1 - similarity.matrix)

#fins the optimal clustering (with max # of clusters being 10)
optimal.clusterCL = get.optimal.cluster(maxNClusters = 10,
disSimMat =  1 - similarity.matrix)

#check if optimal.clusterCL returns different number of
#clusters than optimal.cluster
#if so, then investigate

#get the estimates of random effects (per MCMC iterate)
bi.estimates = estimate.bi(dp=dp, best.cluster = optimal.cluster,
 N=N, start.it = start.it, end.it=end.it)
```

```
#get the final MCMC estimate of random effects
bi.output = colMeans(bi.estimates)

#variance of random effects estimates
vars = matrix( apply(bi.estimates,2,var), nrow=1)

#combined random effects
bi.input = c(unlist(bi.poisson), unlist(bi.normal))

#sort random effects so we can perform calculations on it
bi.input = sort(bi.input)
bi.output = sort(bi.output)

#calculate the MSE
RMSE = 0
RMSE.sum = 0
if(optimal.cluster$nClusters == 2){
     for(i in 1:length(bi.output)){
          RMSE = RMSE + (bi.output[i]-bi.input[i])^2
     }

     RMSE = sqrt(RMSE/length(bi.output))

     RMSE.sum = sum(RMSE)
     RMSE = round(RMSE, digits = 4)
     RMSE.sum = round(RMSE.sum, digits = 4)
}

#optimal clustering
optimal.clustering = optimal.cluster$clustering

class.correct = get.correct(optimal.clustering[1:(N/2)]) +
get.correct(optimal.clustering[(N/2 + 1):N])
class.correct = class.correct * 1.0
class.incorrect = N - class.correct


#estimate fixed effect parameters of the Poisson model
beta.save.poisson = matrix(0, nrow = (end.it - start.it + 1),ncol=3)
for(i in start.it:end.it){
     beta.save.poisson[i-start.it + 1,] = dp$betaChain[[i]][[1]]
}
beta.average.poisson = colMeans(beta.save.poisson)
```

```
#estimate fixed effect parameters of the normal model
beta.save.normal = matrix(0, nrow = (end.it - start.it + 1),ncol=3)
for(i in start.it:end.it){
     beta.save.normal[i-start.it + 1,] = dp$betaChain[[i]][[2]]
}
beta.average.normal = colMeans(beta.save.normal)
############################################
# POST-PROCESS - END
############################################
```