2013

Regularization Methods for Predicting an Ordinal Response using Longitudinal High-dimensional Genomic Data

Jiayi Hou
Virginia Commonwealth University

REGULARIZATION METHODS FOR PREDICTING AN ORDINAL RESPONSE USING LONGITUDINAL HIGH-DIMENSIONAL GENOMIC DATA

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at Virginia Commonwealth University

By

Jiayi Hou
B.S. Mathematics, Sichuan University, 2008

Advisor: Kellie J. Archer
Associate Professor, Department of Biostatistics
Director, VCU Massey Cancer Center Biostatistics Shared Resource

Virginia Commonwealth University
Richmond, Virginia
December, 2013

Acknowledgement

I own my sincere thanks to countless people who have helped, supported, encouraged me during my long Ph.D journey. Without your help, this thesis would not be possible. First and foremost, I must thank my thesis advisor Dr. Kellie Archer for the generous support she gave to help pave the path to achievement. Dr.Archer has inspired me to devote to statistical learning area by her enthusiasm, vision and determination. Learning from Dr. Archer has been the greatest pleasure and will be the lifetime treasure.

Thank you to the rest of my thesis committee: Dr. Chris Gennings, Dr. Robert Johnson who are experts in the fields of biostatistics; Dr. Sam Chen, who has tremendous research experience in genetics and genomics; and Dr. Juan Lu, who contributes enormously to the epidemiology area. Your constructive feedback, rich support and kind encouragement help me grow both as a researcher and as a person. I truly appreciate the time, effort and energy you dedicated to help make this happen. Many other great supervisors I got to know through various projects and positions have taught me a lot. Thank you to Dr. Mark Reimers for introducing me to the field of bioinformatics and offering me the rigorous training. Thank you to Dr. Charlie Kish and Dr. Carol Summitt for providing a unique opportunity to gain real industrial experience and broadening the areas where my knowledge can be applied. Thank you to Dr. Phillip Yates, Dr. Max Kuhn and other team members at Pfizer Inc for offering me an enjoyable, memorable and cool summer in Connecticut.

I owe a big thanks to my classmates Adam Sima, Caroline Carrico, Amber Wilk, Sarah

Reese, Chunfeng Ren, Qing Zhou, Yan Jin who grow, laugh, gossip and enjoy graduate school with me. There were other people in the department who provided generous support and helped me get through school: Yvonne Hargrove, Gayle Spivery, Helen Wang, Brian Bush, Russ Boyle, Dr. Donna McClish, Dr. Roy Sabo, Dr. Jessica Ketchum, Dr. Guimin Gao and Dr. Shumei Sun, I owe you a great debt of gratitude. I am grateful to all my other friends in Richmond, VA for your accompaniment, tolerance and encouragement which makes the journey wonderful.

Finally, I owe everything to my parents and family members who unconditionally support me to pursue my dream.

I feel blessed and will always cherish the memory at Virginia Commonwealth University.

Table of Contents

List of Figures

List of Tables

Abstract

REGULARIZATION METHODS FOR PREDICTING AN ORDINAL RESPONSE USING LONGITUDINAL HIGH-DIMENSIONAL GENOMIC DATA

By Jiayi Hou

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at Virginia Commonwealth University

Virginia Commonwealth University, 2013

Director: Kellie J. Archer, Ph.D., Associate Professor, Department of Biostatistics; Director, VCU Massey Cancer Center Biostatistics Shared Resource

Ordinal scales are commonly used to measure health status and disease related outcomes in hospital settings as well as in translational medical research. Notable examples include cancer staging, which is a five-category ordinal scale indicating tumor size, node involvement, and likelihood of metastasizing. Glasgow Coma Scale (GCS), which gives a reliable and objective assessment of conscious status of a patient, is an ordinal scaled measure. In addition, repeated measurements are common in clinical practice for tracking and monitoring the progression of complex diseases. Classical ordinal modeling methods based on the likelihood approach have contributed to the analysis of data in which the response categories

are ordered and the number of covariates (p) is smaller than the sample size (n). With the emergence of genomic technologies being increasingly applied for obtaining a more accurate diagnosis and prognosis, a novel type of data, known as high-dimensional data where the number of covariates (p) is much larger than the number of samples (n), are generated. However, corresponding statistical methodologies as well as computational software are lacking for analyzing high-dimensional data with an ordinal or a longitudinal ordinal response. In this thesis, we develop a regularization algorithm to build a parsimonious model for predicting an ordinal response. In addition, we utilize the classical ordinal model with longitudinal measurements to incorporate the cutting-edge data mining tool for a comprehensive understanding of the causes of complex disease on both the molecular level and environmental level. Moreover, we develop the corresponding R package for general utilization. The algorithm was applied to several real datasets as well as to simulated data to demonstrate the efficiency in variable selection and precision in prediction and classification. The four real datasets are from: 1) the National Institute of Mental Health Schizophrenia Collaborative Study; 2)the San Diego Health Services Research Example; 3) A gene expression experiment to understand 'Decreased Expression of Intelectin 1 in The Human Airway Epithelium of Smokers Compared to Nonsmokers' by Weill Cornell Medical College; and 4) the National Institute of General Medical Sciences Inflammation and the Host Response to Burn Injury Collaborative Study.

This thesis is organized as follows: In Chapter 1 we introduce the concept of ordinal data and the statistical framework of the ordinal model. In Chapter 2, we review existing regularization methods for fitting linear and logistic models in a high-dimensional data setting. In Chapter 3, we review three statistical models: linear mixed models, nonlinear mixed models and generalized linear mixed models, which are suitable for different types of longitudinal data. We derive the random coefficient model which is capable of fitting longitudinal data with an ordinal response in Chapter 4. In Chapter 5, we first state the problem of interest and introduce the penalized model as a solution for performing feature selection in high-dimensional data with an ordinal response. We then combine the random coefficient model proposed in Chapter 4 and the penalized model to solve the dimension reduction and prediction problem in the longitudinal high-dimensional setting with an ordinal response. In Chapter 6, we present the results from the proposed methods when applied to several real as well as simulated datasets. Conclusions and future work are summarized in Chapter 7. All R, SAS, WinBUGS code and additional tables are provided in the Appendix.

Chapter 1

Introduction to Ordinal Model

Measuring data on an ordinal scale has been common in health status and disease-associated outcome measurement for clinical and medical research. The response of interest is represented as a series of ordered categories with either an increasing or decreasing underlying trend. A typical example of an ordinal response is to measure pain using four ordered categories: No Pain, Mild Pain, Moderate Pain and Severe Pain. Previously, there were two common approaches to analyze an ordinal response. First, treat the ordinal variable as nominal variable by breaking it into multiple dichotomous outcomes and conduct pairwise comparisons for all possible combinations. The drawback of this approach is obvious where information depicted by the underlying trend is totally ignored and not included in the analysis. Moreover, in the presence of high-dimensional data, this approach is questionable due to compounding the multiple comparison issue and may tremendously inflate the Type I error, leading to false positive discoveries. Second, treat the ordinal scale as a continuous response. This approach does not preserve the ceiling and floor effect [Agresti, 2010] of an ordinal measurement, that is, the observed and fitted responses may not have the same range. Besides, the continuous approach reflects more on individual heterogeneity rather than ordinal cluster homogeneity.

The ordinal models, which were extended primarily from the logistic and probit regression models, have been actively used to analyze ordinal response data. In particular, the ordinal model framework under the proportional odds assumption proposed by McCullagh [1980] has served as a bedrock for modern ordinal analysis. It provides a flexible structure to incorporate linear effects, nonlinear effects as well as random effects to construct a more complex model suitable for modeling different datasets.

This chapter is organized as follows: we start with Section 1.1 by introducing the mathematical notations and statistical properties of ordinal responses. In Section 1.2, we briefly review different types of ordinal models. The maximum likelihood approach and optimization techniques for ordinal models in traditional data analysis settings where $n > p$ are discussed in Section 1.3. The example of fitting the ordinal model using data from a Schizophrenia study is provided in Section 1.4.

1.1 Ordinal Responses

Let Y_i be a categorical response for observation i with C categories. We assume the observed probability Y_i falls into the c^{th} category with a probability $\pi_{ic}, c = 1, \cdots, C$. Suppose Y_i follows a multinomial distribution with trial size 1, the corresponding probability mass function for Y_i can be written as:

$$f(Y_i; \pi_1, \cdots, \pi_C) = \pi_{i1}^{y_{i1}} \cdots \pi_{ic}^{y_{ic}} \cdots \pi_{iC}^{y_{iC}}, \sum_{c=1}^{C} \pi_{ic} = 1 \tag{1.1.1}$$

where $(y_{i1}, \cdots, y_{iC})$ is an indicator vector with $y_{ic} = 1$ if Y_i falls into the c^{th} category and $y_{ic', c \neq c'} = 0$ otherwise. Thus, the probability Y_i falls into the c^{th} category can be calculated as $P(Y_i = c) = \pi_{i1}^0 \cdots \pi_{ic}^1 \cdots \pi_{iC}^0 = \pi_{ic}$, which remains consistent with the previous assumption. Correspondingly, the probability that response Y_i falls into lower than or equal to the c^{th} category can be calculated by summing up c mutually exclusive π_{ic} values where $P(Y_i \leq c) = P(Y_i = 1) + \cdots + P(Y_i = c) = \pi_{i1} + \cdots + \pi_{ic}$.

1.2 Model Framework for Ordinal Responses

To build an ordinal model, it is essential to connect the probabilities $(\pi_{i1}, \cdots, \pi_{iC})$ to the explanatory variables $\mathbf{x_i}$. Let γ_{ic} be a function of probabilities $(\pi_{i1}, \cdots, \pi_{iC})$. Suppose a monotone, differentiable link function $g(\cdot)$ connects γ_{ic} to the linear component $\alpha_c + \mathbf{x_i^T}\boldsymbol{\beta}$ such that:

$$g(\gamma_{ic}) = \alpha_c + \mathbf{x_i^T}\boldsymbol{\beta}, c = 1, \cdots, C-1 \tag{1.2.1}$$

where α_c denotes the category-specific intercept; $\boldsymbol{\beta}$ is a $p \times 1$ vector representing the coefficients associated with explanatory variables $\mathbf{x_i}$. Under the proportional odds assumption described by McCullagh [1980], $\boldsymbol{\beta}$ has the same effects for each $g(\gamma_c)$, in other words, the explanatory variables do not have category-specific effects. The validity of the proportional odds assumption can be verified by using a deviance or score test to compare the proportional odds model with the non-proportional odds model [Agresti, 2010]. In total, there are $p + C - 1$ unknown parameters that need to be estimated under the proportional odds assumption in contrast to $(p + 1) \times (C - 1)$ if not under proportional odds assumption. Thus the proportional odds assumption increases the model efficiency and requires a much smaller sample size. In this thesis, we will only focus on models under the proportional odds assumption unless stated otherwise.

There are different types of ordinal models depending on the choice of the link function $g(\cdot)$ as well as the form of probability γ_{ic}. In this thesis, we will primarily discuss the ordinal model extended from logistic regression, wherein the link function connecting γ_{ic} to the linear component is the logit link $g(x) = \log\left(\frac{x}{1-x}\right)$. The logit link along with the proportional odds assumption provide a concise method for coefficient interpretation. Here, we briefly illustrate the general relationship between the log odds ratio and unknown parameters in the model, and a more detailed interpretation will be discussed for each type of ordinal model.

1) The log odds ratio, which measures the degree of association between two ordinal levels, can be explained by the difference between the corresponding intercepts only. That is:

$$logit(\gamma_{ic}) - logit(\gamma_{ic'}) = \log\left(\frac{\gamma_{ic}/(1-\gamma_{ic})}{\gamma_{ic'}/(1-\gamma_{ic'})}\right) = \alpha_c - \alpha_{c'} \tag{1.2.2}$$

2) Given a fixed γ_c, the change of explanatory variable $\Delta\mathbf{x_i}$ will yield to a change of $\Delta\mathbf{x_i^T}\boldsymbol{\beta}$

is the log odds ratio. That is:

$$logit(\gamma_c|\mathbf{x_i}) - logit(\gamma_c|\mathbf{x_{i'}}) = \Delta\mathbf{x_i^T}\boldsymbol{\beta}, \text{where } \Delta\mathbf{x_i} = \mathbf{x_i} - \mathbf{x_{i'}} \tag{1.2.3}$$

1.2.1 Cumulative Logit Model

We start with the most commonly used type of ordinal model: the cumulative logit model. Let γ_{ic} measure the probability of response Y_i falling into no greater than the c^{th} category. Letting $\gamma_{ic} = P(Y_i \leq c|\mathbf{x_i})$ for observation i, the cumulative logit can be written as:

$$\log\left(\frac{\gamma_{ic}}{1-\gamma_{ic}}\right) = \log\left(\frac{P(Y_i \leq c|\mathbf{x_i})}{P(Y_i > c|\mathbf{x_i})}\right) = \alpha_c + \mathbf{x_i^T}\boldsymbol{\beta}, c = 1, \cdots, C-1 \tag{1.2.4}$$

where α_c denotes the category-specific intercept; $\boldsymbol{\beta}$ is a $p \times 1$ vector of coefficients associated with explanatory variables $\mathbf{x_i}$. From (1.2.4), we can rewrite γ_{ic} as a function of the linear component and each probability $\pi_{ic}(\mathbf{x_i})$ can be calculated as the difference between two adjacent γ_{ic} values, where

$$\begin{aligned}\pi_{ic}(\mathbf{x_i}) &= \gamma_{ic} - \gamma_{i,c-1} = P(Y_i \leq c|\mathbf{x_i}) - P(Y_i \leq c-1|\mathbf{x_i}) \\ &= \frac{\exp(\alpha_c + \mathbf{x_i^T}\boldsymbol{\beta})}{1+\exp(\alpha_c + \mathbf{x_i^T}\boldsymbol{\beta})} - \frac{\exp(\alpha_{c-1} + \mathbf{x_i^T}\boldsymbol{\beta})}{1+\exp(\alpha_{c-1} + \mathbf{x_i^T}\boldsymbol{\beta})}.\end{aligned} \tag{1.2.5}$$

Since π_{ic} has to be nonnegative, given the same $\mathbf{x_i^T}\boldsymbol{\beta}$, it is not hard to show $\alpha_c \geq \alpha_{c-1}$ from (1.2.5). Thus for equation (1.2.4), we place the following constraint on the intercepts: $-\infty = \alpha_0 \leq \alpha_1 \leq \cdots \leq \alpha_C = \infty$.

1.2.2 Adjacent Categories Model

The adjacent categories model, as its name suggests, concentrates on the comparisons of probabilities from two adjacent categories. Let γ_{ic} be the conditional probability that Y_i falls into the $(c+1)^{th}$ category given Y_i falls into either the c^{th} or the $(c+1)^{th}$ category. Mathematically, this conditional probability can be denoted as: $\gamma_{ic} = P(Y_i = c+1 | Y_i = c \text{ or } c+1, \mathbf{x_i}) = \frac{\pi_{i,c+1}(\mathbf{x_i})}{\pi_{ic}(\mathbf{x_i}) + \pi_{i,c+1}(\mathbf{x_i})}$. The adjacent categories ordinal model can be written as:

$$\log\left(\frac{\gamma_{ic}}{1-\gamma_{ic}}\right) = \log\left(\frac{\pi_{i,c+1}(\mathbf{x_i})}{\pi_{ic}(\mathbf{x_i})}\right) = \alpha_c + \mathbf{x_i^T}\boldsymbol{\beta},\ c = 1, \cdots, C-1. \tag{1.2.6}$$

The construction of the adjacent-categories model takes the ordering of the response categories into consideration. It also has a close connection to the baseline-category model, which is generally used to model nominal responses. However, rather than model the probabilities from two adjacent categories, the baseline-category model models the log ratio of the probability from a selected category to that from the baseline category. The choice of the baseline category can be the first category, the last category, or any category of research interest. The connections between these two models can be depicted as:

$$\begin{aligned} \log\left(\frac{\pi_{ic}(\mathbf{x_i})}{\pi_{i1}(\mathbf{x_i})}\right) &= \log\left(\frac{\pi_{ic}(\mathbf{x_i})}{\pi_{i,c-1}(\mathbf{x_i})}\right) + \log\left(\frac{\pi_{i,c-1}(\mathbf{x_i})}{\pi_{i,c-2}(\mathbf{x_i})}\right) + \cdots + \log\left(\frac{\pi_{i2}(\mathbf{x_i})}{\pi_{i1}(\mathbf{x_i})}\right) \\ &= (\alpha_{c-1} + \mathbf{x_i^T}\boldsymbol{\beta}) + (\alpha_{c-2} + \mathbf{x_i^T}\boldsymbol{\beta}) + \cdots + (\alpha_1 + \mathbf{x_i^T}\boldsymbol{\beta}) \\ &= \sum_{c=1}^{c-1} \alpha_c + (c-1)\mathbf{x_i^T}\boldsymbol{\beta}. \end{aligned} \tag{1.2.7}$$

Following the framework of the baseline-category model, we can derive an expression for the probability for each category. For an adjacent-categories model with C ordered categories, we need $C-1$ baseline-category models to specify the relation between any category to the baseline category, where we set the 1^{st} category as the baseline category to exemplify. Thus, we can obtain the following set of equations:

$$
\begin{aligned}
\log\left(\tfrac{\pi_{i2}(\mathbf{x_i})}{\pi_{i1}(\mathbf{x_i})}\right) &= \alpha_1 + \mathbf{x_i^T}\boldsymbol{\beta} = \eta_1(\mathbf{x_i}) \\
\log\left(\tfrac{\pi_{i3}(\mathbf{x_i})}{\pi_{i1}(\mathbf{x_i})}\right) &= (\alpha_1 + \alpha_2) + 2\mathbf{x_i^T}\boldsymbol{\beta} = \eta_2(\mathbf{x_i}) \\
&\cdots \\
\log\left(\tfrac{\pi_{iC}(\mathbf{x_i})}{\pi_{i1}(\mathbf{x_i})}\right) &= \textstyle\sum_{c=1}^{C-1}\alpha_c + (C-1)\mathbf{x_i^T}\boldsymbol{\beta} = \eta_{C-1}(\mathbf{x_i}).
\end{aligned}
\tag{1.2.8}
$$

By summing the equations in (1.2.8) and using the properties of logarithms as well as the constraint $\sum_{c=1}^{C}\pi_{ic}(\mathbf{x_i}) = 1$, we obtain an additional equation:

$$
\log\left(\frac{1-\pi_{i1}(\mathbf{x_i})}{\pi_{i1}(\mathbf{x_i})}\right) = \sum_{c=1}^{C-1}\eta_c(\mathbf{x_i}). \tag{1.2.9}
$$

From (1.2.9), we can obtain the expression of $\pi_{i1}(\mathbf{x_i})$ as a function of $\eta_c(\mathbf{x_i})$, where $\pi_{i1}(\mathbf{x_i}) = \frac{\exp(\sum_{c=1}^{C-1}\eta_c(\mathbf{x_i}))}{1+\exp(\sum_{c=1}^{C-1}\eta_c(\mathbf{x_i}))}$. Sequentially, putting $\pi_{i1}(\mathbf{x_i})$ back in equations (1.2.8) and solving, we can write the other probabilities as functions of the unknown parameters and the explanatory variables.

1.2.3 Continuation Ratio Model

The continuation ratio logit model measures the probability of a given category versus probabilities from lower or higher order categories. Correspondingly, let γ_{ic} be the conditional probability that Y_i falls into the c^{th} category given that Y_i falls into any category lower than or equal to the c^{th} category. Mathematically, this conditional probability can be illustrated as: $\gamma_{ic} = P(Y_i = c|Y_i \leq c, \mathbf{x_i})$. Conversely, we can redefine γ_{ic} as $\gamma_{ic} = P(Y_i = c|Y_i \geq c, \mathbf{x_i})$. Therefore, there are two types of continuation ratio models depending on the direction: the backward continuation ratio (1.2.10) and the forward continuation ratio (1.2.11) as shown below.

$$\log\left(\frac{\gamma_{ic}}{1-\gamma_{ic}}\right) = \log\left(\frac{P(Y_i = c|\mathbf{x_i})}{P(Y_i < c|\mathbf{x_i})}\right) = \alpha_c + \mathbf{x_i^T}\boldsymbol{\beta},\ c = 2, \cdots, C \tag{1.2.10}$$

$$\log\left(\frac{\gamma_{ic}}{1-\gamma_{ic}}\right) = \log\left(\frac{P(Y_i = c|\mathbf{x_i})}{P(Y_i > c|\mathbf{x_i})}\right) = \alpha_c + \mathbf{x_i^T}\boldsymbol{\beta},\ c = 1, \cdots, C-1 \tag{1.2.11}$$

Unlike the cumulative logit model, the continuation ratio model is affected by the forward versus backward direction since each logit is making use of partial information available, with the exception that the last logit in backward continuation ratio model and the first logit in forward continuation ratio model. As discussed in [Agresti, 2010], the forward continuation ratio model is often implemented to describe a sequential process with the outcome being determined afterwards, such as the survival years after interventions of tumor growth size. In contrast, the backward continuation ratio model describes the progress of an event to a specific stage given it has passed all previous stages. Often, the backward continuation ratio model is used to describe an irreversible disease process.

To obtain the probability of each category from backward continuation ratio model, we again assume there are C ordered categories in the response and we need $C-1$ models to specify each π_{ic}. Thus, we can obtain the following set of equations:

$$\begin{aligned}
\log\left(\frac{\pi_{i2}(\mathbf{x_i})}{\pi_{i1}(\mathbf{x_i})}\right) &= \alpha_2 + \mathbf{x_i^T}\boldsymbol{\beta} = \eta_2(\mathbf{x_i}) \\
\log\left(\frac{\pi_{i3}(\mathbf{x_i})}{\pi_{i1}(\mathbf{x_i})+\pi_{i2}(\mathbf{x_i})}\right) &= \alpha_3 + \mathbf{x_i^T}\boldsymbol{\beta} = \eta_3(\mathbf{x_i}) \\
&\cdots \\
\log\left(\frac{\pi_{i,C-1}(\mathbf{x_i})}{1-\pi_{iC}(\mathbf{x_i})-\pi_{i,C-1}(\mathbf{x_i})}\right) &= \alpha_{C-1} + \mathbf{x_i^T}\boldsymbol{\beta} = \eta_{C-1}(\mathbf{x_i}) \\
\log\left(\frac{\pi_{iC}(\mathbf{x_i})}{1-\pi_{iC}(\mathbf{x_i})}\right) &= \alpha_C + \mathbf{x_i^T}\boldsymbol{\beta} = \eta_C(\mathbf{x_i}).
\end{aligned} \tag{1.2.12}$$

Solving the last equation (1.2.12), we can obtain the expression of $\pi_{iC}(\mathbf{x_i})$ as a function of $\eta_C(\mathbf{x_i})$. Sequentially, we can write $\pi_{i,C-1}(\mathbf{x_i})$ as a function of $\eta_C(\mathbf{x_i}), \eta_{C-1}(\mathbf{x_i})$ and so on and so forth. Following the same logic, in the forward continuation ratio model, we start with the first category and write $\pi_{i1}(\mathbf{x_i})$ as a function of $\eta_1(\mathbf{x_i})$. Then the probability from the second category $\pi_{i2}(\mathbf{x_i})$ can be written as a function of $\eta_1(\mathbf{x_i}), \eta_2(\mathbf{x_i})$. The other probabilities can be solved successively.

1.3 Estimation of the Coefficients

To estimate the unknown parameters in an ordinal model, a maximum likelihood approach is used. We first construct the likelihood function for the four different ordinal models. Since generally, there is no closed-form of the MLE for the ordinal model. An iterative optimization method is required to obtain the MLE. We then introduce several general-purpose iterative optimization methods as well as the R functions and SAS procedures for fitting the ordinal models.

1.3.1 Maximum Likelihood Estimate

Since the ordinal response Y_i follows a multinomial distribution with trial size 1, the likelihood function for n observation is

$$L(\boldsymbol{\alpha}, \boldsymbol{\beta}; \mathbf{x}) = \prod_{i=1}^{n} f(\mathbf{x_i}; \boldsymbol{\alpha}, \boldsymbol{\beta}) = \prod_{i=1}^{n} \big(\prod_{c=1}^{C} \pi_c(\mathbf{x_i})^{y_{ic}} \big). \tag{1.3.1}$$

As discussed in Section 1.2, $\pi_c(\mathbf{x_i})$ is a function of unknown parameters $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$ and $\mathbf{x_i}$ where $\mathbf{x_i}$ are the independent covariates. $\pi_c(\mathbf{x_i})$ has a specified form determined by the ordinal model type. Here, we illustrate the maximum likelihood estimate approach using cumulative logit ordinal model, where (1.3.1) can be further written as:

$$\begin{aligned} L(\boldsymbol{\alpha}, \boldsymbol{\beta}; \mathbf{x}) &= \prod_{i=1}^{n} \big(\prod_{c=1}^{C} \pi_c(\mathbf{x_i})^{y_{ic}} \big) \\ &= \prod_{i=1}^{n} \left[\prod_{c=1}^{C} \left(\frac{\exp(\alpha_c + \mathbf{x_i^T} \boldsymbol{\beta})}{1 + \exp(\alpha_c + \mathbf{x_i^T} \boldsymbol{\beta})} - \frac{\exp(\alpha_{c-1} + \mathbf{x_i^T} \boldsymbol{\beta})}{1 + \exp(\alpha_{c-1} + \mathbf{x_i^T} \boldsymbol{\beta})} \right)^{y_{ic}} \right]. \end{aligned} \tag{1.3.2}$$

Since the unknown parameters α_c and $\boldsymbol{\beta}$ are nonlinear in the likelihood function (as well as the log likelihood function) to be optimized, an iterative method is required to successively find optimum roots for the function. McCullagh [1980] and Walker and Duncan [1967] proposed Fisher's scoring to obtain the maximum likelihood estimate. Notice there is an inequality constraint on the intercept for the cumulative logit model where $-\infty \leq \alpha_0 \leq \alpha_1 \leq \cdots \alpha_C = \infty$ as discuss in Section 1.2.1. We used the constrained nonlinear optimization algorithm Augmented Lagrangian Adaptive Barrier Minimization proposed by Varadhan [2011] to obtain the maximum likelihood estimate of the cumulative logit model. For other types of ordinal models, the iterative algorithm for solving unconstrained nonlinear optimization problem would be appropriate.

1.3.2 Optimization Technique

There are a variety of algorithms suitable for solving unconstrained nonlinear optimization problems. The three commonly used algorithms based on the second-order Taylor series expansion of the likelihood function are: Newton-Raphson, Fisher's Scoring, and Iteratively Reweighted Least Squares (IRLS). Here, we introduce the Newton-Raphson and Fisher's Scoring algorithms and leave the IRLS to be introduced in Chapter 3.

We first briefly explain the mechanics of the Newton-Raphson algorithm, which got its name from the two inventors: Isaac Newton and Joseph Raphson. Let $L(\boldsymbol{\beta})$ be the log-likelihood where $\boldsymbol{\beta} = (\beta_1, \cdots, \beta_p)$ is a vector of p unknown parameters that needs to be estimated. We denote vector $\mathbf{u^T}$ as the first-order partial derivatives of $L(\boldsymbol{\beta})$ with respect to each $\beta_j,\ j = 1, \cdots, p$ where $\mathbf{u}^{\mathbf{T}} = \frac{\partial L(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = \left(\frac{\partial L(\boldsymbol{\beta})}{\partial \beta_1}, \cdots, \frac{\partial L(\boldsymbol{\beta})}{\partial \beta_p}\right)$. We also denote $\mathbf{H}$ as a $p \times p$ Hessian matrix consisting of second-order partial derivatives of $L(\boldsymbol{\beta})$ where each entry has the form $h_{ij} = \frac{\partial^2 L(\boldsymbol{\beta})}{\partial \beta_i \partial \beta_j}, i = 1, \cdots, p,\ j = 1, \cdots, p$. According to second-order Taylor series expansion, $L(\boldsymbol{\beta})$ can be approximated as:

$$L(\boldsymbol{\beta}) \approx L(\boldsymbol{\beta}^{(s)}) + \mathbf{u}^{\mathbf{T}(s)}(\boldsymbol{\beta} - \boldsymbol{\beta}^{(s)}) + \frac{1}{2}(\boldsymbol{\beta} - \boldsymbol{\beta}^{(s)})'\mathbf{H}^{(s)}(\boldsymbol{\beta} - \boldsymbol{\beta}^{(s)}) \tag{1.3.3}$$

where $\boldsymbol{\beta}^{(s+1)} = \boldsymbol{\beta}^{(s)} - (\mathbf{H}^{(s)})^{-1}\mathbf{u}^{(s)}$ is the approximation to the root evaluated at the $(s+1)^{th}$ iteration by solving the first order partial derivative of second-order Taylor series expansion of $L(\boldsymbol{\beta})$, that is, $\frac{\partial L(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} \approx \mathbf{u}^{(s)} + \mathbf{H}^{(s)}(\boldsymbol{\beta} - \boldsymbol{\beta}^{(s)}) = 0$. This iteration is repeated until the difference between $L(\boldsymbol{\beta}^{s})$ and $L(\boldsymbol{\beta}^{s-1})$ is negligible, that is where the optimum value of $L(\boldsymbol{\beta})$ is reached.

Analogous to the Newton-Raphson algorithm, Fisher's Scoring is also based on the second order Taylor series expansion of the likelihood function. Instead of using the Hessian matrix directly in the approximation, it uses the $p \times p$ Fisher's information matrix which is the negative expectation of the second-order partial derivatives of $L(\boldsymbol{\beta})$ where we denote it as $\boldsymbol{\iota}$ and $\iota_{ij} = -E\left(\frac{\partial^2 L(\boldsymbol{\beta})}{\partial \beta_i \partial \beta_j}\right), i = 1, \cdots, p,\ j = 1, \cdots, p$. By substituting $\mathbf{H}^{(s)}$ with $-\boldsymbol{\iota}^{(s)}$ in (1.3.3), $\boldsymbol{\beta}^{(s+1)}$ at the $(s+1)^{th}$ iteration can be evaluated as $\boldsymbol{\beta}^{(s+1)} = \boldsymbol{\beta}^{(s)} + (\boldsymbol{\iota}^{(s)})^{-1}\mathbf{u}^{(s)}$

The two optimization algorithms discussed above require calculation of the second order derivatives of $L(\boldsymbol{\beta})$ which can be tremendously computationally expensive when $L(\boldsymbol{\beta})$ has a complex form. Alternatively, algorithms based on the approximation of the second-order partial derivative such as quasi-Newton BFGS [Broyden, 1970, Fletcher, 1970], L-BFGS-B [RH. Byrd and Zhu, 1995], Dual quasi-Newton with dogleg strategy [Dennis and Mei, 1979] and BHHH [E. Berndt and Hausman, 1974] are actively involved to solve the unconstrained nonlinear optimization problems. Other algorithms based on a different mechanism, such as Nelder-Mead method [Nelder and Mead, 1965], which is a heuristic search method to minimize an objective function in a multi-dimensional space, and the conjugate gradient method [Fletcher and Reeves, 1964], which provides numerical solution in a sparse system, are also implemented to optimize the likelihood function.

1.3.3 Software Implementation

Currently, there are several software packages capable of fitting ordinal models. In SAS version 9.2, `PROC LOGISTIC` procedure is capable of fitting the cumulative logit model and `PROC NLMIXED` procedure provides the flexibility to fit all types of ordinal models. It is worth mentioning that the optimization techniques in these two procedures are different, which may yield slightly different results under certain situations. The `PROC LOGISTIC` procedure implements the Newton-Raphson algorithm and Fisher's Scoring algorithm (default) while `PROC NLMIXED` has a larger selection of optimization methods including Dual Quasi-Newton (default), Conjugate Gradient methods and Nelder-Mead simplex method, *etc.* For R version 2.13.1, the package `VGAM` [T.W.Yee, 2013] is capable of fitting different types of ordinal models by creating the class of Vector Generalized Linear Models (VGLMs) using the `vglm` function. The default and currently the only optimization method implemented in the `vglm` function is Iteratively Reweighted Least Squares (IRLS). In addition, we also wrote our own code for fitting different types of ordinal models. For the cumulative logit model, we optimize the model using the nonlinear constrained optimization method incorporated in the R package `alabama` [Varadhan, 2011]. For other types of ordinal models, the parameter estimates are obtained using the Newton-Raphson algorithm in the universal nonlinear optimization function `nlm`.

1.4 NIMH Schizophrenia Example

The Schizophrenia data described in [Hedeker and Gibbons, 2006] is from a collaborative study conducted at the National Institute of Mental Health. In this longitudinal study, a total of 1603 observations were obtained from 437 patients. For now, we tentatively ignore the correlations between multiple measurements from the same patient and assume the independence of observations. The outcome was measured as the Inpatient Multidimensional Psychiatric Scale (IMPS)[Lorr and Klett, 1966], which was a eight-category scale ranging from normal (0) to extremely ill (7) that assessed the severity of illness. Hedeker and Gibbons [2006] modified the original scale by aggregating several categories and summarizing it into a four-category measurement ranging from 1 to 4 with 1) normal or borderline mentally ill, 2) mildly or moderately ill, 3) markedly ill, and 4) severely and most extremely ill. Table 1.1 summarizes the distribution of the modified IMPS score. It is of interest to study whether the intervention and time of measurements have a significant impact on the change of IMPS score. To explore this, we fit cumulative logit, adjacent-category, backward continuation ratio and forward continuation ratio ordinal models. To evaluate the concordance of parameter estimates from different computational tools, we wrote our own R code and compared our results to the results from the R package `VGAM` and to SAS. Tables 1.2, 1.5, 1.8 and 1.11 present the parameter estimates from our code. Tables 1.3, 1.6, 1.9 and 1.12 present the parameter estimates from the R package `VGAM`. Tables 1.4, 1.7, 1.10 and 1.13 present the parameter estimates from SAS. The parameter estimates from the three approaches achieve good concordance, from which we can draw conclusions that the drug and time interaction is significant with an extremely small p-value, which can be interpreted as the distribution of the IMPS score changes more dramatically in intervention group compared to that in the control group. As the interaction is significant, the main effects can not be interpreted according to p-value directly. Instead, we interpret the drug and time main effects based on contrasts. The drug effect β_{drug} represents the difference of IMPS score distributions between two groups at baseline (week 0) where the proportion of normal, mildly, markedly and severely ill subjects are similar in two groups. The time effect β_{time} describes the significant change of IMPS score distribution as the study continuous. As the drug effect $\beta_{drug} = 0$ illustrates the changes are the same for the intervention and control groups.

Table 1.1: Summary of IMPS score(modified) in the NIMH Schizophrenia data

IMPS Score	1	2	3	4	Total
	190	474	412	527	1603

Table 1.2: R code for fitting the cumulative logit ordinal model using NIMH Schizophrenia data

Parameter	Estimate	SE	DF	t-value	P-value
(Intercept):1	-3.81	0.189	1597	-20.10	<0.001*
(Intercept):2	-1.76	0.170	1597	-10.37	<0.001*
(Intercept):3	-0.42	0.163	1597	-2.59	0.010*
tx	0.00	0.188	1597	0.00	1.00
sweek	0.54	0.111	1597	4.85	<0.001*
tx$\times$*sweek*	0.75	0.127	1597	5.89	<0.001*
-2logL	3756.20				

Table 1.3: R package `VGAM` for fitting the cumulative logit ordinal model using NIMH Schizophrenia data

Parameter	Estimate	SE	DF	t-value	P-value
(Intercept):1	-3.81	0.193	1597	-19.77	<0.001*
(Intercept):2	-1.76	0.174	1597	-10.12	<0.001*
(Intercept):3	-0.42	0.167	1597	-2.53	0.012*
tx	0.00	0.192	1597	0.00	1.00
sweek	0.54	0.111	1597	4.84	<0.001*
tx$\times$*sweek*	0.75	0.128	1597	5.87	<0.001*
-2logL	3756.20				

Table 1.4: SAS output for fitting the cumulative logit ordinal model using NIMH Schizophrenia data

Parameter	Estimate	SE	DF	t-value	P-value
(Intercept):1	-3.81	0.190	1597	-20.05	<0.001*
(Intercept):2	-1.76	0.170	1597	-10.34	<0.001*
(Intercept):3	-0.42	0.164	1597	-2.58	0.010*
tx	0.00	0.188	1597	0.00	1.00
sweek	0.54	0.111	1597	4.84	<0.001*
tx$\times$*sweek*	0.75	0.128	1597	5.88	<0.001*
-2logL	3756.20				

Table 1.5: R code for fitting the adjacent categories ordinal model using NIMH Schizophrenia data

Parameter	Estimate	SE	DF	t-value	P-value
(Intercept):1	2.23	0.154	1597	14.50	<0.001*
(Intercept):2	0.81	0.133	1597	6.14	<0.001*
(Intercept):3	0.85	0.120	1597	7.06	<0.001*
tx	-0.05	0.126	1597	-0.40	0.692
sweek	-0.36	0.070	1597	-5.18	<0.001*
tx$\times$*sweek*	-0.40	0.081	1597	-4.97	<0.001*
-2logL	3748.63				

Table 1.6: R package `VGAM` for fitting the adjacent categories ordinal model using NIMH Schizophrenia data

Parameter	Estimate	SE	DF	t-value	P-value
(Intercept):1	2.23	0.154	1597	14.50	<0.001*
(Intercept):2	0.81	0.133	1597	6.14	<0.001*
(Intercept):3	0.85	0.120	1597	7.06	<0.001*
tx	-0.05	0.126	1597	-0.40	0.692
sweek	-0.36	0.070	1597	-5.18	<0.001*
tx$\times$*sweek*	-0.40	0.081	1597	-4.97	<0.001*
-2logL	3748.63				

Table 1.7: SAS output for fitting the adjacent categories ordinal model using NIMH Schizophrenia data

Parameter	Estimate	SE	DF	t-value	P-value
(Intercept):1	2.23	0.154	1597	14.50	<0.001*
(Intercept):2	0.81	0.133	1597	6.14	<0.001*
(Intercept):3	0.85	0.120	1597	7.06	<0.001*
tx	-0.05	0.126	1597	-0.40	0.692
sweek	-0.36	0.070	1597	-5.18	<0.001*
tx$\times$*sweek*	-0.40	0.081	1597	-4.97	<0.001*
-2logL	3748.63				

Table 1.8: R code for fitting the backward continuation ratio ordinal model using NIMH Schizophrenia data

Parameter	Estimate	SE	DF	t-value	P-value
(Intercept):1	2.85	0.184	1597	15.49	<0.001*
(Intercept):2	0.96	0.160	1597	5.97	<0.001*
(Intercept):3	0.38	0.147	1597	2.59	0.010
tx	-0.09	0.166	1597	-0.56	0.574
sweek	-0.55	0.095	1597	-5.73	<0.001*
tx $\times sweek$	-0.54	0.110	1597	-4.95	<0.001*
-2logL	3745.77				

Table 1.9: R package `VGAM` for fitting the backward continuation ratio ordinal model using NIMH Schizophrenia data

Parameter	Estimate	SE	DF	t-value	P-value
(Intercept):1	2.85	0.184	1597	15.49	<0.001*
(Intercept):2	0.96	0.160	1597	5.97	<0.001*
(Intercept):3	0.38	0.147	1597	2.59	0.010
tx	-0.09	0.166	1597	-0.56	0.574
sweek	-0.55	0.095	1597	-5.73	<0.001*
tx $\times sweek$	-0.54	0.110	1597	-4.95	<0.001*
-2logL	3745.77				

Table 1.10: SAS output for fitting the backward continuation ratio ordinal model using NIMH Schizophrenia data

Parameter	Estimate	SE	DF	t-value	P-value
(Intercept):1	2.85	0.184	1597	15.55	<0.001*
(Intercept):2	0.96	0.160	1597	6.00	<0.001*
(Intercept):3	0.38	0.147	1597	2.60	0.010
tx	-0.09	0.166	1597	-0.56	0.574
sweek	-0.55	0.095	1597	-5.74	<0.001*
tx $\times sweek$	-0.54	0.110	1597	-4.95	<0.001*
-2logL	3745.77				

Table 1.11: R code for fitting the forward continuation ratio ordinal model using NIMH Schizophrenia data

Parameter	Estimate	SE	DF	t-value	P-value
(Intercept):1	-3.42	0.173	1597	-19.80	<0.001*
(Intercept):2	-1.76	0.156	1597	-11.35	<0.001*
(Intercept):3	-0.99	0.152	1597	-6.55	<0.001*
tx	-0.02	0.168	1597	-0.12	0.906
sweek	0.42	0.096	1597	4.34	<0.001*
tx$\times$*sweek*	0.64	0.111	1597	5.81	<0.001*
-2logL	3785.04				

Table 1.12: R package `VGAM` for fitting the forward continuation ratio ordinal model using NIMH Schizophrenia data

Parameter	Estimate	SE	DF	t-value	P-value
(Intercept):1	-3.42	0.175	1597	-19.60	<0.001*
(Intercept):2	-1.76	0.158	1597	-11.18	<0.001*
(Intercept):3	-0.99	0.155	1597	-6.42	<0.001*
tx	-0.02	0.170	1597	-0.12	0.907
sweek	0.42	0.096	1597	4.33	<0.001*
tx$\times$*sweek*	0.64	0.111	1597	5.79	<0.001*
-2logL	3785.04				

Table 1.13: SAS for fitting the forward continuation ratio ordinal model using NIMH Schizophrenia data

Parameter	Estimate	SE	DF	t-value	P-value
(Intercept):1	-3.42	0.173	1597	-19.80	<0.001*
(Intercept):2	-1.76	0.156	1597	-11.35	<0.001*
(Intercept):3	-0.99	0.152	1597	-6.55	<0.001*
tx	-0.02	0.167	1597	-0.12	0.906
sweek	0.42	0.096	1597	4.34	<0.001*
tx$\times$*sweek*	0.64	0.111	1597	5.81	<0.001*
-2logL	3785.00				

Chapter 2

Regularization Methods for High-dimensional Data

In this chapter, we introduce statistical learning algorithms that are useful for prediction and feature selection problems for high-dimensional data where the number of covariates p is much larger than the number of samples n. This type of data is particularly prevalent in computational biology where high-throughput genomic technologies plays an increasingly important role in medical research and clinical practice. The statistical learning algorithms produce penalized estimates of coefficients for 'important' covariates while shrinking the rest to be exactly zero. This approach has been demonstrated to have superior performance over the traditional hypothesis-based variable selection methods such as Best subsets, Forward Selection and Backward Elimination, *etc.* It also has better performance in aspects of prediction and classification accuracy, stability and consistency for feature selection in data with complex patterns.

The rest of Chapter 2 is organized as follows: In Section 2.1, we briefly review several existing regularization methods that are applicable for continuous responses. In Section 2.2, we present regularization methods that are suitable for dichotomous responses. In Section 2.3, we discuss the optimization algorithm to solve the regularization path in LASSO. Some comparisons between the LASSO and Forward Stagewise methods are discussed in Section 2.4.

2.1 Regularization Methods for Continuous Response

In high-dimensional data, the explosion of dimensionality allows much more information to be examined simultaneously. In the meantime, the curse of dimensionality raises challenges rarely occurred in low-dimensional settings, such as: data sparsity, high-volatility, and nonlinearity. Variable selection and assessment is an important first step to reduce dimensionality such that fewer important factors can be implemented in a general statistical framework. Some traditional variable selection methodologies including Best subsets, Forward Selection and Backward Elimination *etc.* often fail to provide feasible and stable results due to strong assumptions on covariate independence as well as problematic discrete procedures which yield high variability. The regularization methods, also known as penalized models which trade off unbiasedness for lower variability, have demonstrated superior performance in analyzing high-dimensional data. We first review the regularization model framework in the linear model setting.

Consider a linear model of the form (2.1.1)

$$y_i = \mathbf{x_i}^T \boldsymbol{\beta} + \epsilon_i \tag{2.1.1}$$

where y_i is the response for the i^{th} observation, $\mathbf{x_i}^T = (1, x_{i1}, \cdots, x_{ip})$ is a vector of the intercept and p covariates, $\boldsymbol{\beta} = (\beta_0, \beta_1, \cdots, \beta_p)$ is the vector of coefficients associated with the intercept and covariates, and ϵ_i is the error term assumed to follow a normal distribution with mean 0 and variance σ^2. The residual sum of square (RSS) in a linear model is defined as (2.1.2). It can be shown that the ordinary least squares (OLS) estimator of $\boldsymbol{\beta}$ is the solution that minimizes the RSS. The ordinary least squares estimator has some appealing statistical properties. First, it is also the maximum likelihood estimator for a linear regression model. Second, it is an unbiased estimator where $E(\hat{\boldsymbol{\beta}}) = \boldsymbol{\beta}$.

$$RSS(\boldsymbol{\beta}) = \sum_{i=1}^{n} (y_i - \mathbf{x_i}^T \boldsymbol{\beta})^2 \tag{2.1.2}$$

In a high-dimensional setting or when data are highly correlated, retaining the unbiasedness of the estimator is either infeasible or could result in very high variability. Fan and Li [2001] proposed penalized least squares as shown in (2.1.3)

$$Q(\boldsymbol{\beta}) = \sum_{i=1}^{n} (y_i - \mathbf{x_i}^T \boldsymbol{\beta})^2 + p_\lambda(\boldsymbol{\beta}) \tag{2.1.3}$$

where $p_\lambda(\cdot)$ is the penalty function and λ is a tuning parameter that regulates the amount of shrinkage. Different penalty functions lead to different penalized models. The well known shrinkage procedure ridge regression [Hoerl and Kennard, 1970], which is also known as

L_2-norm penalized regression model, has a penalty function $p_\lambda(\boldsymbol{\beta}) = \lambda \sum_j \beta_j^2$. The ridge regression produces the penalized estimates for all coefficients by sacrificing a little bit unbiasedness in return for smaller variance. As pointed out by Myers [1990], ridge regression is particularly useful for solving the multicollinearity problem in linear regression. However, ridge regression neither performs variable selection nor builds a parsimonious model since it cannot shrink the coefficient estimates to be exactly zero. Alternatively, Tibshirani [1996] proposed the famous Least Absolute Shrinkage and Selection Operator (LASSO) method which sets the penalty function to be the L_1 norm of the coefficients where $p_\lambda(\boldsymbol{\beta}) = \lambda \sum_j |\beta_j|$. LASSO has been demonstrated to be useful in variable selection since it can shrink the coefficient estimates associated with unimportant covariates to be exactly zero while producing penalized estimates for the few important covariates. The penalized estimates also yield better prediction accuracy in terms of smaller prediction error. In addition, Zou and Hastie [2005] introduced the elastic net penalty $p_\lambda(\boldsymbol{\beta}) = \lambda[(1-\alpha)\sum_j |\beta_j| + \alpha \sum_j \beta_j^2]$ which improves the stability of the LASSO when exposed to a group of strongly correlated predictors. Moreover, Fan and Li [2001] advocated a penalty function from a different mechanism called smoothly clipped absolute deviation (SCAD) with a form $p'_\lambda(\beta) = \lambda\big(I(\beta \leq \lambda) + \frac{(\alpha\lambda-\beta)_+}{(\alpha-1)\lambda} I(\beta > \lambda)\big)$ where α and λ are the unknown parameters determined by cross-validation. This method is shown to be effective without creating excessive biases especially in situations where the noise to signal ratio is low.

In this Section we concentrate on the L_1 penalized regression model LASSO and two other popular algorithms: Least Angle Regression [Efron et al., 2004] (better known as 'LAR') and incremental forward stagewise regression [Hastie et al., 2007] which were also proposed for solving the penalized least squares problem when the response is continuous.

2.1.1 LASSO

The LASSO, also known as the L_1 penalized regression model proposed by Tibshirani [1996], is a widely used smooth optimization technique for building a parsimonious model from high-dimensional data. It produces penalized estimates for the unknown parameters $\boldsymbol{\beta}$ as shown in (2.1.4)

$$\hat{\boldsymbol{\beta}}^{LASSO} = \ argmin\{\sum_{i=1}^{n}(y_i - \beta_0 - \mathbf{x_i}^T\boldsymbol{\beta})^2 + \lambda\sum_{j=1}^{p}|\beta_j|\} \tag{2.1.4}$$

where the tuning parameter λ controls the amount of shrinkage. The penalized estimates are equivalent to the ordinary least squares estimates if $\lambda = 0$ and the penalized estimates are all 0 if $\lambda \to \infty$. Therefore, the choice of the turning parameter λ can be thought of as a way to determine the number of important predictors. As there is no standard criteria associated with the choice of λ, λ can be the one associated with optimum model fitting criteria (AIC, BIC) or that leads to the smallest cross-validation error decrease.

The rationale behind the LASSO can be illustrated by Figure 2.1 reproduced from [Tibshirani, 1996] which also helps explain the mechanism of variable selection. Figure 2.1 illustrates a situation when there are two covariates $\mathbf{x_1}, \mathbf{x_2}$ in the linear regression model. The blue diamond area is defined by the penalty function $|\beta_1|+|\beta_2| \leq t$, where t is a scale determined by the turning parameter λ. The red contours represent the squared error between the penalized estimate and the ordinary least squares estimate. To illustrate, we assume the data is normalized so that $\mathbf{x_1}^T\mathbf{x_2} = 0$. We define the squared error as follows (2.1.5)

$$\begin{aligned}
& \sum_{i=1}^{n}(\mathbf{x_i}\hat{\boldsymbol{\beta}}^{PLS} - \mathbf{x_i}\hat{\boldsymbol{\beta}}^{OLS})^2 \\
= & \sum_{i=1}^{n}(\hat{\boldsymbol{\beta}}^{PLS} - \hat{\boldsymbol{\beta}}^{OLS})^T\mathbf{x_i}^T\mathbf{x_i}(\hat{\boldsymbol{\beta}}^{PLS} - \hat{\boldsymbol{\beta}}^{OLS}) \\
= & (\hat{\beta}_1^{PLS} - \hat{\beta}_1^{OLS}, \hat{\beta}_2^{PLS} - \hat{\beta}_2^{OLS}) \begin{pmatrix} \mathbf{x_1}^T\mathbf{x_1} & 0 \\ 0 & \mathbf{x_2}^T\mathbf{x_2} \end{pmatrix} \begin{pmatrix} \hat{\beta}_1^{PLS} - \hat{\beta}_1^{OLS} \\ \hat{\beta}_2^{PLS} - \hat{\beta}_2^{OLS} \end{pmatrix} \\
= & (\hat{\beta}_1^{PLS} - \hat{\beta}_1^{OLS})^2\mathbf{x_1}^T\mathbf{x_1} + (\hat{\beta}_2^{PLS} - \hat{\beta}_2^{OLS})^2\mathbf{x_2}^T\mathbf{x_2} = k \qquad (2.1.5)
\end{aligned}$$

where $\mathbf{x_1}^T\mathbf{x_1} = \sum_{i=1}^{n} x_{i1}^2$,$\mathbf{x_2}^T\mathbf{x_2} = \sum_{i=1}^{n} x_{i2}^2$ and k is a constant. The last line of equation (2.1.5) defines several ellipses that are centered at $\hat{\boldsymbol{\beta}} = (\hat{\beta}_1^{OLS}, \hat{\beta}_2^{OLS})$. As the penalized estimates have to satisfy both equation (2.1.5) and the penalty function, a possible situation is when the contour hits the corner of the diamond which yields an exact zero estimate for one coefficient. From here, it is not difficult to generalize to a higher dimensional scenario when multiple coefficients are shrunk to be exactly zero.

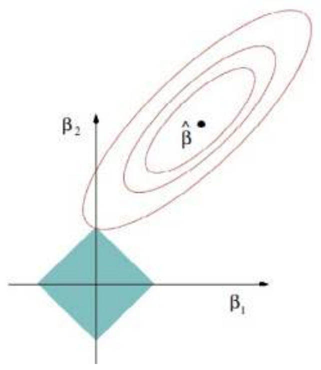


Figure 2.1: Estimation picture for the LASSO

2.1.2 Forward Stagewise Method

The forward stagewise method is a greedy procedure similar to forward stepwise but more cautious. It was historically known as an 'inefficient' procedure since the model is updated by a very small incremental step at each iteration. Recently, it is gained enormous attention when Hastie et al. [2007] linked forward stagewise to a version of boosting proposed by Schapire et al. [1998] for linear models. This 'slow fitting' turns out to be quite competitive in terms of variable selection stability and prediction accuracy. In fact, Efron et al. [2004] showed the forward stagewise profile can be similar or even identical to the LASSO path under certain conditions and thus can be used to solve the penalized regression problem.

The forward stagewise method for linear regression is as follows:

1. Standardize the covariates so that each has mean 0 and unit norm. At the initial step, set the residual vector $\mathbf{r}$ to $\mathbf{y}$ and $\boldsymbol{\beta} = (\beta_1, \cdots, \beta_p) = \mathbf{0}$.

2. Calculate the correlation between $\mathbf{x_j}$ for $j = 1, \cdots, p$ and the current residual $\mathbf{r}$. Select the x_j most correlated with r.

3. Update the corresponding coefficient β_j with $\beta_j \leftarrow \beta_j + \epsilon \cdot sign < \mathbf{x_j}, \mathbf{r} >$ where ϵ is a small amount, *e.g.* $\epsilon = 10^{-4}$ and $< \mathbf{x_j}, \mathbf{r} >$ represents the correlation between $\mathbf{x_j}$ and r.

   Update the residual to $\mathbf{r} \leftarrow \mathbf{r} - \epsilon \cdot sign < \mathbf{x_j}, \mathbf{r} > \mathbf{x_j}$.

4. Repeat steps 2 and 3 many times until r is uncorrelated or at least not highly correlated with any of the covariates.

The mechanism of forward stagewise can be explained by Figure 2.2 reproduced from Hastie et al. [2009]. Recall in the linear regression model, the OLS estimator $\hat{\boldsymbol{\beta}}$ is the estimate associated with minimal residual sum-of-squares (RSS)(2.1.2). In projection presentation, suppose response y is projected onto the space spanned by covariates x_1 and x_2. The residual $r = y - \hat{y}$ measures the distance between y and the predictor space. The RSS is achieved when the residual r is orthogonal to both x_1 and x_2, in other words, r is uncorrelated with all the covariates in model. The mechanism can be easily generalized to a high-dimensional scenario. In the forward stagewise procedure discussed above, the update of coefficients $\boldsymbol{\beta}$ and residual r stops when r is uncorrelated, or at least not highly correlated with any of the covariates in the model, and can be interpreted as when the minimum RSS is reached.

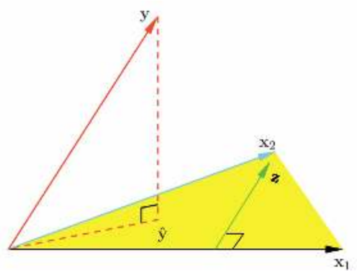


Figure 2.2: Least square projection in linear regression model

2.1.3 LAR

LAR, which stand for 'Least Angle Regression' proposed by Efron et al. [2004], is considered as a more 'democratic' version of the forward stagewise method. After selecting the predictor that is most correlated with the residual, rather than taking the full step towards that direction, LAR takes the largest step possible in that direction until some other predictor has reached the correlation with the current residual. Then the two predictors are moved in the joint least squares direction until the third predictor achieves the amount of correlation with the current residual and joins the selected group. This procedure is repeated many times until the correlation of each predictor to the current residual is smaller than a threshold. This new comer is known to be closely connected to LASSO and provides a more efficient solution to the penalized least squares problem.

2.2 Regularization Methods for Dichotomous Responses

In this section, we move on to discuss regularization methods implemented for the logistic regression model with high-dimensional predictors by first providing the statistical framework for logistic regression in the traditional setting. Consider a logistic regression of the form (2.2.1)

$$\log \frac{P(Y_i = 1|\mathbf{x})}{P(Y_i = 0|\mathbf{x_i})} = \log \frac{\pi(\mathbf{x_i})}{1 - \pi(\mathbf{x_i})} = \alpha + \mathbf{x_i}^T \boldsymbol{\beta} \tag{2.2.1}$$

where $y_i = 1$ indicates 1 success and $y_i = 0$ indicates failure. For each observation i, we denote Y_i being the total number of successes and $1 - Y_i$ being the total number of failures. Therefore, Y_i follows a binomial distribution with mass function $f(Y_i; 1, \pi_i) = \binom{1}{Y_i}\pi_i^{Y_i}(1-\pi_i)^{1-Y_i}$. $\mathbf{x_i}^T = (x_{i1}, \cdots, x_{ip})$ is a vector of p covariates, α represents the intercept, and $\boldsymbol{\beta} = (\beta_1, \cdots, \beta_p)$ is a vector of coefficients associated with the covariates. The maximum likelihood approach is implemented to obtain the estimates of the unknown parameters in the model. The likelihood function for the logistic regression model can be written as:

$$\begin{aligned} L(\alpha, \boldsymbol{\beta}|\mathbf{x_i}) &= \prod_{i=1}^{N} \pi(\mathbf{x_i})^{Y_i}(1-\pi(\mathbf{x_i}))^{1-Y_i} \\ &= \prod_{i=1}^{N} \left(\frac{\exp(\alpha + \mathbf{x_i}^T\boldsymbol{\beta})}{1+\exp(\alpha + \mathbf{x_i}^T\boldsymbol{\beta})}\right)^{Y_i} \left(\frac{1}{1+\exp(\alpha + \mathbf{x_i}^T\boldsymbol{\beta})}\right)^{1-Y_i}. \end{aligned} \tag{2.2.2}$$

Correspondingly, the log-likelihood can be written as:

$$\log L(\alpha, \boldsymbol{\beta}|\mathbf{x_i}) = \sum_{i=1}^{N} Y_i \log \pi(\mathbf{x_i}) + (1-Y_i)\log(1-\pi(\mathbf{x_i})). \tag{2.2.3}$$

It is not hard to show that $\log L(\alpha, \boldsymbol{\beta}|\mathbf{x_i})$ in (2.2.3) is a concave function and its negative, $-\log L(\alpha, \boldsymbol{\beta}|\mathbf{x_i})$ is a convex function.

2.2.1 LASSO for Logistic Regression

In a high-dimensional setting where $N \ll p$, maximum likelihood estimation is no longer feasible due to singularities in the Hessian matrix. The penalized model is again proposed and demonstrated to be useful in this case. Recall for the linear regression model with high-dimensional covariates the penalized estimator is the one associated with the optimum value of the penalized least squares (2.1.3), which consists of the residual sum of squares and the penalty function. Notice the residual sum of squares is the kernel of the likelihood in the linear model case. When the response is discrete, minimizing the residual sum of square is not reasonable, so a modified penalized model that maximizes the penalized log-likelihood function was proposed [Wu et al., 2009]. The penalized log-likelihood consists of the log-likelihood and the penalty function

$$\log Q(\alpha, \boldsymbol{\beta}|\mathbf{x_i}) = \sum_{i=1}^{N} Y_i \log \pi_i(\mathbf{x_i}) + (1 - Y_i)\log(1 - \pi_i(\mathbf{x_i})) - \lambda \sum_{j=1}^{p} |\beta_j| \tag{2.2.4}$$

2.2.2 Forward Stagewise for Logistic Regression

The forward stagewise method for logistic regression inherits the mechanism from the linear model to update one coefficient at a time using a small incremental amount and thus obtains the penalized estimate. Recall for the forward stagewise algorithm implemented for the linear regression model, the coefficient to be updated at each iteration is selected based on the correlation between the covariates and the current residual. As mentioned previously, since the residual is no longer an appropriate measurement for a discrete variable, a modified forward stagewise method where the coefficient to be updated is selected based on the gradient of the likelihood function is therefore proposed for models having a discrete response. In addition, to determine the direction for updating the coefficient requires calculation of the second-order derivatives of the likelihood function at every step, which is computationally expensive given hundreds of thousands iterations needed before the likelihood function reaches the optimum. Hastie et al. [2007] showed the expanded representation of the LASSO problem produces an efficient version of forward stagewise which is able to avoid the cumbersome computation of second-order derivatives of the likelihood function. In the expanded setting, a negative copy of the covariates is created, thus the new predictor space is $\tilde{\mathbf{X}} = (\mathbf{X}, -\mathbf{X})$. Correspondingly, the coefficients are expanded to $\boldsymbol{\beta} = (\beta_1, \cdots, \beta_p, \beta_{p+1}, \cdots, \beta_{2p})$. For each iteration, only one β_j is updated with a small incremental amount and the Karush-Kuhn-Tucker condition (proof can be found in Appendix of [Hastie et al., 2007]) ensures β_j and β_{j+p} representing the same covariate x_j cannot be selected simultaneously in the regularization path. Therefore, the penalized estimate of $\boldsymbol{\beta}$ is obtained by subtracting the β_j associated with $-x_j$ from that associated with x_j, that is $\boldsymbol{\beta} = (\beta_1 - \beta_{p+1}, \beta_2 - \beta_{p+2}, \cdots, \beta_p - \beta_{2p})$ is the final solution.

The forward stagewise regression for logistic model works as follows:

1. Let $\tilde{\mathbf{X}} = (\mathbf{X}, -\mathbf{X})$ and standardize the covariates so that each has mean 0 and unit norm. In initial step, set $\boldsymbol{\beta} = (\beta_1, \cdots, \beta_p, \beta_{p+1}, \cdots, \beta_{2p}) = \mathbf{0}$.

2. Calculate the first-order derivative of $-\log L(\alpha, \boldsymbol{\beta}|\mathbf{x_i})$ with respect to β_j evaluated at current estimate $\boldsymbol{\beta} = \boldsymbol{\beta}^{(s)}$. Find the predictor x_j with the largest negative gradient element.

3. Update the corresponding coefficient β_j with $\beta_j \leftarrow \beta_j + \epsilon$ to yield the new estimate $\boldsymbol{\beta} = \boldsymbol{\beta}^{(s+1)}$ where ϵ is a small amount, *e.g.* $\epsilon = 10^{-4}$.

4. Repeat steps 2 and 3 many times.

There is no standard stopping criteria in the forward stagewise method for logistic regression. We implemented the criteria to stop the iteration if the difference between the adjacent log-likelihood is smaller than a given value, that is $|\log L(\alpha, \boldsymbol{\beta}|\mathbf{x_i})|_{\boldsymbol{\beta}=\boldsymbol{\beta}^{(s)}} - \log L(\alpha, \boldsymbol{\beta}|\mathbf{x_i})|_{\boldsymbol{\beta}=\boldsymbol{\beta}^{(s+1)}}| < \delta$. The forward stagewise method stops if δ is smaller than a certain value.

2.3 Coordinate Descent for LASSO Regularization Paths

It is challenging to solve the convex optimization problem with an inequality constraint to obtain the penalized estimates in LASSO. In the past, quadratic programming was often conducted for such a scenario. Friedman et al. [2007] demonstrated the 'one-at-a-time' coordinate descent algorithm, which was mistakenly considered to be too simple for convex problems, had competitive and efficient performance for certain types of penalty functions including the LASSO and elastic net. Here, we briefly discuss this algorithm which has been implemented in the R package `glmnet` [Friedman et al., 2013].

In the linear model, the 'Lagrange' version of penalized least squares for LASSO can be written as (2.3.1) which is equivalent to (2.1.4)

$$Q(\boldsymbol{\beta}) = \frac{1}{2}\sum_{i=1}^{n}\big(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j\big)^2 + \lambda\sum_{j=1}^{p}|\beta_j|. \tag{2.3.1}$$

If we assume the pairwise independence of each covariate, by standardizing the covariates, the LASSO estimate can be simplified to a soft thresholded version of the least squares estimate proposed by Donoho and Johnstone [1995]. Under the independence assumption, (2.3.1) can be further re-expressed as:

$$\begin{aligned} Q(\boldsymbol{\beta}) &= \frac{1}{2}\sum_{j=0}^{p}\sum_{i=1}^{n} x_{ij}^2\big(\hat{\beta}_j^{LASSO} - \hat{\beta}_j^{OLS}\big)^2 + \lambda\sum_{j=0}^{p}|\hat{\beta}_j^{LASSO}| \\ &= \frac{1}{2}\sum_{j=0}^{p}\big(\hat{\beta}_j^{LASSO} - \hat{\beta}_j^{OLS}\big)^2 + \lambda\sum_{j=0}^{p}|\hat{\beta}_j^{LASSO}|,\ \sum_{i=1}^{n} x_{ij}^2 = 1 \end{aligned} \tag{2.3.2}$$

To minimize the penalized least squares with a form of (2.3.2), the first step is to take the first-order derivative of $Q(\boldsymbol{\beta})$ with respect to each β_j and this gradient can be written as:

$$\frac{\partial Q(\boldsymbol{\beta})}{\partial \beta_j} = \hat{\beta}_j^{LASSO} - \hat{\beta}_j^{OLS} \pm \lambda = 0. \tag{2.3.3}$$

From there the penalized estimate can be written as a function of the least squares estimate and the tuning parameter. As discussed in Friedman et al. [2007], this soft threshold can be depicted as

$$\hat{\beta}_j^{LASSO} = \begin{cases} \hat{\beta}_j^{OLS} - \lambda & \text{if } \hat{\beta}_j^{OLS} > 0 \text{ and } \lambda < |\hat{\beta}_j^{OLS}| \\ \hat{\beta}_j^{OLS} + \lambda & \text{if } \hat{\beta}_j^{OLS} < 0 \text{ and } \lambda < |\hat{\beta}_j^{OLS}| \\ 0 & \text{if } \hat{\beta}_j^{OLS} \geq 0 \end{cases} \tag{2.3.4}$$

for the general case where correlation is allowed between covariates. Suppose $\beta_k(\lambda)$, $k \neq j$ is a set of correlated covariates with their estimates associated with λ. Then the penalized least squares (2.3.1) can be modified correspondingly as

$$Q(\boldsymbol{\beta}) = \frac{1}{2} \sum_{i=1}^{n} \big(y_i - \sum_{k \neq j} x_{ik} \beta_k(\lambda) - \sum_j x_{ij} \beta_j\big)^2 + \lambda \sum_j |\beta_j| + \lambda \sum_{k \neq j} |\beta_k|. \tag{2.3.5}$$

Again, to minimize (2.3.5), we take the first-order derivative with respect to β_j and the gradient can be written as:

$$\frac{\partial Q(\boldsymbol{\beta})}{\partial \beta_j} = \sum_{i=1}^{n}(y_i - \sum_{k \neq j} x_{ik}\beta_k(\lambda))x_{ij} - \beta_j \pm \lambda = 0. \tag{2.3.6}$$

Solving (2.3.6), β_j can be written as a function of $\beta_k(\lambda)$, wherein we denote λ as the soft threshold function $S(\cdot)$. Therefore, for each tuning parameter λ, β_j can be updated as:

$$\beta_j \leftarrow S\left(\sum_{i=1}^{n}(y_i - \sum_{k \neq j} x_{ik}\beta_k(\lambda))x_{ij}, \lambda\right). \tag{2.3.7}$$

In logistic regression, an efficient way to find the penalized estimate is to use the combination of second-order Taylor series approximations (quadratic programming) and the cyclic coordinate descent algorithm. The tremendous value of the cyclic coordinate descent algorithm has been recognized by Friedman et al. [2010] and Wu et al. [2009]. Here, we briefly describe the optimization process and more technical details can be found in the corresponding papers.

1. Start with a large initial λ so that all coefficients are shrunk to zero, $\hat{\boldsymbol{\beta}} = \mathbf{0}$.
2. Decrease λ by a small amount to loose the constraint. For each fixed λ, approximate the unpenalized log-likelihood (2.2.3) using a second-order Taylor series (1.3.3) discussed in Section 1.3.2 with current penalized estimate $\boldsymbol{\beta}^{(s)}$.
3. Apply the coordinate descent algorithm to solve the penalized negative log-likelihood (2.2.4) for an updated penalized estimate $\boldsymbol{\beta}^{(s+1)}$.

4. Repeat steps $1, 2, 3$ until convergence.

2.4 Some Discussion

It is of interest to explore the connections among LASSO, LAR, and forward stagewise. We have briefly mentioned the relation between LAR and forward stagewise where LAR is a more 'democratic' version of forward stagewise. Here, we provide deep analytics to discuss the relationship between LASSO and forward stagewise for solving penalized problems. First, we represent the LASSO (2.1.4) using the expanded predictor space $\tilde{\mathbf{X}} = (\mathbf{X}, -\mathbf{X})$, that is:

$$\hat{\boldsymbol{\beta}^{+}}, \hat{\boldsymbol{\beta}^{-}} = \ argmin\{\sum_{i=1}^{n}(y_i - \mathbf{x_i}^T\boldsymbol{\beta}^{+} + \mathbf{x_i}^T\boldsymbol{\beta}^{-})^2 + \lambda\sum_{j=0}^{p}|\beta_j^{+} + \beta_j^{-}|\} \ \ \boldsymbol{\beta}^{+}, \boldsymbol{\beta}^{-} \geq 0. \quad (2.4.1)$$

Because of the non-negative constraint imposed on the positive version of $\boldsymbol{\beta}^{+}$ as well as the negative version of $\boldsymbol{\beta}^{-}$, it can be shown each version of coefficients compose a monotone regularization path with $\boldsymbol{\beta}^{+}$ creating a non-decreasing regularization path and $\boldsymbol{\beta}^{-}$ following a non-increasing regularization path. More interesting, these monotone LASSO paths are exactly matched with the forward stagewise path but sometimes strikingly different from the LASSO path.

The differences in regularization paths are defined by a different sequence of move directions. For each move, both forward stagewise and LASSO are moving towards the direction associated with the maximum decrease in the residual sum of square (negative log-likelihood, if discrete response). This optimum direction is a function of the L_1 norm of the coefficients β_j in LASSO and it is a function of the L_1 arc-length in forward stagewise which explains the distinct regularization paths produced by each method under most cases. The L_1 arc-length of β is defined as:

$$L_1 \text{ arc-length} = \int_0^t \left| \frac{\partial \beta(t)}{\partial t} \right| dt \tag{2.4.2}$$

where L_1 arc-length is equivalent to L_1 norm when $\beta(t)$ is monotone and piecewise differentiable and that is where the forward stagewise and LASSO have the same regularization path.

Chapter 3

Statistical Models for Longitudinal Data

Longitudinal data analysis has gained remarkable attention in the past 30 years. It has played a prominent role in areas such as translational medicine, clinical practice, sociology, psychology as well as behavior sciences. The longitudinal study design involves repeated measurements on the same object over a given period of time which is capable to address the complex error structure. There are different types of parametric models as well as nonparametric methods to depict the intra-cluster homogeneities and inter-cluster heterogeneities in the repeated measurements based on the distribution of the response, covariance structure, as well as the research question of interest. Here, we concentrate on parametric models for longitudinal data analysis where the models with a complex random error structure stem from the fundamental linear regression model. A parametric statistical model, which depicts the relation between response and covariates in a simplified statistical system, relies on a variety of assumptions. In the simple linear regression model, the error is assumed to be independent and identically distributed with a standard normal distribution. Correspondingly, the response is also assumed to be independent and follow a normal distribution. In addition, the covariate is assumed to have a linear relationship with the response. The generalized linear model assumes independence of the response but does not restrict its distribution to be normal. Although in the generalized linear model, the covariate is nonlinear in the response, it can be transformed using a link function to be linearly linked to the response. In longitudinal analysis, the linear mixed model allows the observations to be correlated while still retaining the normality assumption. The generalized linear mixed model relaxes both the independence and the normality assumption, allowing the observations to be correlated and permits a non-normal distribution. The nonlinear mixed model also relaxes both the independence and normality assumptions. In addition, it allows the covariates to have a true nonlinear relationship with the response.

There are several advantages of using mixed models to take the correlation between measurements into consideration as summarized by Brown and Prescott [2006a] as well as Hedeker and Gibbons [2006]. First, the mixed model leads to a more appropriate estimate of the fixed effects and standard errors when the correlations between observations are not negligible. Second, the estimate of random effects tend to be closer to their population mean compared to if they are treated as fixed effects, which is more robust to the extreme values in the data. Third, as each subject serves as its own control, it can achieve the amount of information as desired with a lower level of statistical power. Fourth, it permits the exploration at the individual level for more precise evaluation of time-dependent effects. Fifth and the most innovative one, the mixed model provides solutions to the challenges posed by missing data by assuming the data is missing at random.

The rest of Chapter 3 is organized as follows: In Section 3.1, we review approaches for measuring correlations between measurements in a normally distributed response, such as repeated ANOVA and MANOVA. We then introduce different types of linear mixed models suitable for longitudinal and hierarchical data. The model fitting strategies are briefly discussed after that. In Section 3.2, we introduce the framework and fitting strategy for the nonlinear mixed model when the response is correlated and follows a non-normal distribution. An example of fitting a nonlinear mixed model using a small dataset is provided as well. In Section 3.4, we introduce the framework of the generalized linear model and its usage for fitting a response having a discrete distribution. We then move to the generalized linear mixed model where a more complex data structure is permitted. Different methods for generalized linear mixed model are also included.

3.1 Linear Mixed Model

3.1.1 Linear Regression Model

We have briefly mentioned the simple linear regression model (2.1.1) in Chapter 2 when introducing the regularization model. Here, we formally introduce this fundamental yet important model using a matrix notation as depicted in (3.1.1):

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon} \tag{3.1.1}$$

where $\mathbf{y}$ is a $n \times 1$ response vector containing all n observations; $\mathbf{X}$ is a design matrix of $n \times p$ dimension; $\boldsymbol{\beta}$ is a $p \times 1$ vector containing p coefficients where the magnitude of each β_j reflects the influence of the corresponding x_j to the response $\mathbf{y}$; and $\boldsymbol{\epsilon}$ is the error, which reflects the non-negligible systematic noise in the model and its elements are assumed to be independent and identically distributed following a normal distribution, $\boldsymbol{\epsilon} \sim N(0, \sigma^2)$. Correspondingly, the response $\mathbf{y}$ is also assumed to be independent and follows a normal distribution $\mathbf{y} \sim N(\mathbf{X}\boldsymbol{\beta}, \sigma^2)$.

It is of interest to estimate the coefficients $\boldsymbol{\beta}$, which describe the relation between the covariates and the response. We mentioned briefly in Chapter 2 the ordinary least squares estimator of $\boldsymbol{\beta}$ is obtained by minimizing the residual sum of square (RSS) (2.1.2). Here, we derive the equation for the ordinary least squares estimator $\hat{\boldsymbol{\beta}}$ using matrix notation. Similar to (2.1.2), we denote the RSS in matrix notation as:

$$RSS(\boldsymbol{\beta}) = (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^T(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) \tag{3.1.2}$$

Using knowledge from calculus, the $\boldsymbol{\beta}$ which minimizes the RSS (3.1.2) can be obtained by solving the first-order derivative of RSS, which can be written as

$$\frac{\partial RSS(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = -2\mathbf{X}^T\mathbf{y} + 2(\mathbf{X}^T\mathbf{X})\boldsymbol{\beta} \tag{3.1.3}$$

Setting (3.1.3) to be zero and solving the equation, we can obtain the ordinary least squares estimator of $\boldsymbol{\beta}$ as:

$$\hat{\boldsymbol{\beta}} = (\mathbf{X^TX})^{-\mathbf{1}}\mathbf{X^Ty} \tag{3.1.4}$$

This OLS estimator $\hat{\boldsymbol{\beta}}$ has some appealing statistical properties which provides great accuracy and precision. First, $\hat{\boldsymbol{\beta}}$ is an unbiased estimator, that is $E(\hat{\boldsymbol{\beta}}) = \boldsymbol{\beta}$. Second, $\hat{\boldsymbol{\beta}}$ has minimum variance in the class of all linear unbiased estimators. The proofs of these properties can be found in [Myers, 1990]. Given these two properties, the OLS estimator $\hat{\boldsymbol{\beta}}$ is often referred to as the *Best Linear Unbiased Estimator* (BLUE). The term 'best' indicates the minimum variance among the class of unbiased estimators. In addition, the BLUE attribute does not depend on the normality assumption of the response.

3.1.2 ANOVA and MANOVA Approaches for Repeated Measurement

We emphasize again in the simple linear regression model, one of the important assumptions is the observations are independent. In reality, it is not uncommon to collect multiple measurements on each subject such that the independence assumption of the observations is violated. Examples of collecting multiple measurements per subject include regular clinical visits of a patient to monitor and control his or her disease progression; measurements of the price of one stock in a given time period; and measurements of tree circumference to infer their growth. Under this scenario, a simple linear regression model is inadequate to capture the correlation among observations and may yield inaccurate estimates regarding the model coefficients. Therefore, a more advanced and comprehensive statistical model is needed.

For now, we assume the observations still follow a normal distribution but the magnitude of correlations among them is not negligible. The first attempt to analyze the hierarchical or longitudinal data is through the 'ANalysis Of VAriance', abbreviated as ANOVA. ANOVA is a statistical procedure used to compare different group means in order to detect certain explanatory variable that contribute significantly to the mean differences, which was first proposed by the famous statistician Ronald Fisher and developed by British astronomer George Biddel Airy. Winer [1971] extended the framework to allow repeated measurements in observations and proposed the mixed-effects ANOVA, or repeated-measure ANOVA. We illustrate the simple but restrictive repeated-measures ANOVA using a simple example. Suppose in a small pilot clinical trial, *three* participating centers were recruiting patients according to a specific protocol. Two interventions, A and B, were randomly assigned to patients in each center. Due to standards of practice being consistent within a center, it is safe to assume there are correlations between clinical outcomes of patients recruited in the same centers. We let y_{ij} denote the clinical outcome for the patient recruited at the i^{th} center and receiving the j^{th} treatment, a repeated-measure ANOVA can be illustrated as a linear model,

$$y_{ij} = \mu + \beta_j + b_i + \epsilon_{ij}, i = 1, 2, 3 \text{ and } j = 1, 2. \tag{3.1.5}$$

where μ is the grand mean; b_i and β_j are quantitative explanatory variables measuring the center effect and the treatment effect, respectively. We further assume the center effect b_i follows a normal distribution $b_i \sim N(0, \sigma_b^2)$, which allows the center-specific variations. The error ϵ also follows a normal distribution where $\epsilon_{ij} \sim N(0, \sigma^2)$. Here, the *treatment* effect is a non-random quantity or fixed effect measuring the magnitude of the independent effect on the dependent variable. In contrast, the *center* effect is a random effect depicting the variabilities among different individuals. It has a subject-specific estimate and allows correlation among observations within each subject which provides a better way to measure and track individual behavior.

To compare the difference between groups and make inference about the explanatory variables, two statistical hypothesis tests are constructed using F-statistics. First, the test for center heterogeneities are constructed as:

$$H_{0,b_i} : \sigma_b^2 = 0; H_{A,b_i} : \sigma_b^2 \neq 0$$
$$F = \frac{MS_{center}}{MS_{residual}} \sim F_{df_{center}, df_{residual}}$$

Second, the test for mean difference in groups receiving different treatments, in other words, the test for treatment effect is constructed as:

$$H_{0,\beta} : \beta_A = \beta_B; H_{A,\beta} : \beta_A \neq \beta_B$$
$$F = \frac{MS_{trmt}}{MS_{residual}} \sim F_{df_{trmt}, df_{residual}}$$

where the mean squares(MS) and the degree of freedom(df) can be found in the ANOVA table presented in Table 3.1.

For a more sophisticated ANOVA model, it is possible to extend the one-way ANOVA model to a two-way ANOVA model by adding the *center* $\times$ *treatment* random-effect interaction. Correspondingly, the linear model (3.1.5) can be revised as

$$y_{ij} = \mu + \beta_j + b_i + (b\beta)_{ij} + \epsilon_{ij}, i = 1, 2, 3; j = 1, 2. \tag{3.1.6}$$

where we still assume $b_i \sim N(0, \sigma_b^2), \epsilon_{ij} \sim N(0, \sigma^2)$. In addition, we assume the two-way interaction follows a normal distribution $(b\beta)_{ij} \sim N(0, \sigma_{b\beta}^2)$ where $\sigma_{b\beta}^2$ accounts for variabilities in two treatment groups from multiple centers. Correspondingly, a modified ANOVA table can be generated and three hypothesis tests will be needed for explore the effects in model. More details can be found in [Hedeker and Gibbons, 2006].

Another historically prevalent method for analyzing nested data is through the 'Multivariate ANalysis Of Variance', commonly known as MANOVA proposed by Bock [1975]. MANOVA is the multi-dimensional form of the univariate ANOVA where the observations from a given group are assumed to follow a multivariate normal distribution, which is more

Table 3.1: One-way random effect ANOVA table

Source	d.f.	Sum of Squares	Mean Squares	E(Mean Squares)
Center	(3-1)=2	$SS_p = 2\sum_{i=1}^{3}(\bar{y}_{i.} - \bar{y}_{..})^2$	$\frac{SS_p}{2}$	$\sigma^2 + 2\sigma_b^2$
Treatment	(2-1)=1	$SS_t = 3\sum_{i=1}^{2}(\bar{y}_{.j} - \bar{y}_{..})^2$	$\frac{SS_t}{1}$	$\sigma^2 + 6\sum(\tau_j - \tau_.)^2$
Residual	(3-1)(2-1)=2	$SS_r = \sum_{i=1}^{3}\sum_{j=1}^{2}(y_{ij} - \bar{y}_{i.} - \bar{y}_{.j} + \bar{y}_{..})^2$	$\frac{SS_r}{(3-1)(2-1)}$	σ^2
Total	$6 \times 2 - 1 = 11$	$SS = \sum_{i=1}^{3}\sum_{j=1}^{2}(y_{ij} - \bar{y}_{..})^2$		

Table 3.2: Data Structure for ANOVA and MANOVA

Center i	Treatment j	Response
1	A	y_{11}
1	B	y_{12}
2	A	y_{21}
2	B	y_{22}
3	A	y_{31}
3	B	y_{32}

Center	Treatment	Response
1	A, B	$\mathbf{y_{11}} = (y_{11}, y_{12})^T$
2	A, B	$\mathbf{y_{21}} = (y_{21}, y_{22})^T$
3	A, B	$\mathbf{y_{31}} = (y_{31}, y_{32})^T$

reflective of the clustered structure of the response. The MANOVA is broadly used to compare multivariate means of several groups with taking the covariance structure from each group into consideration. In MANOVA, all observations from the same subject are evaluated simultaneously. Table 3.2 distinguishes the different structures in data for ANOVA (left) and MANOVA (right), respectively. To illustrate, we still use the pilot clinical trial example discussed previously. Suppose $\mathbf{y_{ik}} = (y_{i1}, y_{i2})$ is a vector of length n_i containing the clinical outcome of 2 patients recruited at the i^{th} center. Therefore, the MANOVA model has a general form of

$$\mathbf{y_{ik}} = \boldsymbol{\mu} + \boldsymbol{\tau_i} + \boldsymbol{\epsilon}_{\mathbf{ik}}, i = 1, 2, 3; k = 2 \tag{3.1.7}$$

where i represents the number of groups or clusters and k represents the number of observations in each group. $\boldsymbol{\mu}$ is the grand mean. $\boldsymbol{\tau}_i = \mathbf{\bar{y}_i} - \mathbf{\bar{y}}$ is the difference between outcome from the i^{th} center versus the overall mean. $\boldsymbol{\epsilon}_{ik} = \mathbf{y_{ik}} - \mathbf{\bar{y}_i}$ measures the residual. Thus, equation (3.1.7) can be rewritten as:

$$\mathbf{y_{ik}} = \mathbf{\bar{y}} + (\mathbf{\bar{y}_i} - \mathbf{\bar{y}}) + (\mathbf{y_{ik}} - \mathbf{\bar{y}_i}) \tag{3.1.8}$$

We denote the total sum of squares and cross products as $\mathbf{T} = (\mathbf{y_{ik}} - \mathbf{\bar{y}})(\mathbf{y_{ik}} - \mathbf{\bar{y}})^T$, the between group sum of squares $\mathbf{B} = (\mathbf{\bar{y}_i} - \mathbf{\bar{y}})(\mathbf{\bar{y}_i} - \mathbf{\bar{y}})^T$ and the within group sum of squares $\mathbf{W} = (\mathbf{y_{ik}} - \mathbf{\bar{y}_i})(\mathbf{y_{ik}} - \mathbf{\bar{y}_i})^T$. To evaluate the center homogeneities, a hypothesis test is constructed as $H_0 : \boldsymbol{\tau}_1 = \boldsymbol{\tau}_2 = \boldsymbol{\tau}_3 = 0$. As discussed in Johnson and Wichern [2007], the null hypothesis is rejected if the Wilk's lambda $\Lambda^* = \frac{|\mathbf{W}|}{|\mathbf{W}+\mathbf{B}|}$ is smaller than a threshold.

Although ANOVA and MANOVA have a long history in modeling longitudinal data, neither of them have demonstrated widespread usage. The major drawbacks prevented both methods from being broadly implemented are the restriction on assumptions and lack of flexibility. The ANOVA model is heavily dependent on a balanced study design, where the number of observations are required to be the same in each experimental group. There are certain modified ANOVA methods suitable for the specific unbalanced or irregular designs, however, it is difficult to generalize for all scenarios. Second, the ANOVA model assumes a compound symmetric covariance structure (See discussion in Section 3.1.3), of which the covariance between any two measurements from one subject is homogeneous. This assumption is somewhat implausible and can be easily violated as the covariance generally increases as time between measurements increases. Third, ANOVA model does not allow for missing data. When missing data occurs, it may be either removed or imputed using group mean or median prior to performing the analysis. However, both approaches will inevitably introduce bias. The MANOVA model permits a more flexible covariance structure with no explicit structure to be assumed. However, it forces each subject to have a homogeneous type of covariance structure. In addition, the MANOVA model does not allow different number of measurements among subjects. It is also inapplicable in situations where some measurements are disordered or missing, which tremendously restricts its usage.

3.1.3 Linear Mixed Model

A linear mixed model is a direct extension of the repeated-measures ANOVA that allows more flexible variance structures. Fisher [1918] first introduced the concept and framework for linear mixed models to study the correlations of trait values between relatives. However, due to the lack of feasible analytical techniques, this model was inhibited from being widely used for a long time. A breakthrough came when Henderson et al. [1959] proposed a mixed model equation by solving which provides a method to estimate the fixed and random effects in the linear mixed model. These estimators have appealing statistical properties in terms of accuracy and precision. This equation opened a door for a new research area concentrated on the techniques to obtain estimates in mixed models where both frequentists and Bayesians have been vigorously involved.

The general form of a linear mixed model is

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\mathbf{b} + \boldsymbol{\epsilon}, \tag{3.1.9}$$

where $\mathbf{y}$ is an $n \times 1$ vector representing the observations; $\mathbf{X}$ and $\mathbf{Z}$ are $n \times p, n \times q$ dimensional design matrices for the fixed and random effects, respectively; $\boldsymbol{\beta}$ is a $p \times 1$ vector of fixed effects, $\mathbf{b}$ is a $q \times 1$ vector of random effects, and $\boldsymbol{\epsilon}$ is a $n \times 1$ error vector. In addition, we assume $\mathbf{b}$ follows a multivariate normal distribution with mean $\mathbf{0}$ and variance $\mathbf{D}$, where $\mathbf{0}$ is a $q \times 1$ vector and $\mathbf{D}$ is a $q \times q$ matrix. The error $\boldsymbol{\epsilon}$ also follows a multivariate normal distribution with mean $\mathbf{0}$ and variance $\mathbf{R}$, where $\mathbf{0}$ is a $n \times 1$ vector and $\mathbf{R}$ is a $n \times n$ matrix. $\boldsymbol{\epsilon}$ is independent of both the fixed and random effects such that $\mathrm{Cov}(\boldsymbol{\beta}, \boldsymbol{\epsilon}) = 0$ and $\mathrm{Cov}(\mathbf{b}, \boldsymbol{\epsilon}) = 0$. $\mathbf{y}$ also follows a multivariate normal distribution with mean $\mathbf{X}\boldsymbol{\beta}$ and variance $\mathbf{V}$, where $\mathbf{V} = \mathrm{Var}(\mathbf{Z}\mathbf{b}) + \mathrm{Var}(\boldsymbol{\epsilon}) = \mathbf{Z}\mathbf{D}\mathbf{Z}^T + \mathbf{R}$.

Suppose y_{ij} represents the j^{th} measurement collected on the i^{th} subject. The linear mixed model can be written as $y_{ij} = \beta_j + b_i + \epsilon_{ij}$, where β_j represents the intercept and coefficients associated with the fixed effects. b_i is the random effect measuring the individual variation. The general form of the linear mixed model (3.1.9) can be derived to three specific types of mixed models by specifying different $\mathbf{D}$ and $\mathbf{R}$ matrices for modeling longitudinal and hierarchical data.

1) Random Effects Model

   The random effects model has its best usage when a clustered structure exists in the response. We illustrate this model using an example. Suppose it is of interest to investigate students' performance on the SAT Exam from $i = 3$ high schools in a region. Given the same educational facilities and teaching resources within one school, it is reasonable to assume the performance of the n_i students within the same school are correlated while uncorrelated between different schools. To build a linear mixed model, we assume the random effect $\mathbf{b}$ follows a multivariate normal distribution with mean $\mathbf{0}$ and variance $\mathbf{D}$ where $\mathbf{D}$ is a diagonal matrix of the form

$$\mathbf{D} = \begin{pmatrix} \sigma_b^2 & 0 & 0 \\ 0 & \sigma_b^2 & 0 \\ 0 & 0 & \sigma_b^2 \end{pmatrix}_{3\times 3} . \tag{3.1.10}$$

In addition we assume the variance of the error $\boldsymbol{\epsilon}$ is also a diagonal matrix of the form

$$\mathbf{R} = \begin{pmatrix} \sigma^2 & 0 & \cdots & 0 \\ 0 & \sigma^2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma^2 \end{pmatrix}_{n\times n, n=\sum_i n_i} . \tag{3.1.11}$$

Therefore, the variance of response $\mathbf{y}$ can be calculated using $\mathbf{V} = \mathbf{Z}\mathbf{D}\mathbf{Z}^{\mathbf{T}} + \mathbf{R}$. The $\mathbf{V}$ matrix is a block diagonal matrix with each block of size $n_i \times n_i$, which represents the correlations of SAT Exam score of n_i students from the same school. Here, as an example, the $\mathbf{V}$ matrix in (3.1.12) represents the covariance structure where there were $n_1 = 3$ students from the first school, $n_2 = 2$ students from the second school, and

$n_3 = 1$ student from the third school.

$$\mathbf{V} = \begin{pmatrix} \sigma_b^2+\sigma^2 & \sigma_b^2 & \sigma_b^2 & 0 & \cdots & 0 \\ \sigma_b^2 & \sigma_b^2+\sigma^2 & \sigma_b^2 & 0 & \cdots & 0 \\ \sigma_b^2 & \sigma_b^2 & \sigma_b^2+\sigma^2 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \sigma_b^2+\sigma^2 & \sigma_b^2 & 0 \\ \vdots & \vdots & \vdots & \sigma_b^2 & \sigma_b^2+\sigma^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma_b^2+\sigma^2 \end{pmatrix}_{n\times n} \tag{3.1.12}$$

Based on this variance structure, the correlation between observations can be calculated as

$$\text{Corr}(y_{ij}, y_{i'j'}) = \begin{cases} 1 & i = i', j = j' \\ \frac{\sigma_b}{\sqrt{\sigma^2+\sigma_b^2}} & i = i', j \neq j' \\ 0 & i \neq i', j \neq j' \end{cases}$$

where the SAT scores of different students from the same school are correlated with correlation coefficient $r = \frac{\sigma_b}{\sqrt{\sigma^2+\sigma_b^2}}$ while the SAT scores of students from different schools are uncorrelated, which is concordant with the previous assumption.

2) Random Coefficient Model

The random coefficient model is particularly useful in scenarios when time-dependent repeated measurements are collected or a growth curve is observed for each subject. The term 'random coefficient' reflects the attribute that at the individual level, a subject-specific intercept as well as slope are fitted to best capture the time trend in the response. There are different variations of random coefficient models appropriate for different types of data , such as the random intercept model where only the intercept is assumed to be subject-specific and random slope model where only the slope is assumed to be subject-specific. Here, we introduce the random coefficient model by assuming both the intercept and slope are heterogeneous across subjects.

We let y_{ij} denote the j^{th} measurement on the i^{th} observation. The random coefficient model can be written as

$$y_{ij} = \beta_j + b_{int,i} + b_{slope,i} \cdot time_{ij} + \epsilon_{ij} \tag{3.1.13}$$

where β_j is the coefficient of the fixed effects; $b_{int,i}$ and $b_{slope,i}$ are the coefficients of the subject-specific intercept and slope, respectively. ϵ_{ij} is the error term. For each observation i, we assume $b_{int,i}$ and $b_{slope,i}$ follow a bivariate normal distribution with mean $\mathbf{0}$ and variance $\mathbf{G_i}$ where $\mathbf{G_i} = \begin{pmatrix} \sigma^2_{int} & \sigma_{int,slope} \\ \sigma_{int,slope} & \sigma^2_{slope} \end{pmatrix}$ is an unstructured variance matrix unless otherwise specified. To illustrate the covariance structure of $\mathbf{V}$ in a random coefficient model, suppose we observed 2 subjects, each with 3 repeated measurements. The design matrix $\mathbf{Z}$ has a form

$$\mathbf{Z} = \begin{pmatrix} 1 & t_{11} & 0 & 0 \\ 1 & t_{12} & 0 & 0 \\ 1 & t_{13} & 0 & 0 \\ 0 & 0 & 1 & t_{21} \\ 0 & 0 & 1 & t_{22} \\ 0 & 0 & 1 & t_{23} \end{pmatrix}. \tag{3.1.14}$$

We assume the variance of the error $\mathbf{R}$ is a diagonal matrix and has a form the same as (3.1.11) and $\mathbf{D}$ is a block diagonal matrix consisting of submatrices $\mathbf{G_i}$. Recall that the covariance matrix $\mathbf{V}$ for the response $\mathbf{y}$ can be calculated using $\mathbf{V} = \mathbf{ZDZ^T} + \mathbf{R}$. By using the $\mathbf{Z}$ matrix that has a similar structure as shown in (3.1.14) to calculate $\mathbf{V}$, it can be shown the covariance matrix $\mathbf{V}$ is also a block diagonal matrix that captures the intra-subject correlations and inter-subject independence, where the elements in the $\mathbf{V}$ matrix can be represented as

$$Cov(y_{ij}, y_{i'j'}) = \begin{cases} \sigma^2 + \sigma^2_{int} + (t_{ij} + t_{ij'})\sigma_{int,slope} + t_{ij}t_{ij'}\sigma^2_{slope} & i = i', j = j' \\ \sigma^2_{int} + (t_{ij} + t_{ij'})\sigma_{int,slope} + t_{ij}t_{ij'}\sigma^2_{slope} & i = i', j \neq j' \\ 0 & i \neq i', j \neq j' \end{cases}.$$

A striking advantage of the random coefficient model that makes it outperform the ANOVA and MANOVA approaches for longitudinal data analysis is that it permits extraordinary flexibilities on the time points of the repeated measurements, that is, the number of measurements collected on each subject can be different and these measurements can be collected at unevenly spaced time points.

3) Covariance Pattern Model

Similar to the random coefficient model, this type of linear mixed model is often used for modeling longitudinal data. Rather than assuming the independence of errors in the diagonal $\mathbf{R}$ matrix and depict the correlations between repeated measurements by estimating elements in the $\mathbf{D}$ matrix, the covariance pattern model characterizes the correlation by assuming a block diagonal structure in matrix $\mathbf{R}$ with individual sub-block measuring the intra-cluster homogeneities as shown in (3.1.15). It permits flexible measurements on each subject by allowing distinct matrix dimensions as well as heterogeneous internal structure of R_i. However, one has to be aware that an overly complicated heterogeneous structure in $\mathbf{R}$ may cause problems in terms of lack of power, over-fitting, and large bias and variation in the parameter estimates. In the covariance pattern model, we no longer assume $\mathbf{b_i}$ as a random effect thus the variance matrix $\mathbf{V}$ of the response $\mathbf{y}$ is equal to the variance matrix $\mathbf{R}$ from the error.

$$\mathbf{V} = \mathbf{R} = \begin{pmatrix} R_1 & 0 & \cdots & 0 \\ 0 & R_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & R_n \end{pmatrix} \tag{3.1.15}$$

The choice of R_i may depend on the patterns and hierarchical structure in data. There are four commonly used structures for R_i listed as follows:

General, or Unstructured

$$\begin{pmatrix} \sigma_{11}^2 & \sigma_{12} & \sigma_{13} \\ \sigma_{21} & \sigma_{22}^2 & \sigma_{23} \\ \sigma_{31} & \sigma_{32} & \sigma_{33}^2 \end{pmatrix}$$

Autoregressive

$$\begin{pmatrix} \sigma^2 & \sigma^2\rho & \sigma^2\rho^2 \\ \sigma^2\rho & \sigma^2 & \sigma^2\rho \\ \sigma^2\rho^2 & \sigma^2\rho & \sigma^2 \end{pmatrix}$$

Compound Symmetry

$$\begin{pmatrix} \sigma^2 & \theta & \theta \\ \theta & \sigma^2 & \theta \\ \theta & \theta & \sigma^2 \end{pmatrix}$$

Toeplitz

$$\begin{pmatrix} \sigma^2 & \sigma_1 & \sigma_2 \\ \sigma_1 & \sigma^2 & \sigma_1 \\ \sigma_2 & \sigma_1 & \sigma^2 \end{pmatrix}$$

In summary, as stated by Brown and Prescott [2006b], the linear mixed model outperforms both ANOVA and MANOVA in the following ways: i) More appropriate and accurate estimates are obtained for the fixed effects and standard errors; ii) the random effects are shrunken with less dispersion; iii) the method is more reasonable and reliable for fitting models having a hierarchical structure; iv) it is better at handling unbalanced and missing data.

3.1.4 Estimating Parameters for a Linear Mixed Model

In general, the unknown parameters need to be estimated in a linear mixed model including: the coefficients for the fixed effects, the coefficients for the random effects and the variance components. The estimates of these parameters can be obtained through the maximum likelihood (ML) or the restricted maximum likelihood (REML) approaches. The restricted likelihood function proposed by Patterson and Thompson [1971] is constructed by marginalizing the fixed effects and is generally preferred over the ML approach to obtain the estimates for the variance component since ML fails to take into account the loss in the degrees of freedom due to estimation of the fixed effects, and thus causes downward bias [Harville, 1977]. In most cases, the estimates of the fixed effects and the random effects should not differ significantly despite which likelihood is implemented. The estimate of the unknown parameters in the mixed model are much more challenging than that in the fixed-effects only model for a few reasons. First, the response in the linear mixed model has a more complex variance structure that measures both intra-subject homogeneity and inter-subject heterogeneity. Second, the fixed effects and random effects are dependent on the variance components where the estimates of the variance components require an effective computational algorithm to iteratively optimize the likelihood function. Previously, analytical mean-based methods were proposed by several researchers to obtain explicit estimates for the variance components in a specific ANOVA model. Notable works include: Herbach [1959] who derived the estimate for the variance component under maximum likelihood for the balanced one-way random-effects model; Thompson [1962] who initiated the estimation using restricted maximum likelihood for any balanced ANOVA model; Hartley and Rao [1967] who provided the solutions for the balanced two-way nested ANOVA model. As the general optimization algorithm be-came available and accessible, Harville [1977] discussed the implementation of an iterative numerical algorithm to obtain the estimates of the variance components. In his paper, he considered two gradient algorithms; the steepest ascent algorithm and the Newton-Raphson algorithm, which are often used for solving constrained maximization problems. Despite being prevalently applied to a variety of optimization circumstances, these two methods were also criticized for their drawbacks. The steepest ascent algorithm was criticized for its intolerably slow convergence [Powell, 1970] while the Newton-Raphson algorithm was subject to the demanding computational effort required for calculating the second-order partial derivatives in the Hessian matrix as well as relying on the cautious choice of the starting value to guarantee convergence. The breakthrough came when Laird and Ware [1982] provided a robust and accurate solution by using the 'Expectation-Maximization' algorithm. The EM algorithm [Dempster et al., 1977] is a two-step iterative algorithm that maximizes a proposed function when some piece of information is missing and where maximum likelihood estimation can not be directly obtained. The EM algorithm was tremendously popular in obtaining the parameter estimates when some observations in the data were missing and were often assumed to be missing at random. In the linear mixed model case, the EM algorithm was used to obtain the estimate of the variance components given that both the fixed effects and the random effects are unknown but generally not exposed to missing observations.

In the rest of this Section, we first derive the MLE estimator for the fixed effects and random effects when the variance component is known. We then describe the model fitting procedure when the variance component is unknown and a detailed description of obtaining the estimates of the variance component through the EM algorithm.

Estimating the Fixed Effects

The estimate of the fixed effects $\boldsymbol{\beta}$ in model (3.1.9) can be obtained directly by solving the first-order derivative of the likelihood function. From the linear mixed model (3.1.9), the response $\mathbf{y}$ follows a normal distribution with mean $\mathbf{X}\boldsymbol{\beta}$ and variance $\mathbf{V}$, where $\mathbf{V} = \mathbf{ZDZ^T} + \mathbf{R}$. To depict the confounding of the variance component to the unknown model parameters, we denote the unobservable elements in $\mathbf{V}$ as $\boldsymbol{\theta} = (\theta_1, \cdots, \theta_m)$, which measures both the intra-subject homogeneity and the inter-subject heterogeneity. Therefore, a joint likelihood for $\boldsymbol{\beta}$ and $\boldsymbol{\theta}$ can be constructed as:

$$L(\boldsymbol{\beta}, \boldsymbol{\theta}; \mathbf{y}) \propto |\mathbf{V}|^{-\frac{1}{2}} \exp\bigg(-\frac{1}{2}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^T \mathbf{V}^{-1}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})\bigg) \tag{3.1.16}$$

The maximum likelihood estimator of $\boldsymbol{\beta}$ is obtained by taking the first-order derivative of $\log L(\boldsymbol{\beta}, \boldsymbol{\theta}; \mathbf{y})$ with respect to $\boldsymbol{\beta}$ and setting the equation to 0, which is $\mathbf{X^T V^{-1}}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) = 0$. Solving the equation, we obtain the expression of $\hat{\boldsymbol{\beta}}$ where

$$\hat{\boldsymbol{\beta}}(\boldsymbol{\theta}) = (\mathbf{X^T V^{-1} X})^{-\mathbf{1}} \mathbf{X^T V^{-1} y}. \tag{3.1.17}$$

Correspondingly, the variance of $\hat{\boldsymbol{\beta}}$ can be calculated as:

$$\mathrm{Var}(\hat{\boldsymbol{\beta}}(\boldsymbol{\theta})) = (\mathbf{X^T V^{-1} X})^{-\mathbf{1}} \mathbf{X^T V^{-1} Var(y) V^{-1} X} (\mathbf{X^T V^{-1} X})^{-\mathbf{1}} = (\mathbf{X^T V^{-1} X})^{-\mathbf{1}}. \tag{3.1.18}$$

We now formally introduce the restricted maximum likelihood and show the estimates of the fixed effects remain consistent under both ML and REML. The REML, also referred to as the residual maximum likelihood, was introduced by Patterson and Thompson [1971]. This likelihood was constructed based on the full residual defined as $\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}$, which captures all source of random variation from both the R-side and D-side. It can be shown the maximum likelihood (3.1.16) can be represented as the product of two likelihoods $L(\boldsymbol{\theta}; \mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}})$ and $L(\boldsymbol{\beta}; \hat{\boldsymbol{\beta}}, \boldsymbol{\theta})$ [Diggle et al., 1994] and the REML can be constructed correspondingly. From equations (3.1.17) and (3.1.18), we assume the posterior distribution of $\hat{\beta}(\boldsymbol{\theta})$ is $\hat{\beta}(\boldsymbol{\theta}) \sim N(\beta(\boldsymbol{\theta}), (\mathbf{X^T V^{-1} X})^{-1})$ thus the likelihood $L(\boldsymbol{\beta}; \hat{\boldsymbol{\beta}}, \boldsymbol{\theta})$ can be written as:

$$L(\boldsymbol{\beta}; \hat{\boldsymbol{\beta}}, \boldsymbol{\theta}) \propto |\mathbf{X^T V^{-1} X}|^{-\frac{1}{2}} \exp\left(\frac{1}{2}(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta})^T \mathbf{X^T V^{-1} X}(\hat{\boldsymbol{\beta}} - \boldsymbol{\beta})\right). \tag{3.1.19}$$

By calculating the ratio of $L(\boldsymbol{\beta}, \boldsymbol{\theta}; \mathbf{y})$ over $L(\boldsymbol{\beta}; \hat{\boldsymbol{\beta}}, \boldsymbol{\theta})$, we can obtain the REML as:

$$L(\boldsymbol{\theta}; \mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}) \propto |\mathbf{X^T V^{-1} X}|^{-\frac{1}{2}} |\mathbf{V}|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}})^T \mathbf{V}^{-1}(\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}})\right) \tag{3.1.20}$$

where we can show the estimate of $\hat{\boldsymbol{\beta}}$ in (3.1.17) also maximizes the REML (3.1.20). Another way to understand the REML is from the Bayesian perspective discussed by Harville [1974]. He proved the REML in (3.1.20) can be obtained by marginalizing the ML (3.1.16) with respect to the fixed effects, assuming the fixed effects have a flat prior.

Estimating of the Random Effects

The estimates of the random effects $\mathbf{b}$ can also be obtained through maximizing the likelihood function. We define a likelihood using the property of conditional probability to distinguish the random variation from different resources. We let $\boldsymbol{\theta}_R$ represent the random variation from the systematic error and $\boldsymbol{\theta}_D$ represent the subject heterogeneity. The likelihood function $L(\boldsymbol{\beta}, \mathbf{b}, \boldsymbol{\theta}; \mathbf{y})$ can be written as the product of two likelihoods $L(\boldsymbol{\beta}, \mathbf{y}, \boldsymbol{\theta}_{\mathbf{R}}|\mathbf{b})$ and $L(\boldsymbol{\theta}_{\mathbf{D}}; \mathbf{b})$. From previous discussion and the knowledge of conditional probability, we can show the probability of $\mathbf{y}$ given the random effects $\mathbf{b}$ follows a normal density with mean $\mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\mathbf{b}$ and variance $\mathbf{R}$ and the random effect $\mathbf{b}$ follows a normal distribution with mean $\mathbf{0}$ and variance $\mathbf{D}$. Therefore, the likelihoods for $L(\boldsymbol{\beta}, \mathbf{y}, \boldsymbol{\theta}_{\mathbf{R}}|\mathbf{b})$ and $L(\boldsymbol{\theta}_{\mathbf{D}}; \mathbf{b})$ can be constructed as follows, respectively:

$$L(\boldsymbol{\beta}, \mathbf{y}, \boldsymbol{\theta}_{\mathbf{R}}|\mathbf{b}) \propto |\mathbf{R}|^{-\frac{1}{2}} \exp\Bigg(-\frac{1}{2}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta} - \mathbf{Z}\mathbf{b})^T \mathbf{R}^{-1} (\mathbf{y} - \mathbf{X}\boldsymbol{\beta} - \mathbf{Z}\mathbf{b})\Bigg) \tag{3.1.21}$$

and

$$L(\boldsymbol{\theta}_{\mathbf{D}}; \mathbf{b}) \propto |\mathbf{D}|^{-\frac{1}{2}} \exp\Bigg(-\frac{1}{2}\mathbf{b}^T \mathbf{D}^{-1} \mathbf{b}\Bigg). \tag{3.1.22}$$

By multiplying the two likelihood functions and taking the log-transformation, the log-likelihood of $L(\boldsymbol{\beta}, \mathbf{b}, \boldsymbol{\theta}; \mathbf{y})$ can be depicted as:

$$\begin{aligned} \log L(\boldsymbol{\beta}, \mathbf{b}, \boldsymbol{\theta}; \mathbf{y}) &= -\frac{1}{2}\Bigg(\log|\mathbf{R}| + (\mathbf{y} - \mathbf{X}\boldsymbol{\beta} - \mathbf{Z}\mathbf{b})^{\mathbf{T}} \mathbf{R}^{\mathbf{-1}} (\mathbf{y} - \mathbf{X}\boldsymbol{\beta} - \mathbf{Z}\mathbf{b}) \\ &+ \log|\mathbf{D}| + \mathbf{b}^{\mathbf{T}} \mathbf{D}^{\mathbf{-1}} \mathbf{b}\Bigg) + C \end{aligned} \tag{3.1.23}$$

where C is a constant. The maximum likelihood estimator of $\mathbf{b}$ is obtained by taking the first-order derivative of (3.1.23) with respect to $\mathbf{b}$. By setting the first-order derivative of equation (3.1.23) to 0 and solving, we obtain the expression of $\hat{\mathbf{b}}$ to be:

$$\hat{\mathbf{b}}(\boldsymbol{\theta}) = (\mathbf{Z^T R^{-1} Z + D^{-1}})^{-1}\mathbf{Z^T R^{-1}}(\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}(\boldsymbol{\theta})) \tag{3.1.24}$$

where the expression of $\hat{\boldsymbol{\beta}}(\boldsymbol{\theta})$ can be obtained in (3.1.17). From previous discussion we know that $\mathbf{V} = \mathbf{ZDZ}^T + \mathbf{R}$ and the inverse of $\mathbf{V}$ can be calculated as $\mathbf{V}^{-1} = (\mathbf{Z}^T)^{-1}\mathbf{D}^{-1}\mathbf{Z}^{-1} + \mathbf{R}^{-1}$. By plugging in this expression, equation (3.1.24) can be further simplified as:

$$\hat{\mathbf{b}}(\boldsymbol{\theta}) = \mathbf{DZ^T V^{-1}}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}(\boldsymbol{\theta})) \tag{3.1.25}$$

The variance of $\hat{\mathbf{b}}(\boldsymbol{\theta})$ can be calculated as:

$$\mathrm{Var}(\hat{\mathbf{b}}(\boldsymbol{\theta})) = \mathbf{DZ^T V^{-1} ZD - DZ^T V^{-1} X(X^T V^{-1} X)^{-1} X^T V^{-1} ZD} \tag{3.1.26}$$

Another way to obtain the estimates of the random effects is through the Bayesian approach. In fact, A.P. Dempster and Tsutakawa [1981] showed by assuming $\mathbf{b}_{prior} \sim N(\mathbf{0}, \mathbf{D})$ and deriving the conditional probability
$\mathbf{b}|\mathbf{y} \sim N(\mathbf{DX^T(XDX^T + R)^{-1}Y, D - DX^T(XDX^T + R)^{-1}XD})$, the posterior mean of $\mathbf{b}$, which minimized the quadratic loss under the Bayesian framework, had the identical expression as (3.1.25). Therefore, this expression is well interpreted by both the frequentists and Bayesians. In addition, from (3.1.25) we can see the estimate of $\mathbf{b}$ is shrunk compared to if it were treated as a fixed effect where $\mathbf{D}^{-1}$ tends to be a zero matrix.

The estimators for the fixed effects $\boldsymbol{\beta}$ and the random effects $\mathbf{b}$ also enjoy appealing statistical properties. When the variance component $\boldsymbol{\theta}$ is known, $\hat{\boldsymbol{\beta}}(\boldsymbol{\theta})$ and $\hat{\mathbf{b}}(\boldsymbol{\theta})$ are solutions to the Henderson mixed model equation system (3.1.27) [Henderson, 1984].

$$\begin{pmatrix} \mathbf{X^T\hat{R}^{-1}X} & \mathbf{X^T\hat{R}^{-1}Z} \\ \mathbf{Z^T\hat{R}^{-1}X} & \mathbf{Z^T\hat{R}^{-1}Z} + \mathbf{\hat{D}^{-1}} \end{pmatrix} \begin{pmatrix} \hat{\boldsymbol{\beta}} \\ \hat{\mathbf{b}} \end{pmatrix} = \begin{pmatrix} \mathbf{X^T\hat{R}^{-1}y} \\ \mathbf{Z^T\hat{R}^{-1}y} \end{pmatrix} \tag{3.1.27}$$

Both Henderson [1984] and Robinson [1991] demonstrated the supreme properties of the estimators in terms of statistical accuracy and precision where $\hat{\boldsymbol{\beta}}(\boldsymbol{\theta})$ is the 'Best Linear Unbiased Estimator'(BLUE) and $\hat{\mathbf{b}}(\boldsymbol{\theta})$ is the 'Best Linear Unbiased Predictor'(BLUP). The term 'estimator' and 'predictor' were historically used to distinguish between the fixed and the random effects. In cases when $\boldsymbol{\theta}$ is unknown, the estimator $\hat{\boldsymbol{\beta}}(\hat{\boldsymbol{\theta}})$ and $\hat{\mathbf{b}}(\hat{\boldsymbol{\theta}})$ can be as close to BLUE and BLUP as possible if $\boldsymbol{\theta}$ is properly estimated.

Estimating of the Variance Component

The estimate of the unknown variance component $\boldsymbol{\theta}$ can be obtained using the general-purpose EM algorithm proposed by Laird and Ware [1982]. The EM algorithm is a two-step iterative algorithm that is able to optimize the proposed function when the function depends on some unobserved data or some piece of information is missing. This unified approach inherits the maximum likelihood approach from the frequentist perspective and the Empirical Bayes estimator from the Bayesian perspective which narrowed the divergence of the two cultures. Another advantage of the EM algorithm is its monotonicity, which increases the likelihood at each iteration and is guaranteed to converge. In the linear mixed model setting, as discussed by Larid et al. [1987], the EM algorithm can be implemented in two apparently different ways. The first approach is referred to as the 'missing data' or the 'incomplete data' [Jennrich and Schluchter, 1985] approach, where the incompleteness implies the number of measurements from certain subjects are less than the standard. The second approach is more natural and intuitive. The observed data is the actual measurements on each subject as in the first approach while the unobserved or the incomplete data refers to the unknown random parameters and error in the mixed model. Here, we primarily concentrate on obtaining the parameter estimates using EM algorithm from the second approach.

We first illustrate the procedure to obtain the ML estimates using the EM algorithm. In the linear mixed model (3.1.9), suppose $\mathbf{R}$ is a diagonal matrix with each diagonal element being σ^2 and $\mathbf{D}$ is a $q \times q$ nonnegative definite matrix. To define a 'Maximization' step, the ML estimates in a complete data setting for σ^2 and $\mathbf{D}$ can be obtained as:

$$\begin{aligned} \hat{\sigma}^2(\boldsymbol{\theta}) &= \sum_i^m \boldsymbol{\epsilon}_i^T(\boldsymbol{\theta})\boldsymbol{\epsilon}_i(\boldsymbol{\theta})/n \\ \hat{\mathbf{D}}(\boldsymbol{\theta}) &= \sum_i^m \mathbf{b_i}(\boldsymbol{\theta})\mathbf{b_i^T}(\boldsymbol{\theta})/m \end{aligned} \tag{3.1.28}$$

where $\hat{\boldsymbol{\epsilon}}_i(\hat{\boldsymbol{\theta}}) = \mathbf{y_i} - \mathbf{x_i}\boldsymbol{\beta}(\hat{\boldsymbol{\theta}}) - \mathbf{z_i}\mathbf{b_i}(\hat{\boldsymbol{\theta}})$ and $n = \sum_i^m n_i$ is the total number of observations and m is the total number of subjects. If the estimate of $\boldsymbol{\theta}$ and $\boldsymbol{\beta}(\boldsymbol{\theta})$ are obtained, we can derive the conditional expectation of $\hat{\sigma}^2(\boldsymbol{\theta}), \hat{\mathbf{D}}(\boldsymbol{\theta})$ given the observed data $\mathbf{y}, \hat{\boldsymbol{\beta}}(\hat{\boldsymbol{\theta}})$ and $\hat{\boldsymbol{\theta}}$, which defines the 'Expectation' step. By using the property of quadratic form of the expectation where $E(\mathbf{y^T A y}) = \boldsymbol{\mu}_y^T \mathbf{A} \boldsymbol{\mu}_y + \text{tr}(\mathbf{AV}),\ \text{Var}(\mathbf{y}) = \mathbf{V}$ (proof can be found in Johnson and Wichern [2007]), the conditional expectation of $E\big(\sum_i^m \boldsymbol{\epsilon}_i^T(\boldsymbol{\theta})\boldsymbol{\epsilon}_i(\boldsymbol{\theta})|\mathbf{y_i}, \hat{\boldsymbol{\beta}}(\hat{\boldsymbol{\theta}}), \hat{\boldsymbol{\theta}}\big)$ and $E\big(\sum_i^m \mathbf{b_i}(\boldsymbol{\theta})\mathbf{b}_i(\boldsymbol{\theta})^T|\mathbf{y_i}, \hat{\boldsymbol{\beta}}(\hat{\boldsymbol{\theta}}), \hat{\boldsymbol{\theta}}\big)$ can be written as the explicit form shown below:

$$
\begin{aligned}
E\big(\sum_i^m \boldsymbol{\epsilon}_i^T(\boldsymbol{\theta})\boldsymbol{\epsilon}_i(\boldsymbol{\theta})|\mathbf{y_i}, \hat{\boldsymbol{\beta}}(\hat{\boldsymbol{\theta}}), \hat{\boldsymbol{\theta}}\big) &= \sum_i^m \big(\hat{\boldsymbol{\epsilon}}_i(\hat{\boldsymbol{\theta}})\hat{\boldsymbol{\epsilon}}_i(\hat{\boldsymbol{\theta}}) + \text{tr Var}(\boldsymbol{\epsilon}_i(\hat{\boldsymbol{\theta}})|\mathbf{y_i}, \hat{\boldsymbol{\beta}}(\hat{\boldsymbol{\theta}}), \hat{\boldsymbol{\theta}})\big) \\
E\big(\sum_i^m \mathbf{b_i}(\boldsymbol{\theta})\mathbf{b}_i(\boldsymbol{\theta})^T|\mathbf{y_i}, \hat{\boldsymbol{\beta}}(\hat{\boldsymbol{\theta}}), \hat{\boldsymbol{\theta}}\big) &= \sum_i^m \big(\hat{\mathbf{b}}_i(\hat{\boldsymbol{\theta}})\hat{\mathbf{b}}_\mathbf{i}(\hat{\boldsymbol{\theta}})^T + \text{tr Var}(\mathbf{b_i}(\hat{\boldsymbol{\theta}})|\mathbf{y_i}, \hat{\boldsymbol{\beta}}(\hat{\boldsymbol{\theta}}), \hat{\boldsymbol{\theta}})\big) \qquad (3.1.29)
\end{aligned}
$$

The EM algorithm works by assigning an initial value of $\boldsymbol{\theta}$ and calculating the conditional expectation of the unknown variance (3.1.30). By substituting the expectations of the ML estimates in the 'M' step and iterating between these two steps, an estimate of $\boldsymbol{\theta}$ is achieved when the log-likelihood reaches its maximum.

We also illustrate the procedure for obtaining the REML estimate using the EM algorithm. To obtain the parameter estimates using REML (3.1.20), the 'M' step remains the same as shown in (3.1.28). In the 'E' step, since the fixed effect is marginalized out of the log-likelihood function, the conditional expectations of $\hat{\sigma}^2(\boldsymbol{\theta}), \hat{\mathbf{D}}(\boldsymbol{\theta})$ only depend on the observed response $\mathbf{y}$ and variance estimate $\hat{\boldsymbol{\theta}}$, where (3.1.30) can be rewritten as:

$$
\begin{aligned}
E\big(\sum_i^m \boldsymbol{\epsilon}_i^T(\boldsymbol{\theta})\boldsymbol{\epsilon}_i(\boldsymbol{\theta})|\mathbf{y_i}, \hat{\boldsymbol{\theta}}\big) &= \sum_i^m \big(\hat{\boldsymbol{\epsilon}}_i(\hat{\boldsymbol{\theta}})\hat{\boldsymbol{\epsilon}}_i(\hat{\boldsymbol{\theta}}) + \text{tr Var}(\boldsymbol{\epsilon}_i(\hat{\boldsymbol{\theta}})|\mathbf{y_i}, \hat{\boldsymbol{\theta}})\big) \\
E\big(\sum_i^m \mathbf{b_i}(\boldsymbol{\theta})\mathbf{b}_i(\boldsymbol{\theta})^T|\mathbf{y_i}, \hat{\boldsymbol{\theta}}\big) &= \sum_i^m \big(\hat{\mathbf{b}}_i(\hat{\boldsymbol{\theta}})\hat{\mathbf{b}}_\mathbf{i}(\hat{\boldsymbol{\theta}})^T + \text{tr Var}(\mathbf{b_i}(\hat{\boldsymbol{\theta}})|\mathbf{y_i}, \hat{\boldsymbol{\theta}})\big) \qquad (3.1.30)
\end{aligned}
$$

The EM iterates between the two steps and leads to smaller estimated variance $\hat{\boldsymbol{\theta}}$ due to gaining an additional degree of freedom from the marginalized fixed effects.

We now illustrate the EM application in estimating variance component in the linear mixed model using a special example. Suppose a linear mixed model has a form (3.1.9) where the random effect $\mathbf{b}$ follows a normal distribution $N(\mathbf{0}, \sigma_1^2 I_{q\times q})$ and the error $\boldsymbol{\epsilon}$ follows a normal distribution $N(\mathbf{0}, \sigma_0^2 I_{n\times n})$. From previous discussion, we know in a complete data setting, the MLE estimates for σ_0^2 and σ_1^2 can be written as:

$$\hat{\sigma_1}^2 = \frac{\mathbf{b^T}(\boldsymbol{\theta})\mathbf{b}(\boldsymbol{\theta})}{q} \tag{3.1.31}$$

$$\hat{\sigma_0}^2 = \frac{\boldsymbol{\epsilon}^T(\boldsymbol{\theta})\boldsymbol{\epsilon}(\boldsymbol{\theta})}{n} \tag{3.1.32}$$

where $\boldsymbol{\epsilon} = \mathbf{y} - \mathbf{X}\boldsymbol{\beta} - \mathbf{Zb}$. Before constructing the quadratic conditional expectation of $E(\mathbf{b^T}(\boldsymbol{\theta})\mathbf{b}(\boldsymbol{\theta})|\mathbf{y}, \hat{\boldsymbol{\beta}}(\hat{\boldsymbol{\theta}}), \hat{\boldsymbol{\theta}})$ and $E(\boldsymbol{\epsilon}^T(\boldsymbol{\theta})\boldsymbol{\epsilon}(\boldsymbol{\theta})|\mathbf{y}, \hat{\boldsymbol{\beta}}(\hat{\boldsymbol{\theta}}), \hat{\boldsymbol{\theta}})$, we first calculate the conditional expectation of $E(\mathbf{b}(\boldsymbol{\theta})|\mathbf{y}, \hat{\boldsymbol{\beta}}(\hat{\boldsymbol{\theta}}), \hat{\boldsymbol{\theta}})$ and $E(\boldsymbol{\epsilon}(\boldsymbol{\theta})|\mathbf{y}, \hat{\boldsymbol{\beta}}(\hat{\boldsymbol{\theta}}), \hat{\boldsymbol{\theta}})$ which will be needed using Bayes rule. Under Bayes Theorem, the posterior distribution $P(\mathbf{b}|\mathbf{y})$ is proportional to the product of the prior and the conditional distribution $P(\mathbf{y}|\mathbf{b}) \times P(\mathbf{b})$, where the conditional distribution $\mathbf{y}|\mathbf{b} \sim N(\mathbf{X}\boldsymbol{\beta} + \mathbf{Zb}, \mathbf{V})$, $\mathbf{V} = \sigma_1^2\mathbf{ZZ^T} + \sigma_0^2\mathbf{I}$ and the prior $\mathbf{b} \sim N(\mathbf{0}, \sigma_1^2 I_{q\times q})$. By conducting some algebra, it can be shown the posterior distribution of $\mathbf{b}$ has the following form:

$$\mathbf{b}|\mathbf{y} \sim N(\sigma_1^2\mathbf{Z^T}\mathbf{V^{-1}}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}), \sigma_1^2\mathbf{I} - \sigma_1^4\mathbf{Z^T}\mathbf{V^{-1}}\mathbf{Z}) \tag{3.1.33}$$

In the same fashion, the posterior distribution of $\boldsymbol{\epsilon}$ given $\mathbf{y}$ follows a normal distribution, that is, $\boldsymbol{\epsilon}|\mathbf{y} \sim N(\sigma_0^2\mathbf{V^{-1}}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}), \sigma_0^2\mathbf{I} - \sigma_0^4\mathbf{V^{-1}})$.

We now can calculate the quadratic conditional expectation according to its property.

$$\begin{aligned}
E(\mathbf{b}^{\mathbf{T}}(\boldsymbol{\theta})\mathbf{b}(\boldsymbol{\theta})|\mathbf{y},\hat{\boldsymbol{\beta}}(\hat{\boldsymbol{\theta}}),\hat{\boldsymbol{\theta}}) &= \mathrm{tr}(\mathrm{Var}(\mathbf{b}|\mathbf{y})) + E(\mathbf{b}|\mathbf{y})^T E(\mathbf{b}|\mathbf{y}) \\
&= \mathrm{tr}(\sigma_1^2(\boldsymbol{\theta})\mathbf{I} - \boldsymbol{\sigma_1^4}(\boldsymbol{\theta})\mathbf{Z^T V}(\boldsymbol{\theta})^{\mathbf{-1}}\mathbf{Z}) + \\
&\sigma_1^4(\boldsymbol{\theta})(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}(\boldsymbol{\theta}))^{\mathbf{T}}\mathbf{V}^{-\mathbf{1}}(\boldsymbol{\theta})\mathbf{Z}\mathbf{Z}^{\mathbf{T}}\mathbf{V}^{-\mathbf{1}}(\boldsymbol{\theta})(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}(\boldsymbol{\theta})) \\
E(\boldsymbol{\epsilon}^T(\boldsymbol{\theta})\boldsymbol{\epsilon}(\boldsymbol{\theta})|\mathbf{y},\hat{\boldsymbol{\beta}}(\hat{\boldsymbol{\theta}}),\hat{\boldsymbol{\theta}}) &= \mathrm{tr}(\sigma_0^2(\boldsymbol{\theta})\mathbf{I} - \boldsymbol{\sigma_0^4}(\boldsymbol{\theta})\mathbf{V}^{-\mathbf{1}}(\boldsymbol{\theta})) + \\
&\sigma_0^4(\boldsymbol{\theta})(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}(\boldsymbol{\theta}))^{\mathbf{T}}\mathbf{V}^{-\mathbf{1}}(\boldsymbol{\theta})\mathbf{V}^{-\mathbf{1}}(\boldsymbol{\theta})(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}(\boldsymbol{\theta}))
\end{aligned}$$

where the estimate of $\boldsymbol{\beta}(\boldsymbol{\theta})$ can be obtained in (3.1.17).

3.2 Nonlinear Mixed Model

In this section we introduce the nonlinear mixed model and the corresponding model fitting strategy. The nonlinear mixed model, as a direct extension from the linear mixed model, provides a powerful and widespread tool to analyze repeated-measured data where the unknown parameters is nonlinear in the response. The nonlinear mixed model is extensively implemented in areas such as pharmacokinetics and pharmacodynamics to evaluate an organism 's response to a given drug dose in order to determine safe dose ranges used in clinical trials. In addition, it has broad applications for measuring growth curves, which we will demonstrate using the 'orange trees' data originally presented by Draper and Smith [1981]. The nonlinear mixed model inherits the appealing properties from the linear mixed model in terms of allowing nonconstant correlation among observations and fitting unbalanced design. However, the true challenge lies in the model fitting procedure where a closed-form expression for the marginalized likelihood does not exist and therefore requires numerical approximations.

3.2.1 The Model Framework

The general form of a nonlinear mixed model was first defined by Lindstrom and Bates [1990] to be:

$$\mathbf{y} = f(\boldsymbol{\phi}, \mathbf{X}) + \boldsymbol{\epsilon} \tag{3.2.1}$$

where $\boldsymbol{\phi} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\mathbf{b_i}$ and $f(\cdot)$ is a nonlinear function. When $f(\cdot)$ is an identity function, the model (3.2.1) is equivalent to the linear mixed model defined in (3.1.9). $\mathbf{X}$ and $\mathbf{Z}$ are the design matrices of dimension $n \times p$ and $n \times q$, respectively. $\boldsymbol{\beta}$ is a $p \times 1$ vector of fixed effects and $\mathbf{b_i}$ is a $q \times 1$ vector of random effects, where $\mathbf{b_i}$ follows a normal distribution $N(\mathbf{0}, \sigma^{\mathbf{2}}\mathbf{D})$. The error $\boldsymbol{\epsilon}$ also follows a normal distribution $N(\mathbf{0}, \sigma^{\mathbf{2}}\mathbf{I})$. In this setting, $\mathbf{D}$ is called the scaled variance because it represents the ratio of variance from the random effect over the variance from the error. The random effect $\mathbf{b_i}$ and the error term $\boldsymbol{\epsilon}$ are independent such that $\mathrm{Cov}(\mathbf{b_i}, \boldsymbol{\epsilon}) = \mathbf{0}$. Therefore, for the j^{th} measurement on the i^{th} subject, the nonlinear mixed model can be written as:

$$y_{ij} = f(\phi_{ij}, \mathbf{x_{ij}}) + \epsilon_{ij}, i = 1, ..., M, j = 1, ...n_i \tag{3.2.2}$$

where $\phi_{ij} = \mathbf{x_{ij}^T}\boldsymbol{\beta} + \mathbf{z_{ij}}\mathbf{b_i}$ and $\mathbf{b_i} \sim N(0, \sigma^2 D)$, $\epsilon_{ij} \sim N(0, \sigma^2)$.

3.2.2 The Marginal Likelihood and its Approximation

A variety of approaches have been proposed to obtain the estimates of model parameters as well as the variance components. Here, we concentrate on the maximum likelihood or restricted maximum likelihood approach to obtain the parameter estimates. In the nonlinear mixed model, we still let $\boldsymbol{\theta}$ represent the variance components that account for the inter-cluster heterogeneity as well as the intra-cluster homogeneity, where the fixed effects $\boldsymbol{\beta}$ and the random effects $\mathbf{b}$ are both functions of $\boldsymbol{\theta}$. We wish to make inference of $\boldsymbol{\theta}$ with respect to the marginal likelihood of $\mathbf{y}$ to make full use of information from the data. The marginal likelihood defined by Lindstrom and Bates [1990] has the following form:

$$p(\mathbf{y}) = \int p(\mathbf{y}|\mathbf{b})p(\mathbf{b})d\mathbf{b}. \tag{3.2.3}$$

As mentioned previously, due to the nonlinearities of the model parameters in the response $\mathbf{y}$, in general there is no closed-form expression of $p(\mathbf{y})$ so that numerical approximation methods are required. Pinheiro and Bates [1995] reviewed four widely accepted approximations to the likelihood: the Linear Mixed Effects approximation proposed by Lindstrom and Bates [1990]; the Laplacian Approximation by L.Tierney and Kadane [1986]; Importance Sampling by Geweke [1989] and Adaptive Gauss-Hermite Quadrature by Davidian and Gallant [1993]. There are close connections among the four approximation methods. In fact, Wolfinger [1993] showed the Linear Mixed Effects Approximation to the restricted log-likelihood is equivalent to the Laplacian approximation to the marginal likelihood (3.2.3) assuming a flat prior for the fixed effect. Important Sampling resembles Adaptive Gauss-Hermite Quadrature in generating the abscissas and weights for approximation. The Laplacian approximation is equivalent to the Gauss-Hermite Quadrature approximation with one quadrature point. Here, we narrow down the discussion on the Adaptive Gauss-Hermite Quadrature approximation.

The Gaussian-Hermite Quadrature algorithm was originally introduced by Davis and Rabinowitz [1975], where the integral of the function with form $f(t)e^{-t^2}$ can be approximated by the sum of a polynomial evaluated at m set of abscissas and weights (z_i^*, w_i) shown in (3.2.4). The abscissas z_i^* are the roots of the m^{th} order Hermite polynomial, which always has m roots in the real number domain. w_i are the corresponding weights calculated by the generalized Hermite weight function. As an attribute of most approximation methods, the precision increases as the number of quadrature points m increases with a price of computational complexity increasing simultaneously.

$$\int_{-\infty}^{\infty} f(t)e^{-t^2} = \sum_{i=1}^{m} w_i f(z_i^*) \tag{3.2.4}$$

A more general Gaussian-Hermite Quadrature approximation was discussed in [Liu and Pierce, 1994] assuming the random variable t follows a normal distribution $N(\mu, \sigma^2)$ that augments more flexibility in the function to be approximated (3.2.5),

$$\int_{-\infty}^{\infty} f(t)e^{-\frac{(t-\mu)^2}{2\sigma^2}} = \sqrt{2}\hat{\sigma} \sum_{i=1}^{m} w_i f(\hat{\mu} + \sqrt{2}\hat{\sigma} z_i^*) \tag{3.2.5}$$

where $\hat{\mu}$ satisfies $\frac{\partial f(t)}{\partial t} = 0$; $\hat{\sigma} = \frac{1}{\sqrt{\hat{j}}}$ and $\hat{j} = \frac{\partial^2 \log f(t)}{\partial t^2}|_{t=\hat{\mu}}$.

In the Gauss-Hermite Quadrature approximation, the abscissas and weights are fixed beforehand to approach the normality of t. Geweke [1989] introduced the Monte Carlo integration where the abscissas and weights are generated randomly from the underlying distribution, which provides a much more efficient way to approximate the integral. Pinheiro and Bates [1995] incorporates this important sampling idea to the Gauss-Hermite Quadrature algorithm to introduce an adaptive Gaussian-Hermite Quadrature (AHQ) procedure where the abscissas and weights are constructed according to both the Hermite polynomial as well as the observed samples.

We denote the AHQ procedure to approximate the marginal likelihood (3.2.3) of the nonlinear mixed model (3.2.2). Similar to constructing the likelihood in the linear mixed model (3.1.23), the likelihood for the nonlinear mixed model can be written as:

$$L(\boldsymbol{\beta}, \mathbf{b_i}, \boldsymbol{\theta}; \mathbf{y}) = (2\pi\sigma^2)^{-q/2}|\mathbf{D}|^{-1/2} \exp\left\{ - \frac{\|\mathbf{y_i} - f(\boldsymbol{\beta}, \mathbf{b_i})\|^2}{2\sigma^2}\right\} \exp\left\{ - \frac{\mathbf{b_i}^T\mathbf{D^{-1}}\mathbf{b_i}}{2\sigma^2}\right\}. \tag{3.2.6}$$

To approximate $\int L(\boldsymbol{\beta}, \mathbf{b}, \boldsymbol{\theta}; \mathbf{y})d\mathbf{b_i}$ using AHQ, we define $g(\boldsymbol{\beta}, \mathbf{D}, \mathbf{b_i}, \mathbf{y_i}) = \|\mathbf{y_i} - f(\boldsymbol{\beta}, \mathbf{b_i})\|^2 + \mathbf{b_i}^T\mathbf{D^{-1}}\mathbf{b_i}$ and $G(\cdot)$ be the second-order derivatives of $g(\cdot)$ with respect to $\mathbf{b_i}$ where $G(\boldsymbol{\beta}, \mathbf{D}, \mathbf{y_i}) = \frac{\partial f(\boldsymbol{\beta}, \mathbf{b_i})}{\partial \mathbf{b_i^T}}|_{b_i=\hat{b_i}} \frac{\partial f(\boldsymbol{\beta}, \mathbf{b_i})}{\partial \mathbf{b_i}}|_{b_i=\hat{b_i}} + \mathbf{D^{-1}}$. We denote $\mathbf{b_i^*} = \hat{\mathbf{b_i}} + \sigma(G(\boldsymbol{\beta}, \mathbf{D}, \mathbf{y_i}))^{-1/2} z_i^*$ to match the right side of (3.2.5), where the quantity $G(\boldsymbol{\beta}, \mathbf{D}, \mathbf{y_i}))^{-1/2}$ represents the important sampling procedure which distinguishes the adaptive Gauss-Hermite procedure from the traditional version. The estimate $\hat{\mathbf{b_i}}$ can be obtained by the mode of $g(\cdot)$, that is $\hat{\mathbf{b_i}} = \arg\min_{\mathbf{b_i}} g(\boldsymbol{\beta}, \mathbf{D}, \mathbf{b_i}, \mathbf{y_i})$. Therefore, the marginal likelihood can be approximated as:

$$\begin{aligned}&\int \left((2\pi\sigma^2)^{-q/2}|\mathbf{D}|^{-1/2} \exp\left\{ - \frac{\|\mathbf{y_i} - f(\boldsymbol{\beta}, \mathbf{b_i})\|^2}{2\sigma^2}\right\} \exp\left\{ - \frac{\mathbf{b_i}^T\mathbf{D^{-1}}\mathbf{b_i}}{2\sigma^2}\right\}\right) d\mathbf{b_i} \\ &\simeq \sum_{j_1=1}^{N_{GHQ}} \cdots \sum_{j_q=1}^{N_{GHQ}} \Big(\exp\left(-g(\beta, \mathbf{D}, \mathbf{b_i}, \mathbf{y_i}, \hat{\mathbf{b_i}} + \sigma(G(\boldsymbol{\beta}, \mathbf{D}, \mathbf{y_i}))^{-1/2} z_i^*\right)/2\sigma^2 + \|z_{j_1^* \cdots j_q^*}\|^2/2)\Big) \prod_{k=1}^{q} w_{jk}.\end{aligned} \tag{3.2.7}$$

Correspondingly, the marginal log-likelihood can be written as

$$\begin{aligned}&-[N\log 2\pi\sigma^2 + M\log|\mathbf{D}| + \sum_{i=1}^{M} \log|G(\boldsymbol{\beta}, \mathbf{D}, \mathbf{y_i})|]/2+ \\ &\sum_{i=1}^{M} \log \sum_{j_1=1\cdots j_q=1}^{N_{GHQ}} \Big(\exp\left(-g(\boldsymbol{\beta}, \mathbf{D}, \mathbf{y_i}, \hat{\mathbf{b_i}} + \sigma(G(\boldsymbol{\beta}, \mathbf{D}, \mathbf{y_i}))^{-1/2} z_i^*\right)/2\sigma^2 + \|z_{j_1^* \cdots j_q^*}\|^2/2)\Big) \prod_{k=1}^{q} w_{jk}.\end{aligned} \tag{3.2.8}$$

As mentioned previously, when one set of abscissas and weight (z_1^*, w_1) is used, the Adaptive Gaussian-Hermite quadrature is equivalent to the Laplacian approximation where (3.2.8) can be simplified as

$$-[N(1+\log 2\pi\hat{\sigma}^2) + M\log|\mathbf{D}| + \sum_{i=1}^{M}\log|G(\boldsymbol{\beta}, \mathbf{D}, \mathbf{y_i})|]/2. \tag{3.2.9}$$

3.2.3 Estimating of the Parameters

The procedures for obtaining the estimates of the variance components and the unknown model parameters in the nonlinear mixed model are similar to that for the linear mixed model. We again let $\boldsymbol{\theta} = (\theta_1, \cdots, \theta_m)$ be the unobservable intra-subject and inter-subject variances where the fixed effects $\boldsymbol{\beta}$ and the random effects $\mathbf{b_i}$ are both functions of $\boldsymbol{\theta}$. The estimates of the variance component $\boldsymbol{\theta}$ can be obtained using the general-purpose EM algorithm with modification for the nonlinear mixed model. After obtaining the estimates of the variance components, the estimates of the fixed effects can be obtained by optimizing the approximation of the marginal likelihood. The optimization techniques have been introduced in Section (1.3.2). The random effects can be estimated using an Empirical Bayes approach, where the posterior distribution of $\mathbf{b_i}$ can be determined by:

$$p(\mathbf{b}|\mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{b})p(\mathbf{b})}{\int p(\mathbf{y}|\mathbf{b})p(\mathbf{b})d\mathbf{b}}. \tag{3.2.10}$$

We demonstrate the parameter estimates procedure using the following example.

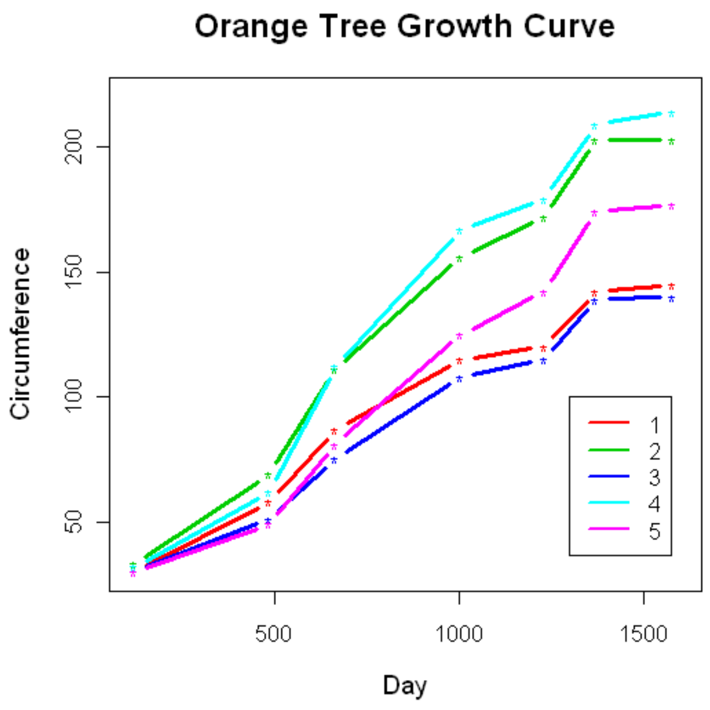


Figure 3.1: Orange Tree Growth Curves

3.2.4 Orange Tree Example

The orange tree data [Draper and Smith, 1981] consists of seven measurements of trunk circumference on five orange trees. Since the trunk circumferences have an apparent nonlinear relationship with time as shown in Figure 3.1, it is of interest to fit a nonlinear model that can predict the growth pattern of the orange trees. Lindstrom and Bates [1990] as well as Pinheiro and Bates [1995] proposed a nonlinear mixed effects model that had good performance in modeling the orange tree data. For the i^{th} orange tree, the j^{th} measurement can be modeled using the following form:

$$y_{ij} = \frac{b_1 + u_{i1}}{1 + \exp(-(d_{ij} - b_2)/b_3)} + \epsilon_{ij} \; i = 1, \cdots, 5 \text{ and } j = 1, \cdots, 7 \tag{3.2.11}$$

where d_{ij} represents the measurement time; $\mathbf{b} = (b_1, b_2, b_3)$ is a vector of three fixed effects; u_{i1} is the random effect which follows a normal distribution, $u_{i1} \sim N(0, \sigma_u^2)$; and ϵ_{ij} is the error term which also follows a normal distribution, $\epsilon_{ij} \sim N(0, \sigma_e^2)$. The scaled variance $\mathbf{D}$ described in (3.2.2) is degenerated to a ratio of σ_u^2/σ_e^2.

To obtain the estimates of variance components σ_e^2 and σ_u^2 in the nonlinear mixed model using the EM algorithm, we rewrite model (3.2.11) accordingly. We denote $y^*_{ij} = y_{ij} - \frac{b_1}{1+\exp(-(d_{ij}-b_2)/b_3)}$ and $z^*_{ij} = \frac{u_{i1}}{1+\exp(-(d_{ij}-b_2)/b_3)}$, thus model (3.2.11) can be rewritten as $y^*_{ij} = z^*_{ij} + \epsilon_{ij}$. Equivalently, the model can be written using matrix notation $\mathbf{y}^* = \mathbf{Z}^* + \boldsymbol{\epsilon}$. Let $\mathbf{V}$ be the variance matrix of $\mathbf{y}^*$ where $\mathbf{V} = \sigma_u^2 \mathbf{Z}^* \mathbf{Z}^{*\mathbf{T}} + \sigma_e^2 \mathbf{I}_{n\times n}$. In the EM algorithm, for unknown $\mathbf{u}$ and $\boldsymbol{\epsilon}$, the 'E-step' can be constructed using the quadratic expectation conditional on the observed response $\mathbf{y}$ and current estimate $\hat{\mathbf{b}}$ according to (3.1.30). In particular, for the orange tree data, the two conditional expectations can be written as:

$$E(\boldsymbol{\epsilon}^T \boldsymbol{\epsilon} | \mathbf{y}^*, \hat{\mathbf{b}}, (\sigma_e^2)^{(k)}) = (\sigma_e^4)^{(k)} \cdot \mathbf{y}^{*\mathbf{T}} (\mathbf{V}^{-1})^{\mathbf{T}} \mathbf{V}^{-1} \mathbf{y}^* + (\sigma_e^2)^{(k)} \cdot N - (\sigma_e^4)^{(k)} \cdot \mathrm{tr}(\mathbf{V}^{-1})$$

$$E(\mathbf{u}\mathbf{u}^{\mathbf{T}} | \mathbf{y}^*, \hat{\mathbf{b}}, (\sigma_e^2)^{(k)}) = (\sigma_u^4)^{(k)} \cdot \mathbf{y}^{*\mathbf{T}} (\mathbf{V}^{-1})^{\mathbf{T}} \mathbf{Z}^* \mathbf{Z}^{*\mathbf{T}} \mathbf{V}^{-1} \mathbf{y}^* + (\sigma_u^2)^{(k)} \cdot N - (\sigma_u^4)^{(k)} \cdot \mathrm{tr}(\mathbf{Z}^{*\mathbf{T}} \mathbf{V}^{-1} \mathbf{Z}^*)$$

and the variance component $(\hat{\sigma_u}^2)^{(k+1)}$ and $(\hat{\sigma_e}^2)^{(k+1)}$ can be estimated using $(\hat{\sigma_u}^2)^{(k+1)} = \frac{E(\boldsymbol{\epsilon}^T \boldsymbol{\epsilon} | \mathbf{y}^*, \hat{\mathbf{b}}, (\sigma_e^2)^{(k)})}{N}$ and $(\hat{\sigma_e}^2)^{(k+1)} = \frac{E(\mathbf{u}\mathbf{u}^{\mathbf{T}} | \mathbf{y}^*, \hat{\mathbf{b}}, (\sigma_e^2)^{(k)})}{N}$. The estimates are incorporated to iteratively update the log-likelihood $L = -\frac{1}{2}(\log|\mathbf{V}| + \mathbf{y}^{*\mathbf{T}} \mathbf{V}^{-1} \mathbf{y}^*)$ until the optimum is reached.

The estimated variances $\hat{\sigma_u^2}$ and $\hat{\sigma_e^2}$ are substituted for the unknown variances in the marginal likelihood for obtaining the estimates of the fixed effects $\mathbf{b}$. We construct the marginal likelihood below and approximate it using the adaptive Gauss-Hermite Quadrature method. For each orange tree i, the likelihood function can be written as:

$$L_i = p(\mathbf{y}|\mathbf{u_i})p(\mathbf{u_i}) = \left(\frac{1}{\sqrt{2\pi\sigma_e^2}}\right)^{n_i+1} |\mathbf{D}|^{-\frac{1}{2}} \exp\left(- \frac{\|y_i - f_i(\mathbf{b};\mathbf{u_i})\|^2 + \mathbf{u_i}\mathbf{D^{-1}}\mathbf{u_i}}{2\sigma_e^2}\right) \quad (3.2.12)$$

where $f_i(\mathbf{b};\mathbf{u_i}) = \frac{\hat{b}_1}{1+\exp(-(d_{ij}-\hat{b}_2)/\hat{b}_3)}$. The marginal likelihood for all observations can be written as:

$$L^* = \int \prod_{i=1}^{M} L_i d\mathbf{u_i} = \left(\frac{1}{\sqrt{2\pi\sigma_e^2}}\right)^{N+M} |\mathbf{D}|^{-\frac{M}{2}} \prod_{i=1}^{M} \exp\left(- \frac{\|y_i - f_i(\mathbf{b};\mathbf{u_i})\|^2 + \mathbf{u_i}\mathbf{D^{-1}}\mathbf{u_i}}{2\sigma_e^2}\right) d\mathbf{u_i} \quad (3.2.13)$$

where M denotes the number of subjects and N denotes the number of observations. To approximate the marginal likelihood (3.2.13) using adaptive Gauss-Hermite Quadrature algorithm (3.2.7), we assume $g(\mathbf{b},\mathbf{D},\mathbf{u_i},y_i) = \|y_i - f(\mathbf{b},\mathbf{u_i})\|^2 + \mathbf{u_i^T}\mathbf{D^{-1}}\mathbf{u_i}$ and $G(\mathbf{b},\mathbf{D},\mathbf{u_i},y_i) = \frac{\partial f(\boldsymbol{\beta},\mathbf{b_i})}{\partial \mathbf{b_i^T}}|_{b_i=\hat{b_i}} \frac{\partial f(\boldsymbol{\beta},\mathbf{b_i})}{\partial \mathbf{b_i}}|_{b_i=\hat{b_i}} + \mathbf{D^{-1}}$. According to (3.2.7), $\log L^*$ can be approximated as:

$$\begin{aligned} &-[N\log(2\pi\hat{\sigma_e}^2) + M\log|\mathbf{D}| + \sum_{i=1}^{M} \log|G(\mathbf{b},\mathbf{D},\mathbf{u_i},y_i)|/2 + \\ &\sum_{i=1}^{M} \log(\sum_{j=1}^{N_{GHQ}} \big(\exp(-g(\mathbf{b},\mathbf{D},y_i,\hat{\mathbf{u}}_\mathbf{i} + \hat{\sigma_e}(G(\mathbf{b},\mathbf{D},\mathbf{u_i},y_i))^{-1/2} z_i^*)/2\hat{\sigma_e}^2 + \|z_j^*\|^2/2)\big)\, w_j. \end{aligned} \quad (3.2.14)$$

When $N_{GHQ} = 1$, equation (3.2.14) is equivalent to the Laplacian approximation with the following form:

$$-\big(N(1+\log(2\pi\hat{\sigma}_e^2)) + M\log|\mathbf{D}| + \sum_{i=1}^{M}\log|G(\mathbf{b},\mathbf{D},\mathbf{u_i},y_i|\big)/2 \qquad (3.2.15)$$

By optimizing the approximated marginal likelihood using the general-purpose optimization techniques, (*e.g.* Nelder-Mead, BFGS) we can obtain the estimates for the fixed effects.

We fit the orange tree data using four different approaches: 1) Our R code, 2) R package `lme4`, 3) SAS `PROC NLMIXED`, and 4) Bayesian approach via WinBUGS version 14. We compared the parameter and the log-likelihood estimates resulting from the different approaches. For the first three computational packages/software, we obtained the marginal likelihood function using both the Laplacian approximation and the adaptive Gauss-Hermite Quadrature with five quadrature points. The techniques for obtaining the estimates of the unknown parameters differ slightly among the three approaches. In our code, the estimated variance components were obtained using the EM algorithm to optimize the ML criterion and the log-likelihood was optimized using Nelder-Mead, which is the default method implemented in R function `optim`. The R package `lme4`, written by D. Bates and Bolker [2011], implements Penalized Iteratively Reweighted Least Squares (PIRLS) [Bates, 2011] to obtain the parameter estimates with options to estimate the latent variable $\boldsymbol{\theta}$ by optimizing either the ML or REML criterion. For comparison purposes, we choose the option to obtain $\hat{\boldsymbol{\theta}}$ by ML criterion. In SAS version 9.2 `PROC NLMIXED` procedure, we used the default Quasi-Newton algorithm to optimize the ML based marginal likelihood. In the Bayesian approach, we set the $burning = 1000, thinning = 5, iteration = 230000$. The priors for the variance component were chosen according to [D.K. Dey and Chang, 1997] and the priors for the fixed effects were the flat priors. Tables 3.3, 3.4, 3.5 and 3.6 present the parameter estimates from the four computational softwares/packages. By comparing the results in the four tables, we can conclude the estimates of the fixed effects are robust across optimization algorithms as well as approximation procedures. However, the estimates of variance can be quite different depending on the optimization techniques as well as the number of quadrature points used in the approximation procedure. In particular, there is apparently a huge inflation in the estimate of variance using Laplacian approximation by the R package `lme4`. In addition, the estimate of variance under the Bayesian framework differed significantly compared to that from the frequentist approaches.

Table 3.3: R code output for the orange tree example

Laplacian Approximation					
Parameter	Estimate	SE	DF	t-value	P-value
b1	192.1	15.65	4	12.33	0.0002*
b2	727.9	35.09	4	21.14	<0.001*
b3	348.0	26.98	4	13.23	0.0002*
σ_u^2	1002.29		4		
σ_e^2	61.51		4		
$-logL$=131.57					
Adaptive Gaussian Quadrature $N_{GHQ} = 5$					
Parameter	Estimate	SE	DF	t-value	P-value
b1	192.1	11.32	4	16.97	<0.001*
b2	728.4	33.32	4	21.85	<0.001*
b3	348.5	30.32	4	11.50	0.0003*
σ_u^2	974.42		4		
σ_e^2	61.80		4		
$-logL$=133.32					

Table 3.4: R package `lme4` output for the orange tree example

Laplacian Approximation					
Parameter	Estimate	SE	DF	t-value	P-value
b1	192.0	104.09	4	1.85	0.14
b2	727.9	31.97	4	22.78	<0.001*
b3	348.0	24.42	4	14.25	0.001*
σ_u^2	53985.2	232.35	4	232.34	<0.001*
σ_e^2	52.9	7.27	4	7.28	0.002*
$-logL$=945.3					
Adaptive Gaussian Quadrature $N_{GHQ} = 5$					
Parameter	Estimate	SE	DF	t-value	P-value
b1	192.1	15.58	4	12.32	0.0002*
b2	727.9	34.44	4	21.14	<0.001*
b3	348.1	26.31	4	13.23	0.0002*
σ_u^2	1001.5	31.65	4	31.64	<0.001*
σ_e^2	61.51	7.84	4	7.84	0.001*
$-logL$=129.9					

Table 3.5: SAS PROC NLMIXED output for the orange tree example

Laplacian Approximation					
Parameter	Estimate	SE	DF	t-value	P-value
b1	192.1	15.65	4	12.27	0.0003*
b2	727.9	35.25	4	20.65	<0.001*
b3	348.1	27.08	4	12.85	0.0002*
σ_u^2	999.9	647.44	4	1.54	0.20
σ_e^2	61.51	15.89	4	3.87	0.02*
$-logL$=132.5					
Adaptive Gaussian Quadrature $N_{GHQ} = 5$					
Parameter	Estimate	SE	DF	t-value	P-value
b1	192.1	15.65	4	12.27	0.0003*
b2	727.9	35.25	4	20.65	<0.001*
b3	348.1	27.08	4	12.85	0.0002*
σ_u^2	999.9	647.44	4	1.54	0.20
σ_e^2	61.51	15.89	4	3.87	0.02*
$-logL$=132.5					

Table 3.6: WinBUGS output for the orange tree example

Parameter	Estimate	SD	MC error	Bayesian Interval
b1	193.1	13.34	0.2634	[168.3, 221.1]
b2	731.5	43.03	0.6831	[658.6, 827.2]
b3	351.7	32.48	0.5181	[292.3, 419.6]
σ_u^2	561.8	185.1	2.881	[278.1, 994.6]
σ_e^2	86	26.46	0.4006	[47.91, 153.2]

3.3 Generalized Linear Model

3.3.1 Generalized Linear Model Framework

The generalized Linear Model (GLM) first introduced by Nelder and Wedderburn [1972] and further extended by McCullagh and Nelder [1989] provides a flexible generalization from the linear model that allows the distribution of the response to be discrete and non-normal. GLM converts the nonlinear relationship between the response and predictors to a linear relationship by linking the function of the response variable to a linear predictor where the statistical inference in the linear model can be directly adapted. In practice, it often restricts GLMs to the exponential family of distribution of the response variable that includes the most commonly used statistical models: linear regression, logistic regression, Poisson regression, *etc.* This unified class of distributions constructs a convenient framework that allows the same model fitting strategy to be applied to models with different error distributions.

A typical GLM consists of three parts: the random component, the systematic component and a link function. The random component specifies the distribution of response y; the systematic component defines the linear predictors; and the link function connects the two components. In the random component, the distribution of y is often belongs to an exponential family, which covers a large number of commonly used distribution, such as: normal, binomial, gamma, exponential, beta, *etc.* A general form of the exponential family can be written as:

$$f(y_i;\theta_i) = a(\theta_i)b(y_i)\exp(y_i Q(\theta_i)) \tag{3.3.1}$$

where y_i is the response from the i^{th} observation and θ is the natural parameter that needs to be estimated; $a(\cdot)$, $b(\cdot)$ and $Q(\cdot)$ are functions with $Q(\cdot)$ specifying the form of the link function, known as $g(\cdot)$. When the response is discrete, one parameter θ may be inadequate to capture the dispersion and variation in data. Alternatively, a modified form of the exponential family with an additional parameter ϕ can be used to adjust for the over-dispersion and under-dispersion. Accordingly, the two-parameter exponential family density $f(y_i;\theta,\phi)$ can be written as:

$$f(y_i;\theta_i,\phi) = \exp\left([y_i\theta_i - b(\theta_i)]/a(\phi) + c(y_i,\phi)\right) \tag{3.3.2}$$

where θ is the natural parameter and ϕ is the dispersion parameter. When ϕ is known or the estimate of ϕ is observed, (3.3.2) simplifies to the form (3.3.1) where function $Q(\cdot)$ still determines the form of the link function $g(\cdot)$.

The systematic component specifies the linear predictor as that in linear regression model. We denote $\eta_i = \sum_j \beta_j x_{ij}, j = 1, \cdots, p$, where β_j represents the regression coefficient and x_{ij} are the corresponding elements in the design matrix $\mathbf{X}$. The mean response of the i^{th} observation (denote as μ_i) is therefore connected to the linear component η_i via link function $g(\cdot)$ with a form of:

$$g(\mu_i) = \eta_i = \sum_j \beta_j x_{ij} \tag{3.3.3}$$

which creates the framework of GLM.

3.3.2 Moments and Likelihood for GLM

We now review some important statistical inference and properties of GLM. According to (3.3.2), the log-likelihood for observation i can be written as:

$$L_i(\theta, \phi; y_i) = [y_i\theta_i - b(\theta_i)]/a(\phi) + c(y_i, \phi). \tag{3.3.4}$$

To simplify the notation, we denote the log-likelihood of L_i instead of $L_i(\theta, \phi; y_i)$. We show in (3.3.5) by taking first-order and second-order derivatives of L_i with respect to the natural parameter θ_i, the mean and variance of response y_i can be written as functions of $a(\phi)$ and $b(\theta_i)$. According to (3.3.4), it follows

$$\frac{\partial L_i}{\partial \theta_i} = \frac{y_i - b'(\theta_i)}{a(\phi)}, \frac{\partial^2 L_i}{\partial \theta_i^2} = \frac{-b''(\theta_i)}{a(\phi)} \tag{3.3.5}$$

where $b'(\theta_i)$ and $b''(\theta_i)$ denote the first and second order derivatives of function $b(\cdot)$ evaluated at θ_i. When the likelihood reaches its maximum yields $E\left(\frac{\partial L_i}{\partial \theta_i}\right) = 0$ follows $\mu_i = E(y_i) = b'(\theta_i)$. In addition, the relation between mean and variance can be illustrated as $E\left(\frac{\partial^2 L_i}{\partial \theta_i^2}\right) = E\left(\frac{\partial L_i}{\partial \theta_i}\right)^2$ follows $Var(y_i) = b''(\theta_i)a(\phi)$.

It is also of interest to make the inference on the model parameter β_j to investigate how the explanatory variable impacts the outcome. Since β_j is connected to the log-likelihood L_i through parameters μ_i and θ_i, by applying the chain rule and taking the first-order derivative of log-likelihood with respect to β_j, the likelihood equation can then be written as:

$$\frac{\partial L_i}{\partial \beta_j} = \frac{\partial L_i}{\partial \theta_i}\frac{\partial \theta_i}{\partial \mu_i}\frac{\partial \mu_i}{\partial \beta_j} = 0. \tag{3.3.6}$$

From previous knowledge, we know $\frac{\partial L_i}{\partial \theta_i} = \frac{y_i - b'(\theta_i)}{a(\phi)}$ and $\frac{\partial \theta_i}{\partial \mu_i} = (\frac{\partial \mu_i}{\partial \theta_i})^{-1} = b''(\theta_i)^{-1} = \frac{a(\phi)}{Var(Y_i)}$. By substituting the two explicit forms into (3.3.6), the likelihood equation can be updated as:

$$\frac{\partial L_i}{\partial \beta_j} = \frac{(y_i - \mu_i)}{Var(y_i)} \frac{\partial \mu_i}{\partial \beta_j} = 0. \tag{3.3.7}$$

Correspondingly, the likelihood equation for all observations according to (3.3.7) can be written as:

$$\frac{\partial L}{\partial \beta_j} = \sum_{i=1}^{N} \frac{\partial L_i}{\partial \beta_j} = \sum_{i=1}^{N} \frac{(y_i - \mu_i)}{Var(y_i)} \frac{\partial \mu_i}{\partial \beta_j} = 0. \tag{3.3.8}$$

Equation (3.3.8) is also known as the quasi-score function and its matrix form is defined as:

$$\mathbf{U}(\boldsymbol{\beta}; \mathbf{y}) = \frac{\partial L}{\partial \boldsymbol{\beta}} = \mathbf{D}'\mathbf{V}^{-\mathbf{1}}(\mathbf{y} - \boldsymbol{\mu}) \tag{3.3.9}$$

where $\mathbf{D}$ is the $n \times p$ matrix with its $(i,j)^{th}$ entry to be $\frac{\partial \mu_i}{\partial \beta_j}$; V is the $n \times n$ diagonal matrix with the i^{th} diagonal entry to be $Var(y_i)$; and $\mathbf{y} - \boldsymbol{\mu}$ is a vector of length n with the i^{th} entry to be $y_i - \mu_i$.

Alternatively, since $\frac{\partial \mu_i}{\partial \beta_j} = \frac{\partial \mu_i}{\partial \eta_i} \frac{\partial \eta_i}{\partial \beta_j} = x_{ij} \frac{\partial \mu_i}{\partial \eta_i}$, (3.3.8) can also be written as

$$\frac{\partial L}{\partial \beta_j} = \sum_{i=1}^{N} \frac{(y_i - \mu_i) x_{ij}}{Var(y_i)} \frac{\partial \mu_i}{\partial \eta_i} = 0 \tag{3.3.10}$$

with the matrix notation as:

$$\mathbf{U}(\boldsymbol{\beta};\mathbf{y}) = \frac{\partial L}{\partial \boldsymbol{\beta}} = \mathbf{X^T W}(\mathbf{y}-\boldsymbol{\mu})\frac{\partial \boldsymbol{\mu}}{\partial \boldsymbol{\eta}} \tag{3.3.11}$$

where $\mathbf{X}$ is a $n \times p$ design matrix; $\mathbf{W}$ is a $n \times n$ diagonal matrix with the i^{th} diagonal entry to be $w_i = (\frac{\partial \mu_i}{\partial \eta_i})^2/Var(y_i)$; and $(\mathbf{y}-\boldsymbol{\mu})\frac{\partial \boldsymbol{\mu}}{\partial \boldsymbol{\eta}}$ is a vector of length n with the i^{th} entry to be $(y_i - \mu_i)\frac{\partial \mu_i}{\partial \eta_i}$.

3.3.3 Maximum Likelihood Estimates for GLM

The estimate of model parameter $\boldsymbol{\beta}$ can be obtained using the maximum likelihood approach. The ML estimates have two appealing theoretical properties: they are approximately unbiased and highly efficient. It is required to solve the gradient of the log-likelihood (3.3.7) to obtain the ML estimate of $\boldsymbol{\beta}$. There is generally no closed form in the expression of $\boldsymbol{\beta}$, the iterative optimization procedures are therefore needed. Besides Newton-Raphson and Fisher's Scoring methods discussed in Section 1.3.2, we here introduce the third prevalent optimization method: Iterative Reweighted Least Squares, often known as IRLS [Green, 1984].

In the discussion of Fisher's Scoring algorithm, we introduced the information matrix $\boldsymbol{\iota_i}$ with a form of $\boldsymbol{\iota_i} = -E\left(\frac{\partial^2 L_i}{\partial \beta_j \partial \beta_{j'}}\right)$ for the i^{th} observation. According to (3.3.10) and

$E(\frac{\partial L_i}{\partial \beta_j}) = 0$, the explicit form of information matrix $\boldsymbol{\iota_i}$ for GLM can be written as:

$$\boldsymbol{\iota_i} = -E\left(\frac{\partial^2 L_i}{\partial \beta_j \partial \beta_{j'}}\right) = E\left(\frac{\partial L_i}{\partial \beta_j}\right)\left(\frac{\partial L_i}{\partial \beta_{j'}}\right) = E\left(\frac{(y_i - \mu_i)^2 x_{ij} x_{ij'}}{Var(y_i)^2}\left(\frac{\partial \mu_i}{\partial \eta_i}\right)^2\right). \tag{3.3.12}$$

As $E(y_i - \mu_i)^2 = Var(y_i)$, after a few steps of algebra, the information matrix $\boldsymbol{\iota}$ for N observations can be written as:

$$\boldsymbol{\iota} = \sum_{i=1}^{N} \frac{x_{ij} x_{ij'}}{Var(y_i)}\left(\frac{\partial \mu_i}{\partial \eta_i}\right)^2 = \mathbf{X'WX} \tag{3.3.13}$$

where $\mathbf{W}$ is a $n \times n$ diagonal matrix with the i^{th} diagonal entry to be $w_i = (\frac{\partial \mu_i}{\partial \eta_i})^2 / Var(y_i)$. Recall in Fisher's Scoring algorithm, $\boldsymbol{\beta}^{(s)}$ can be updated using iterative procedure that satisfies $\boldsymbol{\beta}^{(s)} = \boldsymbol{\beta}^{(s-1)} + \boldsymbol{(\iota^{(s-1)})^{-1} U^{(s-1)}}$. By substituting the form of $\boldsymbol{\iota}$ from (3.3.13) and $\mathbf{U}$ from (3.3.11), $\boldsymbol{\beta}^{(s)}$ can be written as a form of the iteratively weighted least squares (IRLS):

$$\begin{aligned} \boldsymbol{\beta}^{(s)} &= (\mathbf{X'W^{(s-1)}X})^{-1}\left(\mathbf{X'W^{(s-1)}}\left(\mathbf{X}\boldsymbol{\beta}^{\mathbf{(s-1)}} + (\mathbf{y} - \boldsymbol{\mu}^{\mathbf{(s-1)}})\frac{\partial \boldsymbol{\mu}^{\mathbf{(s-1)}}}{\partial \boldsymbol{\eta}^{\mathbf{(s-1)}}}\right)\right) \\ &= (\mathbf{X'W^{(s-1)}X})^{-1}\mathbf{X'W^{(s-1)}z^{(s-1)}}. \end{aligned} \tag{3.3.14}$$

3.3.4 Quasi-Likelihood Estimates for GLM

One drawback of the maximum likelihood approach to obtain the parameter estimate is it heavily relies on the explicit form of the assumed distribution. Thus, its usage is prohibited for a response with abnormal dispersion or that deviates far from the known distribution. Wedderburn [1974] proposed a quasi-likelihood whose structure only depends on the mean $\mu_i(\boldsymbol{\beta})$ and variance $Var_i(\boldsymbol{\beta})$ of the response variable. More surprisingly, he showed that even when the distribution of response y_i is non-normal, the estimate of model parameter $\boldsymbol{\beta}$ can still be asymptotically unbiased if it is obtained by optimizing the log-likelihood from $y_i \sim N(\mu_i(\boldsymbol{\beta}), Var_i(\boldsymbol{\beta}))$.

The quasi-likelihood $Q(\mu_i, \phi)$ is defined as

$$\frac{\partial Q_i(\mu_i, \phi)}{\partial \mu_i}) = \frac{y_i - \mu_i}{Var(y_i)} \tag{3.3.15}$$

where μ_i is the mean response and $Var(y_i) = \phi Var(\mu_i)$ captures the dispersion in the response. It is not difficult to show the quasi-likelihood $Q(\mu_i, \phi)$ enjoys the same appealing statistical properties as log-likelihood $L(\theta, \phi; y_i)$ does. For example, the information matrix $\boldsymbol{\iota}$ constructed using quasi-likelihood has the identical form of (3.3.12), that is

$$\boldsymbol{\iota}_i = -E\left(\frac{\partial^2 Q_i(\mu_i, \phi)}{\partial \beta_j \partial \beta_{j'}}\right) = \frac{x_{ij}x_{ij'}}{Var(y_i)}\left(\frac{\partial \mu_i}{\partial \eta_i}\right)^2 = \frac{\partial \mu_i}{\partial \beta_j}\frac{\partial \mu_i}{\partial \beta_{j'}}\frac{1}{Var(y_i)}. \tag{3.3.16}$$

In matrix notation, the information matrix $\boldsymbol{\iota}$ for n observations can be written as:

$$\boldsymbol{\iota} = \mathbf{D}'\mathbf{V}^{-1}\mathbf{D} \tag{3.3.17}$$

where $\mathbf{D}$ is the $n \times p$ matrix with the $(i,j)^{th}$ entry to be $\frac{\partial \mu_i}{\partial \beta_j}$ and $\mathbf{V}$ is a diagonal matrix with the i^{th} entry to be $Var(y_i)$. The model parameter estimates can be obtained using the iterative optimization method where $\boldsymbol{\beta}$ is updated as $\boldsymbol{\beta}^{(s)} = \boldsymbol{\beta}^{(s-1)} + (\boldsymbol{\iota}^{(s-1)})^{-1}\boldsymbol{U}^{\boldsymbol{(s-1)}}$. By substituting the information matrix $\boldsymbol{\iota}$ using equation (3.3.17) and the gradient $\boldsymbol{U}$ using equation (3.3.9), the estimate of $\boldsymbol{\beta}$ can be again written as a form of the iteratively weighted least squares (IRLS):

$$\boldsymbol{\beta}^{(s)} = \boldsymbol{\beta}^{(s-1)} + (\mathbf{D'^{(s-1)}V^{(s-1)-1}D^{(s-1)}})^{-1}\mathbf{D^{(s-1)\prime}V^{(s-1)-1}}(\mathbf{y} - \boldsymbol{\mu}^{\mathbf{(s-1)}}) \tag{3.3.18}$$

3.4 Generalized Linear Mixed Model

The extension of generalized linear model to incorporate the longitudinal or clustered data usually comes in two flavors: 1) the marginal or population-averaged model; and 2) the random-effect or subject-specific model. The two types of models were developed to answer the underlying question from different perspectives and thus yield different model parameter interpretations. The marginal model focuses on depicting the mean response that only depends on the fixed effect and not on any random effects. On the contrary, the subject-specific model introduces the random effect which allows the mean response to vary across subjects. The two models also differ in the way to measure the correlation among repeated measurements and thus result in different estimation procedures.

3.4.1 Generalized Equation Estimation for Marginal Model

The marginal model, as suggested by its name, emphasizes that the mean response depends only on the fixed-effects of interest and not on the individual heterogeneities. One of the appealing advantages of the marginal model is that a full distribution of the response is often not required. Instead, only the mean and variance of the response are needed to construct the model. This property of the marginal model is exceptional for at least two reasons: 1) only a few tractable distributions exist for discrete responses and the assumed probability density can be easily violated due to dispersion in data, which is especially true for longitudinal or clustered discrete data; 2) the computational complexity of the marginal likelihood for discrete longitudinal or clustered data can be formidable while the estimation procedure for marginal model is relatively standard. We now illustrate the three parts that define a marginal model for discrete longitudinal data:

1. The mean response for the i^{th} subject $\boldsymbol{\mu}_i = E(\mathbf{y_i}|\mathbf{x_i})$ is connected to a set of fixed-effects via a link function $g(\cdot)$ where $g(\boldsymbol{\mu}_i) = \mathbf{x_i}^T\boldsymbol{\beta}$.

2. The variance of the i^{th} subject depends on the variance of the mean response and a dispersion parameter ϕ, that is: $Var(\mathbf{y_i}|\mathbf{x_i}) = \phi v(\boldsymbol{\mu}_i)$.

3. The within-subject association among repeated measurements of subject i can be depicted as a function of a set of additional parameters $\boldsymbol{\alpha}$ and mean response $\boldsymbol{\mu}_i$.

The first two components specify that the mean and variance of the response follow the framework of the GLM which is related to the quasi-likelihood (3.3.15). It is the third component that actually establishes the extension to longitudinal or clustered discrete data by allowing within-subject association for the same individual.

To estimate the model parameters for discrete longitudinal data is always challenging since there is no convenient and simple form of the marginal likelihood of the joint multivariate distribution of the response. The remarkable method, generalized estimating equation (GEE) extended naturally from quasi-likelihood [Wedderburn, 1974] proposed by Liang and Zeger [1986], provides an convenient alternative to the full likelihood approach for parameter estimation. Briefly, the GEE method works by defining the working covariance matrix $\mathbf{V_i}$ of $\mathbf{y_i}$ to be

$$\mathbf{V_i} = \phi \mathbf{A_i^{1/2}} \mathbf{R_i}(\boldsymbol{\alpha}) \mathbf{A_i^{1/2}} \tag{3.4.1}$$

where $\mathbf{A_i}$ is a $n_i \times n_i$ diagonal matrix with the j^{th} entry to be $v(\mu_{ij})$; $\mathbf{R_i}(\boldsymbol{\alpha})$ is a working correlation matrix depended on the parameter $\boldsymbol{\alpha}$ that specifies the within-subject correlation structure. The term 'working' reflects both the covariance and correlation are from the assumed model not the true observations. To solve the GEE, it requires iterating between the quasi-likelihood estimate of $\boldsymbol{\beta}$ (3.3.18) and the estimate of $\boldsymbol{\alpha}$ as a function of $\boldsymbol{\beta}$. It starts with an initial guess of $\mathbf{R_i}(\boldsymbol{\alpha})$, usually an identity matrix or autoregressive structure and an initial estimate of dispersion parameter ϕ. The initial estimate of $\boldsymbol{\beta}$ can be obtained based on the initial estimate of $\mathbf{R_i}(\boldsymbol{\alpha})$ and ϕ. Once a new estimate of $\boldsymbol{\beta}$ is obtained, it is then used to update the estimate of $\mathbf{R_i}(\boldsymbol{\alpha})$ and ϕ. This procedure iterates until convergence.

The parameters estimated from GEE method enjoy several appealing statistical properties. First, the estimator of $\boldsymbol{\beta}$ can be as efficient as the maximum likelihood estimator for continuous and certain discrete longitudinal responses. Second, the estimator is robust and asymptotically unbiased that only requires the mean response to be correct and allow the within-subject correlation to be misspecified. However, under this scenario, the standard error obtained may be largely biased and invalid which requires further improvement.

3.4.2 Penalized Quasi-likelihood for GLMM

In contrast to the marginal model, the second way to extend the GLM to allow longitudinal or clustered data is via the generalized linear mixed model (GLMM), which directly extended from the linear mixed model framework and allows the mean response from the non-normal distribution varying among individuals. As the mean response depends on both the fixed and random effects, the model parameter $\boldsymbol{\beta}$ will have a subject-specific interpretation conditional on the random effect. The GLMM is particularly useful when the individual pattern is of interest and the main research purpose is to investigate the statistical inference of the set of explanatory variables that contribute to the dynamic individual heterogeneities. In addition, the GLMM is irreplaceable when classification and prediction are of primary concern in analyzing longitudinal or clustered data. We now illustrate the framework of GLMM, which is closely related to the nonlinear mixed model discussed in Section 3.2. Suppose $\mu_{ij} = E(y_{ij}|\mathbf{b_i})$ is the conditional mean of the i^{th} object j^{th} measurement. The GLMM is constructed by connecting the conditional mean μ_{ij} to the fixed and random effects via a link function $g(\cdot)$, that is:

$$g(\mu_{ij}) = \mathbf{x_{ij}^T}\boldsymbol{\beta} + \mathbf{z_{ij}}\mathbf{b_i} \tag{3.4.2}$$

where $\boldsymbol{\beta}$ is the fixed effect; and $\mathbf{b_i}$ is the random effect that follows a normal distribution with mean zero and $q \times q$ variance matrix $\mathbf{G_i}$. Similar to the marginal model, the conditional variance $Var(y_{ij}|\mathbf{b_i})$ is a function of conditional mean and dispersion parameter, that is $\mathrm{Var}(\mathrm{y}_{ij}|\mathbf{b_i}) = \phi v(y_{ij}|\mathbf{b_i})$. We further assume the error term $\boldsymbol{\epsilon}_i$ follows a normal distribution $N(0, \mathbf{R_i})$ and correspondingly, the response for the i^{th} subject $\mathbf{y_i}$ can be written as:

$$\mathbf{y_i} = g^{-1}(\mathbf{x_i^T}\boldsymbol{\beta} + \mathbf{z_i}\mathbf{b_i}) + \boldsymbol{\epsilon}_i \tag{3.4.3}$$

Despite the notation of the link function, (3.4.3) has the exact same form as the nonlinear mixed model shown in equation (3.2.2).

The parameter estimates for the GLMM can be obtained primarily through two approaches: marginal likelihood and penalized quasi-likelihood. We have discussed the marginal likelihood approach for the nonlinear mixed model for obtaining the fixed-effects estimates and the empirical Bayes estimates for the random-effects in Section 3.2.2. The implementation of the marginal likelihood approach to GLMM is almost identical, which also requires integration over the distribution of the random effects and the numerical approximation is often used to fill the analytical absence. The marginal likelihood approach provides an accurate estimate of model parameter while the computational process to approximate the integrands can be extremely challenging, especially under the high-dimensional numerical integration scenario. Several authors have proposed alternative methods to avoid the need for computing the approximation of the integrand. Notably, Stiratelli et al. [1984] developed a penalized quasi-likelihood (PQL) method to fit a data with longitudinal dichotomous response, which enhanced and generalized by Green [1987], Breslow and Clayton [1993] and Wolfinger [1993] as a Laplace approximation to the marginal likelihood which considered as a general approach for fitting GLMM. As with most low-dimensional approximation approaches, PQL can yield largely biased estimates of the variance component and model parameters under certain circumstances. Here, we briefly review the version proposed by Lindstrom and Bates [1990] which was originally used to fit the nonlinear mixed model. With proper modification and extension, this algorithm has been implemented to the prevalent statistical software/packages, for example `SAS PROC GLIMMIX` [Wolfinger and O'connell, 1993] and R package `lme4` [D. Bates and Bolker, 2011] for fitting a variety of mixed-effects models including GLMM. To begin, we let $\boldsymbol{\theta}$ be the variance component that accounts for the inter-cluster heterogeneity and intra-cluster homogeneity. In equation (3.4.3), the fixed effects $\boldsymbol{\beta}$

and random effects $\mathbf{b_i}$ are functions of $\boldsymbol{\theta}$. By augmenting the data with a set of 'pseudo-data', it can be shown that $\boldsymbol{\beta}(\boldsymbol{\theta})$ and $\mathbf{b_i}(\boldsymbol{\theta})$ jointly maximize the log pseudo-likelihood, which can be written as:

$$L^*(\boldsymbol{\beta}, \mathbf{b_i}, \boldsymbol{\theta}; \mathbf{x_i}) \propto \log|\mathbf{R_i}| + (\mathbf{y_i} - g^{-1}(\boldsymbol{\eta}_i))^T \mathbf{R_i^{-1}} (\mathbf{y_i} - g^{-1}(\boldsymbol{\eta}_i) + \log|\mathbf{G_i}| + \mathbf{b_i^T} \mathbf{G_i} \mathbf{b_i} + c \tag{3.4.4}$$

Therefore, once the estimate of variance component $\boldsymbol{\theta}$ is obtained, the estimate of $\boldsymbol{\beta}$ and $\mathbf{b_i}$ can be obtained subsequently using the methods for the linear mixed model described in Section 3.1. To obtain the estimate of $\boldsymbol{\theta}$ requires optimizing the marginal likelihood with respect to the response, which has a form of $p(\mathbf{y}) = \int p(\mathbf{y}|\mathbf{b})p(\mathbf{b})d\mathbf{b}$ (3.2.3). The random effect $\mathbf{b_i}$ is nonlinear in the response and yields no closed-form of the marginal likelihood. To accomplish, an approximation of the residual $\boldsymbol{\epsilon}_i = \mathbf{y_i} - g^{-1}(\mathbf{x_i^T}\boldsymbol{\beta} + \mathbf{z_i}\mathbf{b_i})$ near $\hat{\mathbf{b}}_\mathbf{i}$ can be conducted using the first-order Taylor series:

$$\mathbf{y_i} - g^{-1}(\mathbf{x_i^T}\boldsymbol{\beta} + \mathbf{z_i}\mathbf{b_i}) \approx \mathbf{y_i} - g^{-1}(\mathbf{x_i^T}\boldsymbol{\beta} + \mathbf{z_i}\hat{\mathbf{b}}_\mathbf{i}) + \left.\frac{\partial g^{-1}(\boldsymbol{\eta}_i)}{\partial \boldsymbol{\eta}_i}\right|_{\hat{\boldsymbol{\beta}}, \hat{\mathbf{b_i}}} (\hat{\mathbf{b}}_\mathbf{i} - \mathbf{b_i}). \tag{3.4.5}$$

Then the conditional distribution of the residual given random effect $\mathbf{b_i}$ can be approximated as:

$$\left[\mathbf{y_i} - g^{-1}(\mathbf{x_i^T}\boldsymbol{\beta} + \mathbf{z_i}\hat{\mathbf{b}}_\mathbf{i}) + \left.\frac{\partial g^{-1}(\boldsymbol{\eta}_i)}{\partial \boldsymbol{\eta}_i}\right|_{\hat{\boldsymbol{\beta}}, \hat{\mathbf{b_i}}} (\hat{\mathbf{b}}_\mathbf{i} - \mathbf{b_i})\right] \Bigg| \mathbf{b_i} \sim N(\mathbf{0}, \mathbf{R_i}). \tag{3.4.6}$$

It follows the conditional distribution of response $\mathbf{y}$ given random effect $\mathbf{b_i}$ can be approximated as

$$\mathbf{y}|\mathbf{b_i} \;\sim\; N\left(g^{-1}(\mathbf{x_i^T}\boldsymbol{\beta} + \mathbf{z_i}\hat{\mathbf{b}}_\mathbf{i}) - \frac{\partial g^{-1}(\boldsymbol{\eta}_i)}{\partial \boldsymbol{\eta}_i}\bigg|_{\hat{\boldsymbol{\beta}},\hat{\mathbf{b_i}}}(\hat{\mathbf{b}}_\mathbf{i} - \mathbf{b_i}), \mathbf{R_i}\right). \tag{3.4.7}$$

In combination with the distribution of random effect $\mathbf{b_i} \sim N(\mathbf{0}, \mathbf{G_i})$ and using the properties of normal distribution multiplication, the marginal distribution of $\mathbf{y}$ can be approximated as:

$$\mathbf{y} \sim N\left(g^{-1}(\mathbf{x_i^T}\boldsymbol{\beta} + \mathbf{z_i}\hat{\mathbf{b}}_\mathbf{i}) - \frac{\partial g^{-1}(\boldsymbol{\eta}_i)}{\partial \boldsymbol{\eta}_i}\bigg|_{\hat{\boldsymbol{\beta}},\hat{\mathbf{b_i}}} \cdot \hat{\mathbf{b_i}}, \mathbf{V_i}\right) \tag{3.4.8}$$

where $\mathbf{V_i}(\boldsymbol{\theta}) = \mathbf{R_i}(\boldsymbol{\theta}) + \frac{\partial g^{-1}(\boldsymbol{\eta}_i(\boldsymbol{\theta}))}{\partial \boldsymbol{\eta}_i(\boldsymbol{\theta})}\mathbf{G_i}(\boldsymbol{\theta})\big(\frac{\partial g^{-1}(\boldsymbol{\eta}_i(\boldsymbol{\theta}))}{\partial \boldsymbol{\eta}_i(\boldsymbol{\theta})}\big)^T$. By optimizing (3.4.8) using prevalent iterative optimization methods, such as Newton-Raphson, the estimate of $\boldsymbol{\theta}$ can be obtained. As most optimization methods require the second-order derivatives which is potentially computationally cumbersome, further approximating the residual around the fixed-effect $\boldsymbol{\beta}$ can avoid this problem at the cost of sacrificing a proportion of the estimates accuracy. The estimate of $\boldsymbol{\beta}(\hat{\boldsymbol{\theta}})$ and $\mathbf{b_i}(\hat{\boldsymbol{\theta}})$ can be obtained as a function of $\hat{\boldsymbol{\theta}}$ and the updated estimates of the fixed and random effects are then used to obtain a new estimate of $\boldsymbol{\theta}$. This iterative procedure is repeated until convergence.

Chapter 4

Random Coefficient Model with Ordinal Response

The statistical methods suitable for modeling the clustered or longitudinal data with an ordinal response have become increasingly important in a variety of fields. A significant amount of work has been done to model a longitudinal ordinal response through different approaches. Harville and Mee [1984] initiated a mixed model procedure for analyzing clustered data with an ordinal response where the random effects estimates were approximated through a Taylor series expansion. Jansen [1990] utilized the numerical quadrature method to fit the mixed-effect ordinal model with one random effect. Ezzet and Whitehead [1991] implemented the Newton-Raphson procedure to fit a random-effects model with an ordinal response. Hedeker and Gibbons [1994] developed a random-effects model for an ordinal response allowing multiple random effects suitable for clustered or longitudinal data with complex correlation structure. Besides the full likelihood approaches for obtaining the parameter estimates, several authors have contributed by fitting the random-effects ordinal model based on different mechanisms. Notably, Yang [2001] used marginal quasi-likelihood (MQL) and predictive quasi-likelihood (PQL) to obtain the parameter estimates which is with less computationally demanding but could be subject to larger bias. In this chapter, we primarily review the random coefficient method for modeling an ordinal response as discussed by Hedeker and Gibbons [1994].

Chapter 4 is organized as follows: In Section 4.1, we focus on the random coefficient model for modeling longitudinal data having an ordinal response. We then present the marginal likelihood for the ordinal random coefficient model and derive its approximation using traditional and adaptive Gauss-Hermite Quadrature methods in Section 4.2. Estimates of model parameters and random effects are described in Sections 4.3 and 4.4, respectively. In Section 4.5, we revisit the NIMH Schizophrenia Example introduced in Chapter 1. An additional example, the Health Services Research Example is reproduced to demonstrate the usage of the ordinal random coefficient model in Section 4.6. The parameter estimates from the two examples were compared using several available computational softwares/packages.

4.1 Random Coefficient Model with Ordinal Response

As discussed in Section 3.1, the random coefficient model has its best usage in scenarios when time-dependent repeated measurements are collected where the random term in the model can capture the subject-specific variations in data. There are two types of the random coefficient models: 1) the random intercept model, where only the intercept is assumed to be subject-specific; 2) the random coefficient model, where both the intercept and slope vary across subjects. The ordinal random coefficient model is constructed by adding the additional subject-specific random effects $\mathbf{u_i}$ to the traditional ordinal model under the proportional odds assumption. We exemplify the ordinal random coefficient model for the i^{th} subject using the cumulative logit link $\log\left(\frac{\gamma_{ic}}{1-\gamma_{ic}}\right)$, where $\log\left(\frac{\gamma_{ic}}{1-\gamma_{ic}}\right)$(4.1.1) is a matrix of dimension $n_i \times C$ and n_i is the number of repeated measurements in subject i.

$$\log\left(\frac{\boldsymbol{\gamma}_{ic}}{1-\boldsymbol{\gamma}_{ic}}\right) = \begin{pmatrix} \log\frac{\gamma_{i11}}{1-\gamma_{i11}} & \cdots & \log\frac{\gamma_{i1c}}{1-\gamma_{i1c}} & \cdots & \log\frac{\gamma_{i1C}}{1-\gamma_{i1C}} \\ \log\frac{\gamma_{i21}}{1-\gamma_{i21}} & \cdots & \log\frac{\gamma_{i2c}}{1-\gamma_{i2c}} & \cdots & \log\frac{\gamma_{i2C}}{1-\gamma_{i2C}} \\ \vdots & & \vdots & & \vdots \\ \log\frac{\gamma_{i,n_i1}}{1-\gamma_{i,n_i1}} & \cdots & \log\frac{\gamma_{i,n_ic}}{1-\gamma_{i,n_ic}} & \cdots & \log\frac{\gamma_{i,n_iC}}{1-\gamma_{i,n_iC}} \end{pmatrix}_{n_i \times C} \tag{4.1.1}$$

Each of the elements in the matrix (4.1.1) can be linked to the implicit expression of the intercept, fixed effects and random effects as shown in (4.1.2) which defines the ordinal random coefficient model for the j^{th} measurement on the i^{th} subject.

$$\log\left(\frac{\gamma_{ijc}}{1-\gamma_{ijc}}\right) = \log\left(\frac{P(Y_{ij} \leq c|\mathbf{x_{ij}}, \mathbf{u_i})}{P(Y_{ij} > c|\mathbf{x_{ij}}, \mathbf{u_i})}\right) = \alpha_c + \mathbf{x_{ij}^T}\boldsymbol{\beta} + \mathbf{z_i}\mathbf{u_i} \tag{4.1.2}$$

where α_c denotes the category-specific intercept; $\boldsymbol{\beta}$ is a $p \times 1$ vector of coefficients associated with explanatory variables $\mathbf{x_{ij}}$ of dimension $p \times 1$. For the random intercept model, $\mathbf{z_i}$ is an indicator denoting the intercept and $\mathbf{u_i}$ is a random variable that follows a univariate normal distribution $N(0, \sigma^2_{int})$. For the random coefficient model, $\mathbf{z_i}$ is a 1×2 design matrix that includes the intercept and the time points when the j^{th} measurement was taken for the i^{th} subject. Correspondingly, $u_i = (u_{1i}, u_{2i})$ is a vector that follows a bivariate normal distribution with mean $\mathbf{0}$ and variance $\mathbf{G_i}$, where $\mathbf{G_i} = \begin{pmatrix} \sigma^2_{u_1} & \sigma_{u_1,u_2} \\ \sigma_{u_1,u_2} & \sigma^2_{u_2} \end{pmatrix}$. From (4.1.2), we can calculate the probabilities $\pi_c(\mathbf{x_{ij}}, \mathbf{u_i})$ that represents the j^{th} measurement of subject i falls into the c^{th} categories:

$$\begin{aligned} \pi_c(\mathbf{x_{ij}}, \mathbf{u_i}) &= P(Y_{ij} \leq c|\mathbf{x_{ij}}, \mathbf{u_i}) - P(Y_{ij} \leq c-1|\mathbf{x_{ij}}, \mathbf{u_i}) \\ &= \frac{\exp(\alpha_c + \mathbf{x^T_{ij}}\boldsymbol{\beta} + \mathbf{z_i u_i})}{1+\exp(\alpha_c + \mathbf{x^T_{ij}}\boldsymbol{\beta} + \mathbf{z_i u_i})} - \frac{\exp(\alpha_{c-1} + \mathbf{x^T_{ij}}\boldsymbol{\beta} + \mathbf{z_i u_i})}{1+\exp(\alpha_{c-1} + \mathbf{x^T_{ij}}\boldsymbol{\beta} + \mathbf{z_i u_i})}. \end{aligned} \tag{4.1.3}$$

It is also worth noting that the error variance in the ordinal random coefficient model is fixed and its value can be obtained using a latent variable model approach for logistic regression model. Suppose y^* is a continuous latent variable , y is an observed dichotomous response and γ is a threshold where $y = 0$ if $y^* \leq \gamma$ and $y = 1$ otherwise. We assume the latent variable model can be written as: $\mathbf{y_i} = \mathbf{x_i}^T\boldsymbol{\beta} + \epsilon_i$. According to [Hedeker and Gibbons, 2006], in a logistic regression setting, ϵ_i is assumed to follow a standard logistic distribution with mean 0 and variance $\pi^2/3$. Since the ordinal random coefficient model (4.1.2) has the same structure as a logistic regression model, it can be assumed the variance in ordinal random coefficient model is fixed at $\pi^2/3$. For a random intercept model, the intra-subject correlation can be calculated as $\frac{\sigma^2}{\sigma^2+\pi^2/3}$, where σ^2 is the variance explains individual heterogeneity.

4.2 The Marginal Likelihood and its Approximation

Estimating the unknown parameters in an ordinal random coefficient model is challenging. We review the marginal likelihood approach discussed in [Hedeker and Gibbons, 1994] and derive the form of the marginal likelihood for the random intercept ordinal model and the random coefficient ordinal model approximated by traditional and adaptive Gauss-Hermite Quadrature methods, respectively.

For the ordinal random intercept model, since the random effect u_i follows a univariate normal distribution $N(0, \sigma^2_{int})$, the likelihood function for subject i with n_i measurements can be written as:

$$L_i(\boldsymbol{\alpha}, \boldsymbol{\beta}, \mathbf{u_i}, \sigma^2_{int}|\mathbf{x_i}) = \frac{1}{\sqrt{2\pi\sigma^2_{int}}} \exp\bigg(- \frac{u_i^2}{2\sigma^2_{int}}\bigg) \prod_{j=1}^{n_i}\prod_{c=1}^{C} \pi_c(\mathbf{x_{ij}}, u_i)^{y_{ijc}} \tag{4.2.1}$$

where $(y_{ij1}, \cdots, y_{ijC})$ is an indicator vector with $y_{ijc} = 1$ if Y_{ij}, which is the j^{th} measurement on subject i, belongs to the c^{th} category and 0 otherwise. According to equation (3.2.3), the marginal likelihood for the i^{th} subject, which eliminates the random effects can be written as:

$$\begin{aligned} L_i^*(\boldsymbol{\alpha}, \boldsymbol{\beta}, \sigma^2_{int}|\mathbf{x_i}) &= \int L_i(\boldsymbol{\alpha}, \boldsymbol{\beta}, \mathbf{u_i}, \sigma^2_{int}|\mathbf{x_i}) d\mathbf{u_i} \\ &= \frac{1}{\sqrt{2\pi\sigma^2_{int}}} \int \exp\bigg(- \frac{u_i^2}{2\sigma^2_{int}}\bigg) \prod_{j=1}^{n_i}\prod_{c=1}^{C} \pi_c(\mathbf{x_{ij}}, u_i)^{y_{ijc}} du_i \qquad (4.2.2) \\ &= \frac{1}{\sqrt{2\pi\sigma^2_{int}}} \int \exp\bigg(- \frac{u_i^2}{2\sigma^2_{int}} + \log \prod_{j=1}^{n_i}\prod_{c=1}^{C} \pi_c(\mathbf{x_{ij}}, u_i)^{y_{ijc}}\bigg) du_i. \qquad (4.2.3) \end{aligned}$$

As u_i is nonlinear in the response, no closed-form for this integral can be directly obtained. To solve this problem, we used two numerical integration methods to approximate the integral: nonadaptive and adaptive Gauss-Hermite Quadrature approximation discussed in Section 3.2. Let (z_i^*, w_i) be the set of abscissas and weights from N^{th} order Hermite polynomial as shown in [Abramowitz and Stegun, 1972]. To simplify the notation, we denote $g(\boldsymbol{\alpha}, \boldsymbol{\beta}, \mathbf{x_i}, \mathbf{y_{ijc}}, \mathbf{u_i}, \sigma^2_{int}) = -\frac{u_i^2}{2\sigma^2_{int}} + \log \prod_{j=1}^{n_i} \prod_{c=1}^{C} \pi_c(\mathbf{x_{ij}}, u_i)^{y_{ijc}}$. Then the marginal likelihood for the ordinal random intercept model can be approximated using Gauss-Hermite Quadrature method as:

$$\begin{aligned} L_i^*(\boldsymbol{\alpha}, \boldsymbol{\beta}, \sigma^2_{int}|\mathbf{x_i}) &= \frac{1}{\sqrt{2\pi\sigma^2_{int}}} \int \exp\big(g(\boldsymbol{\alpha}, \boldsymbol{\beta}, \mathbf{x_i}, \mathbf{y_{ijc}}, u_i, \sigma^2_{int})\big) du_i \\ &= \frac{1}{\sqrt{\pi}} \int \exp\left(g(\boldsymbol{\alpha}, \boldsymbol{\beta}, \sigma^2_{int}, \mathbf{x_i}, \mathbf{y_{ijc}}, \mathbf{z_i^*}) + \frac{\|\mathbf{z_i^*}\|^2}{2}\right) \exp\left(-\frac{\|\mathbf{z_i^*}\|^2}{2}\right) d\mathbf{z_i^*} \\ &= \frac{1}{\sqrt{\pi}} \sum_i^{N_{GQ}} \exp\big(g(\boldsymbol{\alpha}, \boldsymbol{\beta}, \sigma^2_{int}, \mathbf{x_i}, \mathbf{y_{ijc}}, \mathbf{z_i^*})\big) \mathbf{W_i} \qquad (4.2.4) \end{aligned}$$

where $\mathbf{W_i} = \exp(\|\mathbf{z_i^*}\|^2) \prod_i w_i$. Correspondingly, the log-likelihood for each subject can be approximated as:

$$\log L_i^*(\boldsymbol{\alpha}, \boldsymbol{\beta}, \sigma^2_{int}|\mathbf{x_i}) = -\frac{1}{2}\log \pi + \log\left(\sum_i^{N_{GQ}} \exp\big(g(\boldsymbol{\alpha}, \boldsymbol{\beta}, \sigma^2_{int}, \mathbf{x_i}, \mathbf{y_{ijc}}, \mathbf{z_i^*})\big) \mathbf{W_i}\right) \qquad (4.2.5)$$

and the marginal log likelihood for all subjects can be written as:

$$\log L^*(\boldsymbol{\alpha}, \boldsymbol{\beta}, \sigma^2_{int}|\mathbf{x_i}) = -\frac{N}{2}\log \pi + \sum_i^{N} \log\left(\sum_i^{N_{GQ}} \exp\big(g(\boldsymbol{\alpha}, \boldsymbol{\beta}, \sigma^2_{int}, \mathbf{x_i}, \mathbf{y_{ijc}}, \mathbf{z_i^*})\big) \mathbf{W_i}\right). \qquad (4.2.6)$$

We then use the adaptive Gauss-Hermite Quadrature approximation method to approximate the marginal likelihood. For the adaptive procedure, we assume the integrand $\exp\big(g(\boldsymbol{\alpha}, \boldsymbol{\beta}, \sigma^2_{int}, \mathbf{x_i}, \mathbf{y_{ijc}}, \mathbf{z_i^*})\big)$ in equation (4.2.4) can be approximated by a normal distribution $N(\hat{u_i}, f^{-1}(\boldsymbol{\alpha}, \boldsymbol{\beta}, \sigma^2_{int}, \mathbf{x_i}, \mathbf{y_{ijc}}, u_i))$, where $\hat{u_i}$ is the Empirical Bayes estimate of u_i as well as the mode of function $f(\cdot)$ defined in equation (4.2.7). Let $u_i = \hat{u_i} + \sqrt{2} f''(\boldsymbol{\alpha}, \boldsymbol{\beta}, \sigma^2_{int}, \mathbf{x_i}, \mathbf{y_{ijc}}, u_i)^{-1/2} z_i^*$

and $du_i = \sqrt{2} f''(\boldsymbol{\alpha}, \boldsymbol{\beta}, \sigma^2_{int}, \mathbf{x_i}, \mathbf{y_{cij}}, u_i)^{-1/2} dz_i^*$, where $f''(\cdot, u_i)^{-1/2}$ is a square root of $f''(\cdot, u_i)^{-1}$ if $f''(\cdot, u_i)^{-1}$ is a scale and the Cholesky decomposition of $f''(\cdot, u_i)^{-1}$ if $f''(\cdot, u_i)^{-1}$ is a matrix. When function $f(\cdot, u_i)$ has a rather complicated form, the exact form of the second derivative with respect to the random effect can be incredibly burdensome. Therefore, an approximate form of the second derivative is often used to reduce the amount of calculation and accelerate computational speed. In R, the second derivative of function $f(\cdot, u_i)$ can be obtained by extracting the Hessian matrix when optimizing the function. As the approximation of Hessian matrix varies from optimization algorithms, the result of numerical integration through adaptive Gauss-Hermite quadrature can be slightly different.

$$\begin{aligned} \hat{u_i} &= \arg\min - \log\left(\exp\left(g(\boldsymbol{\alpha}, \boldsymbol{\beta}, \sigma^2_{int}, \mathbf{x_i}, \mathbf{y_{ijc}}, u_i)\right)\right) \\ &= \arg\min f(\boldsymbol{\alpha}, \boldsymbol{\beta}, \sigma^2_{int}, \mathbf{x_i}, \mathbf{y_{ijc}}, u_i) \qquad (4.2.7) \end{aligned}$$

Then the marginal likelihood for the ordinal random intercept with ordinal response can be approximated using adaptive Gauss-Hermite Quadrature method as:

$$\begin{aligned} L_i^*(\boldsymbol{\alpha}, \boldsymbol{\beta} | \mathbf{x_i}, \sigma^2_{int}) &= \frac{1}{\sqrt{2\pi\sigma^2_{int}}} \int \exp\left(g(\boldsymbol{\alpha}, \boldsymbol{\beta}, \sigma^2_{int}, \mathbf{x_i}, \mathbf{y_{ijc}}, u_i, \sigma^2_{int})\right) du_i \\ &= \frac{1}{\sqrt{\pi\sigma^2_{int} f_i''(\cdot, u_i)}} \int \exp\left(g(\cdot, \hat{u_i} + \sqrt{2} f_i''^{-1/2}(\cdot, u_i)\mathbf{z_i^*}) + \frac{\|\mathbf{z_i^*}\|^2}{2}\right) \exp\left(-\frac{\|\mathbf{z_i^*}\|^2}{2}\right) d\mathbf{z_i^*} \\ &= \frac{1}{\sqrt{\pi\sigma^2_{int} f_i''(\cdot, u_i)}} \sum_{i}^{N_{GQ}} \exp\left(g(\boldsymbol{\alpha}, \boldsymbol{\beta}, \sigma^2_{int}, \mathbf{x_i}, \mathbf{y_{ijc}}, \hat{u_i} + \sqrt{2} f_i''^{-1/2}(\cdot, u_i)\mathbf{z_i^*})\right) \mathbf{W_i} \qquad (4.2.8) \end{aligned}$$

where $\mathbf{W_i} = \exp(\|\mathbf{z_i^*}\|^2) \prod_i w_i$. Correspondingly, the marginal log likelihood for each subject

i can be approximated as:

$$\log L_i^* = -\frac{1}{2}\log(\pi\sigma_{int}^2 f_i''(\cdot, u_i)) + \log\left(\sum_i^{N_{GQ}} \exp\left(g(\boldsymbol{\alpha}, \boldsymbol{\beta}, \sigma_{int}^2, \mathbf{x_i}, \mathbf{y_{ijc}}, \hat{u}_i + \sqrt{2} f_i''^{-1/2}(\cdot, u_i)\mathbf{z_i^*})\right)\mathbf{W_i}\right)$$

and the marginal log likelihood for all subjects can be approximated as:

$$\begin{aligned}\log L^* = &-\frac{N}{2}\log(\pi\sigma_{int}^2) + \sum_i \log f_i''(\cdot, u_i) + \\ &\log\left(\sum_i^{N_{GQ}} \exp\left(g(\boldsymbol{\alpha}, \boldsymbol{\beta}, \sigma_{int}^2, \mathbf{x_i}, \mathbf{y_{ijc}}, \hat{u}_i + \sqrt{2} f_i''^{-1/2}(\cdot, u_i)\mathbf{z_i^*})\right)\mathbf{W_i}\right).\end{aligned} \tag{4.2.9}$$

For the ordinal random coefficient model, the random effect u_i follows a bivariate normal distribution $N(\mathbf{0}, \mathbf{G_i})$ with $\mathbf{G_i}$ is an unstructured variance matrix. We let u_{1i} denote the subject-specific intercept and u_{2i} denote the subject-specific slope; σ_{u1}^2 and σ_{u2}^2 are the variance components, respectively; and $\rho = \frac{\sigma_{u_1u_2}}{\sigma_{u1}\sigma_{u2}}$ captures the correlation between u_1 and u_2. Then the likelihood function for subject i with n_i measurements can be written as:

$$\begin{aligned}L_i(\boldsymbol{\alpha}, \boldsymbol{\beta}, \mathbf{G_i}, u_{1i}, u_{2i}|\mathbf{x_i}) = &\frac{1}{2\pi\sigma_{u1}\sigma_{u2}\sqrt{1-\rho^2}} \exp\left(-\frac{1}{2(1-\rho^2)}\left(\frac{u_{1i}^2}{\sigma_{u1}^2} + \frac{u_{2i}^2}{\sigma_{u2}^2} - \frac{2\rho u_{1i}u_{2i}}{\sigma_{u1}\sigma_{u2}}\right)\right)\times \\ &\prod_{j=1}^{n_j}\prod_{c=1}^{C} \pi_c(u_{1i}, u_{2i}, \mathbf{x_i})^{y_{ijc}}.\end{aligned} \tag{4.2.10}$$

As a special case, when u_1 and u_2 are independent implies $\rho = 0$, equation (4.2.10) can be simplified as:

$$L_i(\boldsymbol{\alpha}, \boldsymbol{\beta}, \mathbf{G_i}, u_{1i}, u_{2i}|\mathbf{x_i}) = \frac{1}{2\pi\sigma_{u1}\sigma_{u2}} \exp\left(-\frac{u_{1i}^2}{2\sigma_{u1}^2} - \frac{u_{2i}^2}{2\sigma_{u2}^2}\right)\prod_{j=1}^{n_j}\prod_{c=1}^{C} \pi_c(u_{1i}, u_{2i}, \mathbf{x_i})^{y_{ijc}}. \tag{4.2.11}$$

Then the marginal likelihood for the i^{th} subject, which eliminates the random effects can be written as:

$$L_i^*(\boldsymbol{\alpha}, \boldsymbol{\beta}, \mathbf{G_i}|\mathbf{x_i}) = \int\int L_i(\boldsymbol{\alpha}, \boldsymbol{\beta}, \mathbf{G_i}, u_{1i}, u_{2i}|\mathbf{x_i}) du_{1i} du_{2i}. \qquad (4.2.12)$$

To approximate the marginal likelihood function using numerical approximation, we let (z_i^*, w_i) be the set of abscissas and weights from the N^{th} order Hermite polynomial [Abramowitz and Stegun, 1972]. Let $u_1 = \sqrt{2}\sigma_{u_1} z_i^*$, $u_2 = \sqrt{2}\sigma_{u_2} z_i^*$ and $du_1 = \sqrt{2}\sigma_{u_1} dz_i^*$, $du_2 = \sqrt{2}\sigma_{u_2} dz_i^*$. To simply the notation, let $h(u_{1i}, u_{2i}) = \frac{1}{2(1-\rho^2)}\left(\frac{u_{2}{}_{1i}}{\sigma_{u1}^2} + \frac{\mathbf{u_{2i}}^2}{\sigma_{u2}^2} - \frac{2\rho \mathbf{u_{1i}} \mathbf{u_{2i}}}{\sigma_{u1}\sigma_{u2}}\right)$, then the marginal likelihood for the ordinal random coefficient model can be approximated using Gauss-Hermite Quadrature method as:

$$\begin{aligned}
L_i^*(\boldsymbol{\alpha}, \boldsymbol{\beta}, \mathbf{G_i}|\mathbf{x_i}) &\simeq \int\int \exp\left(\log \prod_{j=1}^{n_j}\prod_{c=1}^{C} (\pi_c(\sqrt{2}\sigma_{u_1}\mathbf{z_i}^*, u_{2i})^{y_{ijc}} - h(\sqrt{2}\sigma_{u_1}\mathbf{z_i}^*, u_{2i})\right) dz_i^* du_{2i} \\
&= \int \sum_{i}^{N_{GQ}} \exp\left(\log \prod_{j=1}^{n_j}\prod_{c=1}^{C} (\pi_c(\sqrt{2}\sigma_{u_1}\mathbf{z_i}^*, u_{2i})^{y_{ijc}} - h(\sqrt{2}\sigma_{u_1}\mathbf{z_i}^*, u_{2i})\right) \mathbf{W_i} du_{2i} \\
&= \sum_{i}^{N_{GQ}} \left(\sum_{i}^{N_{GQ}} \exp\left(\log \prod_{j=1}^{n_j}\prod_{c=1}^{C} (\pi_c(\sqrt{2}\sigma_{u_1}\mathbf{z_i}^*, \sqrt{2}\sigma_{u_2}\mathbf{z_i}^*)^{y_{ijc}} - \right.\right. \\
&\left.\left. h(\sqrt{2}\sigma_{u_1}\mathbf{z_i}^*, \sqrt{2}\sigma_{u_2}\mathbf{z_i}^*)\right) \mathbf{W_i}\right) \mathbf{W_i}
\end{aligned}$$

and the log-likelihood for all subjects can be written as:

$$\log L^*(\boldsymbol{\alpha}, \boldsymbol{\beta}, \mathbf{G_i}|\mathbf{x_i}) = -N \log \pi\sqrt{1-\rho^2} + \sum_{i}^{N} \log L_i^*(\boldsymbol{\alpha}, \boldsymbol{\beta}, \mathbf{G_i}|\mathbf{x_i}). \qquad (4.2.13)$$

To approximate the marginal likelihood for the random coefficient model with ordinal response using adaptive Gauss-Hermite Quadrature method, we denote $u_{1i} = \hat{u}_{1i} + \sqrt{2}|f''_{1i}(\cdot, u_{i1})|^{-1/2}z_i^*$, $u_{2i} = \hat{u}_{2i} + \sqrt{2}|f''_{2i}(\cdot, u_{2i})|^{-1/2}z_i^*$ and $du_1 = \sqrt{2}|f''_{1i}(\cdot, u_{1i})|^{-1/2}dz_i^*$, $du_2 = \sqrt{2}|f''_{2i}(\cdot, u_{2i})|^{-1/2}dz_i^*$. Let $g(\boldsymbol{\alpha}, \boldsymbol{\beta}, \mathbf{G_i}, \mathbf{x_i}, \mathbf{y_{ijc}}, u_{1i}, u_{2i}) = -\frac{u_{1i}^2}{2\sigma_{u_1}^2} - \frac{u_{2i}^2}{2\sigma_{u_2}^2} + \log \prod_{j=1}^{n_j} \prod_{c=1}^{C} \pi_c^{y_{ijc}}(u_{1i}, u_{2i})$ and $f(\cdot, u_{1i}, u_{2i}) = -\log(\exp(g(\cdot, u_{1i}, u_{2i})))$. $\hat{u}_{1i}$ and $\hat{u}_{2i}$, which are the Empirical Bayes estimate of u_{1i} and u_{2i} can be obtained from equation (4.2.14) simultaneously.

$$\begin{aligned}(\hat{u}_{1i}, \hat{u}_{2i}) &= \arg\min -\log(\exp(g(\boldsymbol{\alpha}, \boldsymbol{\beta}, \mathbf{G_i}, \mathbf{x_i}, \mathbf{y_{ijc}}, u_{1i}, u_{2i}))) \\ &= \arg\min f(\boldsymbol{\alpha}, \boldsymbol{\beta}, \mathbf{G_i}, \mathbf{x_i}, \mathbf{y_{ijc}}, u_{1i}, u_{2i}) \qquad (4.2.14)\end{aligned}$$

$f''_{1i}(\cdot, u_{1i})$ is the second derivative of $f(\cdot, u_{1i}, u_{2i})$ with respect to u_{1i} and $f''_{2i}(\cdot, u_{2i})$ is the second derivative of $f(\cdot)$ with respect to u_{2i}. Equivalently, $f''_{1i}(\cdot, u_{1i})$ and $f''_{2i}(\cdot, u_{2i})$ are the diagonal of the Hessian matrix of function $f(\cdot, u_{1i}, u_{2i})$. Since the exact form of the second-order derivative of function $f(\cdot, u_{1i}, u_{2i})$ can be quite complex, an approximated form of the Hessian matrix is often applied. We also denote $h(u_{1i}, u_{2i}) = \frac{1}{2(1-\rho^2)}\left(\frac{{u_{1i}}^2}{\sigma_{u_1}^2} + \frac{{u_{2i}}^2}{\sigma_{u_2}^2} - \frac{2\rho u_{1i}u_{2i}}{\sigma_{u_1}\sigma_{u_2}}\right)$.

Then the marginal likelihood can be approximated as

$$L_i^*(\boldsymbol{\alpha},\boldsymbol{\beta},\mathbf{G_i}|\mathbf{x_i}) \simeq \int\int \exp\Bigg(\log\prod_{j=1}^{n_j}\prod_{c=1}^{C}(\pi_c^{y_{ijc}}(\hat{u_1}+\sqrt{2}|f_{1i}^{''}(\cdot,u_{1i})|^{-1/2}z_i^*,u_{2i})- h(\hat{u_1}+\sqrt{2}|f_{1i}^{''}(\cdot,u_{1i})|^{-1/2}z_i^*,u_{2i})\Bigg)\,dz_i^* du_{2i}$$

$$= \int\sum_{i}^{N_{GQ}}\exp\Bigg(\log\prod_{j=1}^{n_j}\prod_{c=1}^{C}\pi_c^{y_{ijc}}(\hat{u_1}+\sqrt{2}|f_{1i}^{''}(\cdot,u_{1i})|^{-1/2}z_i^*,u_{2i})- h(\hat{u_1}+\sqrt{2}|f_{1i}^{''}(\cdot,u_{i1})|^{-1/2}z_i^*,u_{2i})\Bigg)\mathbf{W_i}\,du_{2i}$$

$$= \sum_{i}^{N_{GQ}}\Bigg(\sum_{i}^{N_{GQ}}\exp\Bigg(\log\prod_{j=1}^{n_j}\prod_{c=1}^{C}\pi_c^{y_{ijc}}(\hat{u_1}+\sqrt{2}|f_{1i}^{''}(\cdot,u_{1i})|^{-1/2}z_i^*,\hat{u_2}+\sqrt{2}|f_{2i}^{''}(\cdot,u_{2i})|^{-1/2}z_i^*)-h(\hat{u_1}+ \sqrt{2}|f_{1i}^{''}|(\cdot,u_{1i})^{-1/2}z_i^*,\hat{u_2}+\sqrt{2}|f_{2i}^{''}(\cdot,u_{2i})|^{-1/2}z_i^*)\Bigg)\mathbf{W_i}\Bigg)\mathbf{W_i}$$

Correspondingly, the log-likelihood for all subjects can be approximated as:

$$\log L^*(\boldsymbol{\alpha},\boldsymbol{\beta},\mathbf{G_i}|\mathbf{x_i}) = -N\log\pi - \frac{N}{2}\log(\sigma_{u_1}^2\sigma_{u_2}^2(1-\rho^2)) - \frac{1}{2}\sum_{i}^{N}\log|f_{1i}^{''}(\cdot,u_{1i})| - \frac{1}{2}\sum_{i}^{N}\log|f_{2i}^{''}(\cdot,u_{2i})| + \sum_{i}^{N}\log L_i^*(\boldsymbol{\alpha},\boldsymbol{\beta},\mathbf{G_i}|\mathbf{x_i}). \tag{4.2.15}$$

In a special case when there is no correlation between subject-specific intercept and slope indicating $\rho = 0$, equation 4.2.15 can be simplified as:

$$\log L^*(\boldsymbol{\alpha},\boldsymbol{\beta},\mathbf{G_i}|\mathbf{x_i}) = -N\log\pi + \sum_{i}^{N}\log L_i^*(\boldsymbol{\alpha},\boldsymbol{\beta},\mathbf{G_i}|\mathbf{x_i}) \tag{4.2.16}$$

4.3 Estimating Model Parameters

The parameter estimates can be directly obtained by optimizing the approximated marginal likelihood function using general-purpose optimization method. We fit the ordinal random coefficient model using two statistical software/packages: SAS version 9.2 and R version 2.13.1. In SAS, we fit the model using `PROC NLMIXED` and `PROC GLIMMIX` with the default Quasi-Newton algorithm. In R version 2.13.1, we wrote our own code to optimize the marginal likelihood approximated using both nonadaptive and adaptive Gauss-Hermite Quadrature methods. The optimization methods used include the quasi-Newton method L-BFGS-B which can handle simple box constraints in variables, implemented to R function `optim`, as well as the BFGS method implemented to the R function `optimx` [Nash and Varadhan, 2012] with extended optimization capabilities. The `ordinal` package [Christensen, 2012] is capable of fitting cumulative logit ordinal model with random effects so we also fit ordinal random intercept models using it.

Since the unknown parameters to be estimated are $(\alpha_c, \boldsymbol{\beta}, \sigma^2_{int})$ for the random intercept model and $(\alpha_c, \boldsymbol{\beta},\sigma^2_{u_1}, \sigma^2_{u_2}, \sigma_{u_1u_2})$ for the random coefficient model, it is of ultimate importance to assure the variance components $\sigma^2_{int}, \sigma^2_{u_1}, \sigma^2_{u_2}$ remain nonnegative during each iteration of the optimization process. For the random coefficient model specifically, the covariance term $\sigma_{u_1u_2}$ has to be constrained between $-\sqrt{\sigma^2_{u_1}, \sigma^2_{u_2}}$ and $\sqrt{\sigma^2_{u_1}, \sigma^2_{u_2}}$ since the correlation coefficient $\rho = \frac{\sigma_{u_1u_2}}{\sqrt{\sigma^2_{u_1}, \sigma^2_{u_2}}} \in [-1, 1]$. To ensure both constraints are satisfied during the optimization process, we implement the Cholesky-Banachiewicz decomposition to the variance matrix $\mathbf{G_i}$. We illustrate the usefulness of Cholesky decomposition in the random coefficient model with unstructured variance matrix $\mathbf{G_i}$. According to Cholesky decomposition, the positive-definite variance matrix $\mathbf{G_i}$ can be decomposed into the product of a lower triangular matrix and its transpose as shown in (4.3.1)

$$\mathbf{G_i} = \mathbf{LL^T} = \begin{pmatrix} \sigma^2_{u_1} & \sigma_{u_1,u_2} \\ \sigma_{u_1,u_2} & \sigma^2_{u_2} \end{pmatrix} = \begin{pmatrix} L_1 & 0 \\ L_2 & L_3 \end{pmatrix} \begin{pmatrix} L_1 & L_2 \\ 0 & L_3 \end{pmatrix} \tag{4.3.1}$$
$$= \begin{pmatrix} L_1^2 & L_1L_2 \\ L_1L_2 & L_2^2 + L_3^2 \end{pmatrix}$$

For a special case if the correlation coefficient $\rho = 0$, the Cholesky decomposition can be conducted as equation (4.3.2).

$$\mathbf{G_i} = \mathbf{LL^T} = \begin{pmatrix} \sigma^2_{u_1} & 0 \\ 0 & \sigma^2_{u_2} \end{pmatrix} = \begin{pmatrix} L_1 & 0 \\ 0 & L_3 \end{pmatrix} \begin{pmatrix} L_1 & 0 \\ 0 & L_3 \end{pmatrix} = \begin{pmatrix} L_1^2 & 0 \\ 0 & L_3^2 \end{pmatrix} \tag{4.3.2}$$

We let $L_1 = \exp(ll_1)$ and $L_3 = \exp(ll_3)$ to ensure the positivity of σ_{u1} and σ_{u2}. From equation (4.3.1), the correlation coefficient can be illustrated as $\rho = \frac{L_1L_2}{L_1\sqrt{L_2^2+L_3^2}}$ and $|L_2| \leq \sqrt{L_2^2 + L_3^2}$ which ensures ρ is always within the proper range.

The standard error of the estimated variance components $\hat{\sigma}^2_{u1}, \hat{\sigma}^2_{u2}$ and $\hat{\sigma}_{u_1,u_2}$ can be approximated using Delta method. Suppose a statistic T_n satisfying $\sqrt{n}[T_n - \theta] \to N(0, \sigma^2)$, then any function of statistic T_n, $g(T_n)$ satisfies $\sqrt{n}[g(T_n) - g(\theta)] \to N(0, \sigma^2[g'(\theta)]^2)$ where the standard error of $g(T_n)$ is asymptotically equal to $g'(\theta)^2 \frac{\sigma^2}{\sqrt{n}}$. Then from Cholesky decomposition of $\mathbf{G_i}$ (4.3.1), the standard error of $\hat{\sigma}^2_{u_1} = \exp(ll_1)^2$ can be approximated as:

$$\mathrm{SE}(\hat{\sigma}^2_{u_1}) = \frac{d\exp(ll_1)^2}{d^2 ll_1} \times \mathrm{SE}(ll_1). \tag{4.3.3}$$

Also from equation (4.3.1), the variance $\hat{\sigma}^2_{u_2}$ consists of elements L_2 and $ll3$. We denote $\hat{\sigma}^2_{u_2} = g(L_2, ll_3) = L_2^2 + \exp(ll_3)^2$, the variance of $\hat{\sigma}^2_{u_2}$ can be obtained by:

$$\mathrm{Var}(\hat{\sigma}^2_{u_2}) = \left(\frac{\partial g(\cdot)}{\partial L_2}, \frac{\partial g(\cdot)}{\partial ll_3}\right) \begin{pmatrix} \mathrm{Var}(L_2^2) & \mathrm{Cov}(L_2, ll_3) \\ \mathrm{Cov}(L_2, ll_3) & \mathrm{Var}(ll_3^2) \end{pmatrix} \begin{pmatrix} \frac{\partial g(\cdot)}{\partial L_2} \\ \frac{\partial g(\cdot)}{\partial ll_3} \end{pmatrix}. \tag{4.3.4}$$

Correspondingly, the standard error of $\hat{\sigma}^2_{u_2}$ can be calculated as $\mathrm{SE}(\hat{\sigma}^2_{u_2}) = \sqrt{\mathrm{Var}(\hat{\sigma}^2_{u_2})}$. To approximate the standard error of covariance $\hat{\sigma}_{u_1,u_2}$ requires an additional log-transformed step as $\hat{\sigma}_{u_1,u_2} = \exp(ll1) \cdot L2$ is a product of two elements. Let $\log \hat{\sigma}_{u_1,u_2} = h(ll_1, L_2) = ll_1 + \log L_2$, the variance of $\log \hat{\sigma}_{u_1,u_2}$ can be calculated as:

$$\mathrm{Var}(\log \hat{\sigma}_{u_1,u_2}) = \left(\frac{\partial h(\cdot)}{\partial ll_1}, \frac{\partial h(\cdot)}{\partial L_2}\right) \begin{pmatrix} \mathrm{Var}(ll_1^2) & \mathrm{Cov}(L_2, ll_1) \\ \mathrm{Cov}(L_2, ll_1) & \mathrm{Var}(L_2^2) \end{pmatrix} \begin{pmatrix} \frac{\partial h(\cdot)}{\partial ll_1} \\ \frac{\partial h(\cdot)}{\partial L_2} \end{pmatrix} \tag{4.3.5}$$

By using Delta method again and transforming back, the standard error of $\hat{\sigma}_{u_1,u_2}$ can be obtained by $\mathrm{SE}(\hat{\sigma}_{u_1,u_2}) = \sqrt{\hat{\sigma}^2_{u_1,u_2} \times \mathrm{Var}(\log \hat{\sigma}_{u_1,u_2})}$.

4.4 Estimating the Random Effects

The estimates of random intercepts and slopes $\mathbf{u_i}$ are also of interest. The Empirical Bayes (EB) method, which is the mean of the posterior distribution of the random effects $\mathbf{u_i}$ given the observed data, are often used to assess the individual deviation from the population mean. The posterior distribution of the random effects $\mathbf{u_i}$ in the ordinal random coefficient model can be written as:

$$\varphi_i(\mathbf{u_i}|\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}}, \hat{\mathbf{G}}_\mathbf{i}, \mathbf{x_i}) = \frac{L_i(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}}, \hat{\mathbf{G}}_\mathbf{i}, \mathbf{x_i}|\mathbf{u_i})}{\int L_i(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}}, \hat{\mathbf{G}}_\mathbf{i}, \mathbf{x_i}|\mathbf{u_i}) d\mathbf{u_i}} \tag{4.4.1}$$

where $L_i(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}}, \hat{\mathbf{G}}_\mathbf{i}, \mathbf{x_i}|\mathbf{u_i})$ is the likelihood function with a form of equation (4.2.1) when fitting an ordinal random intercept model and equation (4.2.10) when fitting an ordinal random coefficient model. $\int L_i(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}}, \hat{\mathbf{G}}_\mathbf{i}, \mathbf{x_i}|\mathbf{u_i}) d\mathbf{u_i}$ is the marginal likelihood does not depend on the random effect $\mathbf{u_i}$. Once the estimates of model parameters for $\boldsymbol{\alpha}, \boldsymbol{\beta}$ and $\mathbf{G_i}$ have been obtained, the marginal likelihood $\int L_i(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}}, \hat{\mathbf{G}}_\mathbf{i}, \mathbf{x_i}|\mathbf{u_i}) d\mathbf{u_i}$ becomes a constant. Therefore, the posterior distribution of random effects $\mathbf{u_i}$ is proportional to the likelihood, that is $\varphi_i(\mathbf{u_i}|\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}}, \hat{\mathbf{G}}_\mathbf{i}, \mathbf{x_i}) \propto L_i(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}}, \hat{\mathbf{G}}_\mathbf{i}, \mathbf{x_i}|\mathbf{u_i})$. Then the Empirical Bayes estimate of $\mathbf{u_i}$ can be obtained by optimizing $L_i(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}}, \hat{\mathbf{G}}_\mathbf{i}, \mathbf{x_i}|\mathbf{u_i})$ with respect to $\mathbf{u_i}$.

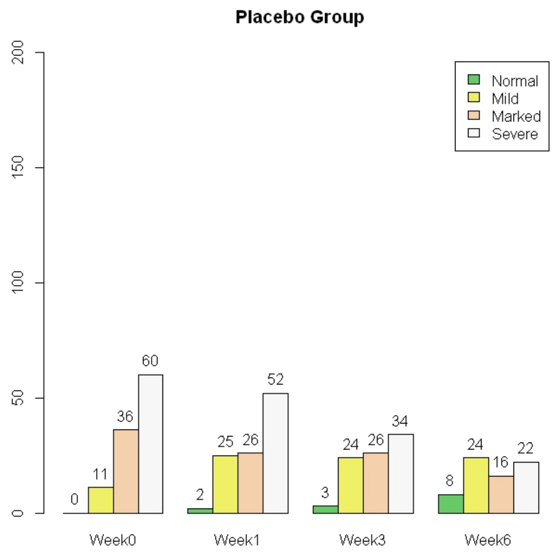
4.5 NIMH Schizophrenia Example Revisited

We revisit the NIMH Schizophrenia example described in Section 1.4. Here, we treat the data as longitudinal measurements where the correlations between repeated measurement from the same patient are not negligible. The Inpatient Multidimensional Psychiatric Scale (IMPS) score assessments for each patients were originally designed to be collected at four time points: Week 0, Week 1, Week 3 and Week 6. Due to noncompliance and loses to follow-up, a few samples were collected at the time points differing from those scheduled and several subjects missed scheduled visits as well. In the preprocessing stage, all missing data were removed from the analysis. Table 4.1 summarizes the IMPS score by treatment group and time from which we can see an unbalanced design was implemented with the number of patients recruited to the intervention group was roughly three times to that in the control group. In addition, the number of patients seen over the four time points decreased in both groups. A more detailed 'Severity of Illness' change with respect to time in the two groups can be observed from Figures 4.1 and 4.2 reproduced from [Hedeker and Gibbons, 2006]. From these barplots we can see, the number of severely ill patients in both groups drop dramatically as the study continuing while the number of normal people increases significantly, especially in the intervention group.

Table 4.1: Summary of number of patients seen at each of the four time points

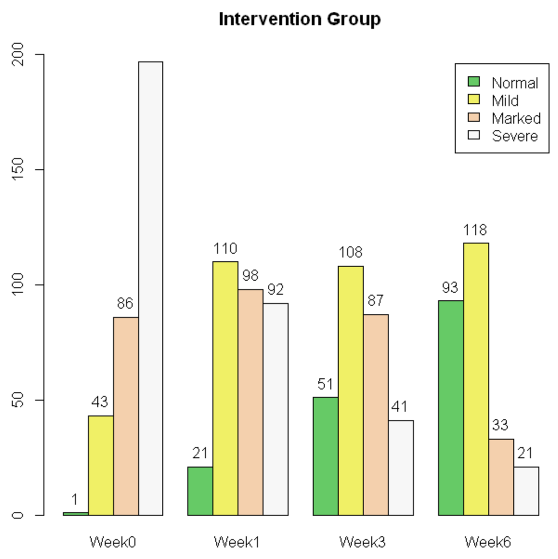
	Week0	Week1	Week3	Week6
Placebo	107	105	87	70
Intervention	327	321	287	265
Total	434	426	374	335

Figure 4.1: Summary of IMPS score (Normal, Mild, Marked, Severe) by Time in the Placebo Group



To further explore the treatment and time effects accounting for inter-subject variation as well as the intra-subject correlations, for each of the cumulative logit, adjacent-category, forward continuation ratio and backward continuation ratio ordinal model framework, we fit three types of mixed-effect models: the random intercept model, the random coefficient model with assuming no correlation between the random slope and random intercept, and the random coefficient model with an unstructured covariance matrix. We primarily present the results from cumulative logit random coefficient/intercept ordinal response models. Additional results can be found in Appendix D. Drug (0=placebo, 1=drug), time ($\sqrt{week}$) and the drug by time interaction terms were included as fixed effects in the model. The model performance can be evaluated using the likelihood-ratio based deviance test. To compare the parameter estimates obtained from different software and optimization methods, we fit the

Figure 4.2: Summary of IMPS score (Normal, Mild, Marked, Severe) by Time in the Intervention Group



ordinal mixed-effect models by three approaches: 1) Our own R program. The parameter estimates are obtained by optimizing the approximated marginal likelihood using 'L-BFGS-B' in the general-purpose optimization function `optim` for cumulative logit ordinal model and the 'BFGS' method in function `optimx` from R package `optimx` [Nash and Varadhan, 2012] for other types of the ordinal models. 2) SAS version 9.2. We used `PROC GLIMMIX` procedure for fitting random coefficient model with cumulative logit and `PROC NLMIXED` for fitting all other ordinal models. All parameter estimates are obtained using the Dual Quasi-Newton method. 3) R package `ordinal` developed by Christensen [2012]. The function `clmm` is capable for fitting the cumulative link mixed models with one or more random effects using the Penalized Iteratively Reweighted Least Squares (PIRLS) method implemented in R package `lme4`. However, since the `clmm` function is developed for fitting the random effects model rather than the random coefficient model, we are only able to compare the parameter estimates from the random intercept model with cumulative logit.

Tables 4.3, 4.4 and 4.5 present the model fitting results from our R code, SAS and R package `ordinal`. For each statistical software/package, the parame ter estimates remain consistent between the two numerical approximation methods. In this example, a consistent result and interpretation can be drawn from the statistical inference of the fixed effects when treating the data as longitudinal compared to treating it as independent observations. The drug and time interaction results in an extremely small p-value, which can be interpreted as the distribution of the IMPS score changes more dramatically in intervention group compared to that in the control group. As the interaction is significant, the interpretation of two main effects cannot be based on their own inferences solely. Instead, we interpret the drug and time main effects based on contrasts between intervention and placebo groups. The drug effect β_{drug} represents the difference of IMPS score distributions between intervention and placebo groups at baseline (week0). From Table 4.2 we can see the proportion of normal, mildly, markedly and severely ill subjects are similar in two groups which indicates no significant difference in the distribution of IMPS score at baseline. While the time effect β_{time} illustrates the change of IMPS score distribution as the study continues. Also from Table 4.2, the proportions of patients from markedly and severely ill groups drop dramatically in the intervention group compared with those in the placebo group, where a faster recovery rate is observed in the intervention group than the placebo group. In addition, the variance of the intercept σ^2_{int} measures subject heterogeneity at baseline which a large value indicates dynamic illness conditions.

Table 4.2: Contrast of proportions in severity of illness between intervention and placebo groups.

Time	IMPS score	Intervention	Placebo	Contrast
Week0	Normal	0.31%	0.29%	0.02%
	Mild	5.64%	5.34%	0.31%
	Marked	28.57%	27.56%	1.02%
	Severe	65.48%	66.82%	-1.34%
Week1	Normal	2.17%	0.62%	1.54%
	Mild	29.05%	10.78%	18.27%
	Marked	47.87%	40.34%	7.52%
	Severe	20.92%	48.25%	-27.33%
Week3	Normal	8.55%	1.09%	7.46%
	Mild	57.15%	17.3%	39.85%
	Marked	28.4%	46.86%	-18.45%
	Severe	5.9%	34.75%	-28.85%
Week6	Normal	27.89%	1.89%	26%
	Mild	60.91%	26.4%	34.5%
	Marked	9.71%	48.2%	-38.49%
	Severe	1.49%	23.51%	-22.01%

Tables 4.6 and 4.7 present the parameter estimates from the random coefficient model assuming the covariance between the random intercept and the random slope is 0. Tables 4.8 and 4.9 present parameter estimates from the random coefficient model with an unstructured variance matrix. Due to the different optimization methods, estimates of the variance component are slightly different from our R code and SAS. The interpretation of the fixed effects remain consistent to that of the random intercept model. As the random coefficient model takes the subject-specific heterogeneities in slope into consideration, it provides a more precise assessment of the time trend in both groups. When assuming the independence of the random intercept and random slope, from the fixed effects estimate in Table 4.6, we

can conclude the mean change of the IMPS score distribution along with time is 0.70 in the placebo group and $0.70 + 1.63 = 2.33$ in the intervention group. By using the variance of slope σ^2_{slope} from the random effects, we can caculate the 95% confidence interval for the two groups where the 95% CI=[-1.39, 2.79] for the placebo group and 95% CI=[0.24, 4.42] for the intervention group. Both confidence intervals have wide spread indicate large variations of the trends in two groups. Notice the 95% CI of the intervention group covers 0, which indicates both positive and negative trends have been observed. Similarly, in the random coefficient model with unstructured variance matrix, the mean and confidence interval for the time trend is 0.83, 95%CI=[-1.74, 3.40] for the placebo group and 2.5, 95% CI=[-0.07, 5.07] for the intervention group, which also addresses the subject variation in the time trend. From the unstructured variance matrix, the covariance between the random intercept and slope is also estimated where the correlation of them can be estimated as $r = -0.33$ from Table 4.8. This negative correlation can be interpreted as for most severely ill patients, a more dramatic improvement has been observed than the mildly and moderately ill patients.

Table 4.3: R output: Random Intercept Model with Cumulative Logit

Nonadaptive Gauss-Hermite ,$N_{GQ} = 7$					
Parameter	Estimate	SE	DF	t-value	P-value
α_1	-5.79	0.315	436	-18.39	<0.0001*
α_2	-2.78	0.273	436	-10.18	<0.0001*
α_3	-0.68	0.259	436	-2.61	0.009*
β_{drug}	0.08	0.300	436	0.26	0.80
β_{time}	0.77	0.130	436	5.87	<0.0001*
$\beta_{drug \times time}$	1.19	0.152	436	7.84	<0.0001*
σ^2_{int}	3.73	0.453			
$-logL$	1701.8				
Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 3$					
Parameter	Estimate	SE	DF	t-value	P-value
α_1	-5.84	0.331	436	-17.68	<0.0001*
α_2	-2.82	0.289	436	-9.75	<0.0001*
α_3	-0.70	0.274	436	-2.56	0.011*
β_{drug}	0.06	0.312	436	0.18	0.857
β_{time}	0.77	0.131	436	5.86	<0.0001*
$\beta_{drug \times time}$	1.20	0.153	436	7.88	<0.0001*
σ^2_{int}	3.72	0.456			
$-logL$	1702.1				

Table 4.4: SAS output: Random Intercept Model with Cumulative Logit

Parameter	Estimate	SE	DF	t-value	P-value
Nonadaptive Gauss-Hermite Quadrature, $N_{GQ} = 7$					
α_1	-5.79	0.315	436	-18.40	<0.0001*
α_2	-2.78	0.273	436	-10.19	<0.0001*
α_3	-0.67	0.259	436	-2.61	0.0094*
β_{drug}	0.08	0.300	436	0.26	0.80
β_{time}	0.77	0.131	436	5.87	<0.0001*
$\beta_{drug\times time}$	1.19	0.152	436	7.84	<0.0001*
σ^2_{int}	3.73	0.453			
$-logL$	1701.8				
Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 3$					
Parameter	Estimate	SE	DF	t-value	P-value
α_1	-5.84	0.331	436	-17.68	<0.0001*
α_2	-2.82	0.289	436	-9.75	<0.0001*
α_3	-0.70	0.274	436	-2.56	0.0107*
β_{drug}	0.06	0.313	436	0.18	0.86
β_{time}	0.77	0.131	436	5.86	<0.0001*
$\beta_{drug\times time}$	1.20	0.153	436	7.88	<0.0001*
σ^2_{int}	3.72	0.456			
$-logL$	1702.1				

Table 4.5: R package `ordinal`: Random Intercept Model with Cumulative Logit

Nonadaptive Gauss-Hermite Quadrature, $N_{GQ} = 7$

Parameter	Estimate	SE	DF	t-value	P-value
α_1	-5.85	0.334	436	-17.48	<0.0001*
α_2	-2.82	0.293	436	-9.61	<0.0001*
α_3	-0.70	0.278	436	-2.53	0.01*
β_{drug}	0.05	0.319	436	0.17	0.86
β_{time}	0.77	0.131	436	5.83	<0.0001*
$\beta_{drug \times time}$	1.21	0.153	436	7.86	<0.0001*
σ^2_{int}	3.75	1.937			
$-logL$	1701.6				

Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 3$

Parameter	Estimate	SE	DF	t-value	P-value
α_1	-5.84	0.326	436	-17.92	<0.0001*
α_2	-2.82	0.283	436	-9.94	<0.0001*
α_3	-0.70	0.268	436	-2.62	0.01*
β_{drug}	0.06	0.303	436	0.18	0.86
β_{time}	0.77	0.130	436	5.90	<0.0001*
$\beta_{drug \times time}$	1.20	0.151	436	7.95	<0.0001*
σ^2_{int}	3.72	1.928			
$-logL$	1702.1				

Table 4.6: R output: Random Coefficient Model with Cumulative Logit assuming COV$(\sigma_{int}, \sigma_{slope}) = 0$

Nonadaptive Gauss-Hermite Quadrature, $N_{GQ} = 11$					
Parameter	Estimate	SE	DF	t-value	P-value
α_1	-6.69	0.390	435	-17.13	<0.0001*
α_2	-2.94	0.307	435	-9.55	<0.0001*
α_3	-0.54	0.285	435	-1.88	0.06
β_{drug}	-0.14	0.325	435	-0.43	0.67
β_{time}	0.69	0.183	435	3.75	0.0002*
$\beta_{drug \times time}$	1.66	0.223	435	7.43	<0.0001*
σ^2_{int}	4.01	0.629			
σ^2_{slope}	1.23	0.257			
$-logL$	1547.3				
Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 5$					
Parameter	Estimate	SE	DF	t-value	P-value
α_1	-6.71	0.382	435	-17.58	<0.0001*
α_2	-2.98	0.301	435	-9.91	<0.0001*
α_3	-0.59	0.279	435	-2.11	0.04*
β_{drug}	-0.08	0.318	435	-0.26	0.80
β_{time}	0.70	0.179	435	3.88	<0.0001*
$\beta_{drug \times time}$	1.63	0.218	435	7.50	<0.0001*
σ^2_{int}	3.90	0.602			
σ^2_{slope}	1.14	0.240			
$-logL$	1737.0				

Table 4.7: SAS output: Random Coefficient Model with Cumulative Logit assuming COV($\sigma_{int}, \sigma_{slope}$) $= 0$

Nonadaptive Gauss-Hermite Quadrature, $N_{GQ} = 11$					
Parameter	Estimate	SE	DF	t-value	P-value
α_1	-6.80	0.399	435	-17.06	<0.0001*
α_2	-3.02	0.312	435	-9.70	<0.0001*
α_3	-0.61	0.287	435	-2.11	0.035*
β_{drug}	-0.07	0.326	435	-0.21	0.83
β_{time}	0.70	0.183	435	3.80	0.0002*
$\beta_{drug \times time}$	1.67	0.224	435	7.46	<0.0001*
σ^2_{int}	4.08	0.640			
σ^2_{slope}	1.25	0.260			
$-logL$	1669.4				
Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 5$					
Parameter	Estimate	SE	DF	t-value	P-value
α_1	-6.78	0.394	435	-17.19	<0.0001*
α_2	-3.01	0.310	435	-9.72	<0.0001*
α_3	-0.60	0.286	435	-2.08	0.038*
β_{drug}	-0.08	0.325	435	-0.26	0.79
β_{time}	0.70	0.184	435	3.80	0.0002*
$\beta_{drug \times time}$	1.66	0.223	435	7.44	<0.0001*
σ^2_{int}	4.08	0.641			
σ^2_{slope}	1.23	0.257			
$-logL$	1669.5				

Table 4.8: R output: Random Coefficient Model with Cumulative Logit

Nonadaptive Gauss-Hermite Quadrature, $N_{GQ} = 21$					
Parameter	Estimate	SE	DF	t-value	P-value
α_1	-7.32	0.473	435	-15.45	<0.0001*
α_2	-3.42	0.387	435	-8.85	<0.0001*
α_3	-0.82	0.351	435	-2.32	0.02*
β_{drug}	-0.05	0.391	435	-0.14	0.89
β_{time}	0.88	0.218	435	4.05	<0.0001*
$\beta_{drug \times time}$	1.70	0.253	435	6.70	<0.0001*
σ^2_{int}	6.99	1.322			
$\sigma_{int,slope}$	-1.51	0.536			
σ^2_{slope}	2.01	0.592			
$-logL$	1662.8				
Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 7$					
Parameter	Estimate	SE	DF	t-value	P-value
α_1	-7.13	0.446	435	-16.00	<0.0001*
α_2	-3.29	0.363	435	-9.08	<0.0001*
α_3	-0.75	0.332	435	-2.25	0.02*
β_{drug}	-0.06	0.371	435	-0.16	0.87
β_{time}	0.83	0.206	435	4.03	0.0001*
$\beta_{drug \times time}$	1.67	0.241	435	6.92	<0.0001*
σ^2_{int}	6.01	1.077			
$\sigma_{int,slope}$	-1.06	0.410			
σ^2_{slope}	1.72	0.462			
$-logL$	1664.8				

Table 4.9: SAS output: Random Coefficient Model with Cumulative Logit

Nonadaptive Gauss-Hermite Quadrature ,$N_{GQ} = 21$

Parameter	Estimate	SE	DF	t-value	P-value
α_1	-7.32	0.473	435	-15.49	<0.0001*
α_2	-3.42	0.386	435	-8.86	<0.0001*
α_3	-0.81	0.351	435	-2.32	0.02*
β_{drug}	-0.06	0.391	435	-0.14	0.89
β_{time}	0.88	0.218	435	4.05	<0.0001*
$\beta_{drug\times time}$	1.69	0.252	435	6.72	<0.0001*
σ^2_{int}	7.00	1.320			
$\sigma_{int,slope}$	-1.51	0.533			
σ^2_{slope}	2.01	0.419			
$-logL$	1662.8				

Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 7$

Parameter	Estimate	SE	DF	t-value	P-value
α_1	-7.32	0.472	435	-15.50	<0.0001*
α_2	-3.42	0.386	435	-8.86	<0.0001*
α_3	-0.81	0.351	435	-2.32	0.02*
β_{drug}	-0.06	0.391	435	-0.14	0.89
β_{time}	0.88	0.218	435	4.06	<0.0001*
$\beta_{drug\times time}$	1.69	0.252	435	6.72	<0.0001*
σ^2_{int}	6.99	1.317			
$\sigma_{int,slope}$	-1.51	0.531			
σ^2_{slope}	2.00	0.416			
$-logL$	1662.8				

4.6 Health Services Research Example

We now reproduce the results from the Health Services Research example described in [Hedeker and Gibbons, 2006]. The data was collected from the McKinney Homeless Research Project study conducted in San Diego, CA with the aim to evaluate the usefulness and effectiveness of section 8 certificate as a way to improve living condition for the severely mentally ill homeless. A total of 361 people participated in this study among which, 181 with section 8 certificate and 180 without section 8 certificate. The housing status of these participants were classified into three groups: 1) streets or shelters; 2) community housing; and 3) independent housing and the status were repeatedly recorded at four time points: baseline, 6 month, 12 month and 24 month. Figures 4.4 and 4.3 present the distribution of housing status in groups with and without section 8 certificates at the four time points being evaluated. From both figures we observe that as the study continued, the living conditions for both groups tremendously improved given a significant number of people moved from streets or shelters to independent housing. Also from the two figures, we conclude that a larger proportion of people from the group with section 8 certificates were able to afford better living condition. To demonstrate, for each of the cumulative logit, adjacent-category, forward continuation ratio and backward continuation ratio ordinal model framework, we fit a random intercept model to capture the individual variation. In each model, the fixed effects are time effects (treated as categorical and measured at: 6 month, 12 month and 24 month), certificate effect (treated as categorical indicating whether a person had a section 8 certificate) as well as time and certificate interactions. To compare the parameter estimates obtained from different software, we fit the random intercept model with ordinal response by three approaches: 1) Our own R program; 2) `PROC GLIMMIX` procedure in SAS version 9.2;

and 3) R package `ordinal`. Tables 4.10, 4.11 and 4.12 present the fitting results for the random intercept model with cumulative logit using the three computational tools, respectively. The estimates remain consistent except the sign difference due to different parameterizations. All three time effects are associated with small p-values indicating the distribution of the housing status is significantly different compared to that at baseline. In addition, the proportion of participants living on the the street or in a shelter consistently decreased as the study continued. The time and certificate interactions are significant at 6 and 12 months and marginally significant at 24 months suggesting that the group with section 8 certificate had better living conditions than the other group at 6 and 12 month, while there was no significant difference at 24 month. It is interesting to see the certificate effect is marginally significant, indicating the overall living conditions of two groups are not significantly different. We should be skeptical of this conclusion as the results may be contradictory due to violation of proportional odds assumption. Therefore, a more complicated model should be used to address the issue. It is also of interest to explore the intra-subject correlation. According to discussion in Section 4.1, the within-subject correlation can be estimated as $\frac{\sigma_{int}^2}{\sigma_{int}^2+\pi^2/3} = 0.39$.

Figure 4.3: Summary of Housing Status by Time in Group with Section 8 Certificates

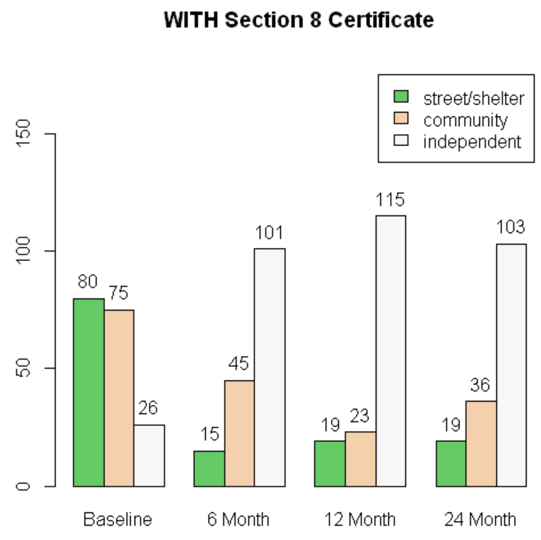


Figure 4.4: Summary of Housing Status by Time in Group without Section 8 Certificates

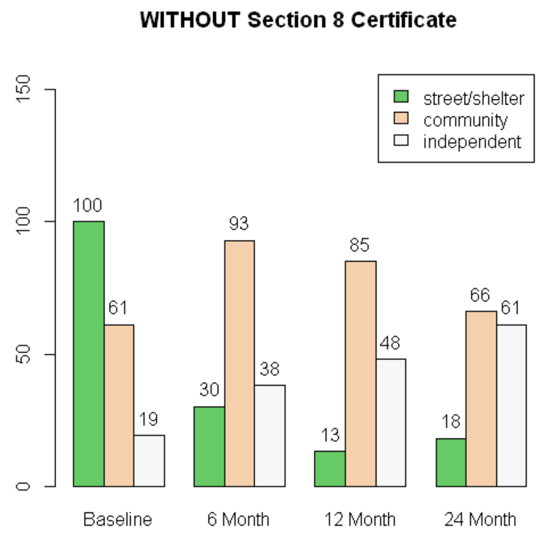


Table 4.10: San Diego Homeless Example R output: Random Intercept Model with Cumulative Logit

Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 10$					
Parameter	Estimate	SE	DF	t-value	P-value
α_1	0.22	0.198	360	1.11	0.27
α_2	2.96	0.230	360	12.92	<0.0001*
6 month	-1.74	0.235	360	-7.40	<0.0001*
12 month	-2.32	0.247	360	-9.39	<0.0001*
24 month	-2.50	0.253	360	-9.88	<0.0001*
Section 8 (Yes=1, No=0)	-0.50	0.276	360	-1.80	0.07
Section 8 at 6 month	-1.41	0.341	360	-4.13	<0.0001*
Section 8 at 12 month	-1.17	0.353	360	-3.32	<0.0001*
Section 8 at 24 month	-0.64	0.349	360	-1.83	0.07
σ^2_{int}	2.13	0.354			
-logL	1137.2				

Table 4.11: San Diego Homeless Example SAS output: Random Intercept Model with Cumulative Logit

Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 10$					
Parameter	Estimate	SE	DF	t-value	P-value
α_1	0.22	0.198	359	1.11	0.27
α_2	2.96	0.230	359	12.92	<0.0001*
6 month	-1.74	0.235	921	-7.40	<0.0001*
12 month	-2.32	0.247	921	-9.39	<0.0001*
24 month	-2.50	0.253	921	-9.88	<0.0001*
Section 8 (Yes=1, No=0)	-0.50	0.276	921	-1.80	0.07
Section 8 at 6 month	-1.41	0.341	921	-4.13	<0.0001*
Section 8 at 12 month	-1.17	0.353	921	-3.32	0.0009*
Section 8 at 24 month	-0.64	0.349	921	-1.83	0.07
σ^2_{int}	2.13	0.354			
-logL	1137.2				

Table 4.12: San Diego Homeless Example R package `ordinal` : Random Intercept Model with Cumulative Logit

Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 10$					
Parameter	Estimate	SE	DF	t-value	P-value
α_1	0.22	0.210	360	1.05	0.30
α_2	2.96	0.229	360	12.94	<0.0001*
6 month	1.74	0.242	360	7.17	<0.0001*
12 month	2.31	0.248	360	9.32	<0.0001*
24 month	2.50	0.258	360	9.70	<0.0001*
Section 8 (Yes=1, No=0)	0.50	0.300	360	1.65	0.10
Section 8 at 6 month	1.41	0.357	360	3.94	<0.0001*
Section 8 at 12 month	1.17	0.365	360	3.21	<0.0001*
Section 8 at 24 month	0.64	0.371	360	1.72	0.09
σ^2	2.13	0.354			
-logL	1137.2				

Chapter 5

Penalized Model for Traditional Longitudinal High-dimensional Data with an Ordinal Response

We have introduced the ordinal response and reviewed the ordinal model under the proportional odds assumption in Chapter 1. In Chapter 2, we have reviewed several prevalent regularization models for deriving a parsimonious model using high-dimensional data when the response is either continuous or dichotomous. The statistical techniques for analyzing the longitudinal data was reviewed in Chapter 3. Three classes of mixed effects models: Linear Mixed Effects Model, Nonlinear Mixed Effects Model, and the Generalized Linear Mixed Effects Model were discussed for analyzing a variety of repeated measured data with responses having distinct distributions. In Chapter 4, we provided a comprehensive discussion on the statistical model for the clustered and longitudinal data with an ordinal response. In this chapter, we concentrate on solving the problem of interest. The problem of interest is to **identify a subset of important features that is monotonically associated with an ordinal response and can be utilized to build a parsimonious model for prediction and classification in a high-dimensional or a longitudinal high-dimensional setting**. We propose our methodologies for building a parsimonious model to predict an ordinal outcome using features selected from high-dimensional or longitudinal high-dimensional data. The rest of Chapter 5 is organized as follows: in Section 5.1, we first review the incremental forward stagewise method introduced in Chapter 2 with more details. In Section 5.2, we extend the Forward Stagewise algorithm to be suitable for the ordinal response setting. In Section 5.3, we combine the random coefficient model with an ordinal response discussed in Chapter 4 and the forward stagewise algorithm to build a parsimonious model for prediction and classification using longitudinal high-dimensional data. Model assessment and selection criteria are provided in Section 5.4. A brief description of the software implementation and our R package with a list of primary R functions are included in Section 5.5. Two simulation examples with the data generation procedure and model fitting results are presented in Section 5.6. Additional discussion follows in Section 5.7.

5.1 Review of Forward Stagewise Method

The forward stagewise method, also known as incremental forward stagewise regression, is a slow learning procedure to provide a greedy approximation to the proposed function. The terminology 'learning' defined by Arthur Samuel indicates a process that gives computers the ability to learn for achieving a goal without being explicitly programmed. Since a typical forward stagewise method consists of hundreds of thousands of tiny little steps before converging, it was once considered as an inefficient algorithm and remained for a long time. It has gained enormous attention when Hastie et al. [2007] discovered the prevalent supervised learning method. Boosting [Schapire et al., 1998] is the combination of a sequence of adaptively constructed basis function and a forward stagewise procedure. Another striking discovery of forward stagewise by Efron et al. [2004] is that it approximates the solution to an L_1 constrained regression (LASSO) problem and the coefficient profiles look exactly the same under certain scenarios.

The forward stagewise method adopts the gradient descent optimization method, which is also known as steepest descent method. It is a first-order optimization algorithm where for each iteration, one takes a step proportional to the negative of the gradient in order to achieve the local minimum of the purposed function. There are two versions of gradient descent method, cyclic and greedy. We illustrate the two methods using a simple function $F(\cdot)$ with two variables x_1 and x_2. Equations (5.1.1) and (5.1.2) illustrate the two methods in mathematical formulae:

$$\begin{pmatrix} x_1^{(k+1)} \\ x_2^{(k+1)} \end{pmatrix} = \begin{pmatrix} x_1^{(k)} \\ x_2^{(k)} \end{pmatrix} - \gamma^{(k)} \left(\nabla F_{x_1=x_1^{(k)}} + \nabla F_{x_2=x_2^{(k)}} \right) \tag{5.1.1}$$

$$\begin{pmatrix} x_1^{(k+1)} \\ x_2^{(k+1)} \end{pmatrix} = \begin{pmatrix} x_1^{(k)} - \gamma^{(k)} \nabla F_{x_1 = x_1^{(k)}} \\ x_2^{(k)} \end{pmatrix} \tag{5.1.2}$$

where $\nabla F_{x_1} = \frac{\partial F(x_1,x_2)}{\partial x_1}, \nabla F_{x_2} = \frac{\partial F(x_1,x_2)}{\partial x_2}$ and $\gamma^{(k)}$ is a small number that determines the magnitude. At each iteration, a greedy method selects one coefficient which leads to the maximum decrease in the purposed function and moves along that direction. On the contrary, a cyclic method updates every coefficient and moves along the joint direction. For example, at the $(k+1)^{th}$ iteration, the local optimal points $(x_1^{(k+1)}, x_2^{(k+1)})$ are determined by its previous position $(x_1^{(k)}, x_2^{(k)})$ and the gradient evaluated at the k^{th} step. For the cyclic method, the direction and magnitude of the gradient are determined by both x_1 and x_2 and therefore the point is moved on both axes as shown in equation (5.1.1). For the greedy method however, the direction and magnitude are determined by the largest negative gradient and the point is moved in the associated direction. In equation (5.1.2), we assume $|\nabla F_{x_1}| > |\nabla F_{x_2}|$, therefore the point is moved along the x_1 axis while remaining fixed on the x_2 axis.

There are appealing advantages for using the greedy version of the gradient descent optimization method such as the method is computationally inexpensive and the results are readily interpretable. In addition, by intentionally setting $\gamma^{(k)}$ to be small enough, high consistency regarding the selected features can be achieved. One common criticism of the greedy algorithm is it may fail to produce the globally optimal solution since it neither searches exhaustively on all possible paths nor adjusts for the previous paths. Because of this limited vision problem, the greedy algorithm is often implemented to perform the feature selection rather than merely optimization.

5.2 Regularization Method for High-dimensional Data with Ordinal Response

We modified the Generalized Monotone Incremental Forward Stagewise algorithm for logistic regression [Hastie et al., 2007] to be suitable for the ordinal model, particularly the cumulative logit ordinal model under the proportional odds assumption. Recall in equation (5.2.1), we introduced the cumulative logit ordinal model having the form

$$\log\left(\frac{\gamma_{ic}}{1-\gamma_{ic}}\right) = \log\left(\frac{P(Y_i \leq c|\mathbf{x_i})}{P(Y_i > c|\mathbf{x_i})}\right) = \alpha_c + \mathbf{x_i^T}\boldsymbol{\beta}, c = 1, \cdots, C-1; i = 1, \cdots, n \quad (5.2.1)$$

where α_c denotes the category-specific intercept; $\boldsymbol{\beta}$ is a $p \times 1$ vector of coefficients associated with explanatory variables $\mathbf{x_i}$. In a high-dimensional setting, where the number of features (p) is much greater than the number of observations (N), the maximum likelihood estimates for the model parameters $\alpha_c, \boldsymbol{\beta}$ are no longer feasible due to insufficient degrees of freedom. Alternatively, the forward stagewise algorithm, by updating one coefficient β_j at a time using a small incremental amount, provides a greedy approximation to the penalized likelihood function. This incremental learning procedure tremendously increases the prediction accuracy by trading off the unbiasedness for smaller variation.

To implement the forward stagewise algorithm for the ordinal model, the first-order derivative of the negative log-likelihood function $-\log L(\boldsymbol{\alpha}, \boldsymbol{\beta}; \mathbf{x})$ with respect to each β_j is required to determine the coefficient update direction. To reduce the complexity of computation and accelerate the optimization process, through the selection procedure the intercept $\boldsymbol{\alpha}$ remains the same and is estimated under the null hypothesis where $\boldsymbol{\beta} = 0$. That is, for the intercept α_c corresponding to the c^{th} category in cumulative logit ordinal model, the estimate $\hat{\alpha}_c$ can be written as $\alpha_c = \log \frac{P(Y_i \leq c)}{1-P(Y_i \leq c)}$. The estimate of $\boldsymbol{\alpha}$ will be updated given the penalized estimate of $\boldsymbol{\beta}$ for a more accurate calculation of the model fitting criteria once the selection process completes. In equation (1.3.1), we presented the general form of the likelihood function for the ordinal model. Correspondingly, the negative log-likelihood function can be written as:

$$\begin{aligned} -\log L(\boldsymbol{\alpha}, \boldsymbol{\beta}; \mathbf{x}) &= -\sum_{i=1}^{n} \log L_i(\boldsymbol{\alpha}, \boldsymbol{\beta}; \mathbf{x_i}) \\ &= -\sum_{i=1}^{n} \log \left(\prod_{c=1}^{C} \pi_c(\mathbf{x_i})^{y_{ci}} \right) = -\sum_{i=1}^{n} \sum_{c=1}^{C} y_{ci} \log \pi_c(\mathbf{x_i}) \end{aligned} \tag{5.2.2}$$

For a given β_j, the first-order derivative of $-\log L(\boldsymbol{\alpha}, \boldsymbol{\beta}; \mathbf{x})$ can be written as:

$$-\frac{\partial \log L(\boldsymbol{\alpha}, \boldsymbol{\beta}; \mathbf{x})}{\partial \beta_j} = -\sum_{i=1}^{n} \left(\sum_{c=1}^{C} y_{ci} \cdot \frac{\frac{\partial \pi_c(\mathbf{x_i})}{\partial \beta_j}}{\pi_c(\mathbf{x_i})} \right) \tag{5.2.3}$$

For cumulative logit ordinal model specifically, the probability for each category $\pi_c(\mathbf{x_i})$ can be written as:

$$\begin{aligned} \pi_c(\mathbf{x_i}) &= P(Y_i \leq c|\mathbf{x_i}) - P(Y_i \leq c-1|\mathbf{x_i}) \\ &= \frac{\exp(\alpha_c + \mathbf{x_i}^T\boldsymbol{\beta})}{1+\exp(\alpha_c + \mathbf{x_i}^T\boldsymbol{\beta})} - \frac{\exp(\alpha_{c-1} + \mathbf{x_i}^T\boldsymbol{\beta})}{1+\exp(\alpha_{c-1} + \mathbf{x_i}^T\boldsymbol{\beta})} \end{aligned} \tag{5.2.4}$$

and then for any given category c, the partial derivative of $\pi_c(\mathbf{x_i})$ with respect to β_j can be calculated as:

$$\begin{aligned} \frac{\partial \pi_c(\mathbf{x_i})}{\partial \beta_j} &= x_{ij} \cdot \big(P(Y_i \leq c|\mathbf{x_i})(1-P(Y_i \leq c|\mathbf{x_i})) - P(Y_i \leq c-1|\mathbf{x_i})(1-P(Y_i \leq c-1|\mathbf{x_i}))\big) \\ &= x_{ij} \cdot \left(\frac{\exp(\alpha_c + \mathbf{x_i}^T\boldsymbol{\beta})}{(1+\exp(\alpha_c + \mathbf{x_i}^T\boldsymbol{\beta}))^2} - \frac{\exp(\alpha_{c-1} + \mathbf{x_i}^T\boldsymbol{\beta})}{(1+\exp(\alpha_{c-1} + \mathbf{x_i}^T\boldsymbol{\beta}))^2}\right). \end{aligned} \tag{5.2.5}$$

Correspondingly, the element in equation (5.2.3) can be written as:

$$\begin{aligned} \frac{\frac{\partial \pi_c(\mathbf{x_i})}{\partial \beta_j}}{\pi_c(\mathbf{x_i})} &= x_{ij} \cdot (1 - P(Y_i \leq c-1|\mathbf{x_i}) - P(Y_i \leq \mathbf{x_i})) \\ &= x_{ij} \cdot \left(1 - \frac{\exp(\alpha_{c-1} + \mathbf{x_i}^T\boldsymbol{\beta})}{1+\exp(\alpha_{c-1} + \mathbf{x_i}^T\boldsymbol{\beta})} - \frac{\exp(\alpha_c + \mathbf{x_i}^T\boldsymbol{\beta})}{1+\exp(\alpha_{c-1} + \mathbf{x_i}^T\boldsymbol{\beta})}\right). \end{aligned} \tag{5.2.6}$$

Equations (5.2.5) and (5.2.6) can be further simplified for two extreme categories. When $c = 1$ follows $\alpha_0 = -\infty$ and $\frac{\exp(\alpha_0 + \mathbf{x_i}^T\boldsymbol{\beta})}{(1+\exp(\alpha_0 + \mathbf{x_i}^T\boldsymbol{\beta}))^2} = 0$, therefore

$$\frac{\partial \pi_1(\mathbf{x_i})}{\partial \beta_j} = x_{ij} \cdot \frac{\exp(\alpha_1 + \mathbf{x_i}^T\boldsymbol{\beta})}{(1+\exp(\alpha_1 + \mathbf{x_i}^T\boldsymbol{\beta}))^2} \tag{5.2.7}$$

and

$$\frac{\frac{\partial \pi_1(\mathbf{x_i})}{\partial \beta_j}}{\pi_1(\mathbf{x_i})} = x_{ij} \cdot \left(1 - \frac{\exp(\alpha_1 + \mathbf{x_i}^T\boldsymbol{\beta})}{1+\exp(\alpha_1 + \mathbf{x_i}^T\boldsymbol{\beta})}\right). \tag{5.2.8}$$

When c=C follows $\alpha_C = \infty$ and $\frac{\exp(\alpha_C + \mathbf{x_i}^T \boldsymbol{\beta})}{(1+\exp(\alpha_C+\mathbf{x_i}^T\boldsymbol{\beta}))^2} = 1$, therefore

$$\frac{\partial \pi_C(\mathbf{x_i})}{\partial \beta_j} = -x_{ij} \cdot \frac{\exp(\alpha_{C-1} + \mathbf{x_i}^T \boldsymbol{\beta})}{(1 + \exp(\alpha_{C-1} + \mathbf{x_i}^T\boldsymbol{\beta}))^2} \tag{5.2.9}$$

and

$$\frac{\frac{\partial \pi_C(\mathbf{x_i})}{\partial \beta_j}}{\pi_C(\mathbf{x_i})} = -x_{ij} \cdot \frac{\exp(\alpha_{C-1} + \mathbf{x_i}^T \boldsymbol{\beta})}{1 + \exp(\alpha_{C-1} + \mathbf{x_i}^T\boldsymbol{\beta})} \tag{5.2.10}$$

By plugging the exact form of $\frac{\partial \pi_c(\mathbf{x_i})}{\partial \beta_j}$ into equation (5.2.3) and solve, the gradient of negative log-likelihood for the ordinal model can be written as:

$$\begin{aligned}
-\frac{\partial \log L(\boldsymbol{\alpha}, \boldsymbol{\beta}; \mathbf{x})}{\partial \beta_j} &= \sum_{i=1}^{n} \Bigg(\frac{y_{1i} \cdot \frac{\partial \pi_1(\mathbf{x_i})}{\partial \beta_j}}{\pi_1(\mathbf{x_i})} + \sum_{c=2}^{C-1} \frac{y_{ci} \cdot \frac{\partial \pi_c(\mathbf{x_i})}{\partial \beta_j}}{\pi_c(\mathbf{x_i})} + \frac{y_{Ci} \cdot \frac{\partial \pi_C(\mathbf{x_i})}{\partial \beta_j}}{\pi_C(\mathbf{x_i})} \Bigg) \\
&= \sum_{i=1}^{n} x_{ij} \cdot \Bigg(y_{1i} \frac{1}{1 + \exp(\alpha_1 + \mathbf{x_i}^T\boldsymbol{\beta})} + \sum_{c=2}^{C-1} y_{ci} \Bigg(1 - \frac{\exp(\alpha_{c-1} + \mathbf{x_i}^T\boldsymbol{\beta})}{1 + \exp(\alpha_{c-1} + \mathbf{x_i}^T\boldsymbol{\beta})} \\
&- \frac{\exp(\alpha_c + \mathbf{x_i}^T\boldsymbol{\beta})}{1 + \exp(\alpha_{c-1} + \mathbf{x_i}^T\boldsymbol{\beta}} \Bigg) - y_{Ci} \frac{\exp(\alpha_{C-1} + \mathbf{x_i}^T\boldsymbol{\beta})}{1 + \exp(\alpha_{C-1} + \mathbf{x_i}^T\boldsymbol{\beta})} \Bigg)
\end{aligned} \tag{5.2.11}$$

where $-\frac{\partial \log L(\boldsymbol{\alpha},\boldsymbol{\beta};\mathbf{x})}{\partial \boldsymbol{\beta}} = \big(-\frac{\partial \log L(\boldsymbol{\alpha},\boldsymbol{\beta};\mathbf{x})}{\partial \beta_1}, \cdots, -\frac{\partial \log L(\boldsymbol{\alpha},\boldsymbol{\beta};\mathbf{x})}{\partial \beta_p}\big)$ is a gradient vector of length p. The β_j associated with the largest negative gradient of the log-likelihood is selected for updating by a small incremental amount.

At each iteration, the intercept α_c is also updated by fitting the cumulative logit ordinal model using the penalized estimates of $\boldsymbol{\beta}$ where the likelihood function $L(\boldsymbol{\alpha}, \boldsymbol{\beta}; \mathbf{x})$ in equation (1.3.2) is only dependent on the unknown parameter $\boldsymbol{\alpha}$. That is, equation (1.3.2) can be modified as:

$$
\begin{aligned}
L(\boldsymbol{\alpha}, \hat{\boldsymbol{\beta}}; \mathbf{x}) &= \prod_{i=1}^{n} \big(\prod_{c=1}^{C} \pi_c(\mathbf{x_i})^{y_{ic}} \big) \\
&= \prod_{i=1}^{n} \Bigg[\prod_{c=1}^{C} \Bigg(\frac{\exp(\alpha_c + \mathbf{x_i^T} \hat{\boldsymbol{\beta}})}{1 + \exp(\alpha_c + \mathbf{x_i^T} \hat{\boldsymbol{\beta}})} - \frac{\exp(\alpha_{c-1} + \mathbf{x_i^T} \hat{\boldsymbol{\beta}})}{1 + \exp(\alpha_{c-1} + \mathbf{x_i^T} \hat{\boldsymbol{\beta}})} \Bigg)^{y_{ic}} \Bigg]. \quad (5.2.12)
\end{aligned}
$$

The estimate of $\boldsymbol{\alpha}$ is obtained by optimizing the likelihood using the constrained nonlinear optimization algorithm Augmented Lagrangian Adaptive Barrier Minimization proposed by Varadhan [2011].

We now present the Generalized Monotone Incremental Forward Stagewise for ordinal response with high-dimensional data in Algorithm 1.

Algorithm 1

1. Create a negative version $-x_j$ of each predictor x_j and expand the predictor space to $\widetilde{\mathbf{X}} = (\mathbf{X}, -\mathbf{X})$. Set the initial values for the coefficients $\boldsymbol{\beta} = (\beta_1, \cdots, \beta_{2p}) = \mathbf{0}$. Obtain the estimate of intercepts under the null hypothesis where $\alpha_c = \log \frac{\sum_{c=1}^{c} P(Y \leq c)}{1-\sum_{c=1}^{c} P(Y \leq c)}$ for $c = 1 \cdots, C-1$ and $\alpha_0 = -\infty, \alpha_C = \infty$.

2. Find the predictor $x_j, j = 1, \cdots, 2p$ with the largest negative gradient of the log-likelihood $-\frac{\partial \log L(\boldsymbol{\alpha}, \boldsymbol{\beta}; \mathbf{x})}{\partial \beta_j}$ evaluated at the current estimate $\boldsymbol{\beta}^{(s)}$.

3. Update the coefficient estimate of the selected predictor x_j in step 2 with $\beta_j^{(s+1)} \leftarrow \beta_j^{(s)} + \epsilon$, where ϵ is a small positive amount; a rational choice is $\epsilon = 1 \times 10^{-4}$.

4. Fit the ordinal model with the updated predictor estimates $\boldsymbol{\beta}^{(s+1)}$ to obtain the updated intercept estimates $\boldsymbol{\alpha}^{(s+1)}$. The corresponding model fitting criteria AIC and BIC are also calculated.

5. Repeat steps $2, 3$ and 4 many times until convergence.

Since there is no standard stopping criteria in forward stagewise method for the ordinal model, we implemented the criteria to stop the iterative process when the difference between successive log-likelihoods is smaller than a given value, that is $\log L(\alpha, \boldsymbol{\beta}|\mathbf{x_i})_{\boldsymbol{\beta}=\boldsymbol{\beta}^{(s+1)}} - \log L(\alpha, \boldsymbol{\beta}|\mathbf{x_i})_{\boldsymbol{\beta}=\boldsymbol{\beta}^{(s)}} < \delta$. The forward stagewise method stops if this difference is smaller than a certain value δ. Note in step 4, it can be extremely cumbersome to fit an ordinal model at every single iteration for obtaining the intercept estimates $\boldsymbol{\alpha}$ since it is not uncommon for the forward stagewise algorithm to go through hundreds of thousands of iterations before converging. An efficient modification of step 4 can be to fit an ordinal model using the last set of features with nonzero penalized coefficients before a new feature enters into the model, which largely reduces the computational complexity. The final model can be selected based on either Akaike Information Criterion (AIC) [Akaike, 1974] with the form of

$$AIC = -2\log L(\boldsymbol{\alpha}, \boldsymbol{\beta}; \mathbf{x}) + 2k \tag{5.2.13}$$

or Bayesian Information Criterion (BIC)[Schwarz, 1978] with the form of

$$BIC = -2\log L(\boldsymbol{\alpha}, \boldsymbol{\beta}; \mathbf{x}) + k\log(n) \tag{5.2.14}$$

where in both criterion, k is the number of free parameters to be estimated and n is the number of observations. The ordinal model having a minimum AIC or BIC can be used for model selection such that the corresponding biased estimates $\hat{\boldsymbol{\alpha}}$ and $\hat{\boldsymbol{\beta}}_{biased}$ are utilized for prediction purposes.

5.3 Regularization Method for Longitudinal High-dimensional Data with an Ordinal Response

Longitudinal data analysis has been playing a profound and irreplaceable role in analyzing clustered and correlated data from a variety of fields. Substantial effort has been devoted to developing statistical models and inferential procedures for explaining the relationship between independent and dependent variables. However, little work has been done in developing the variable selection procedure for longitudinal data, especially for longitudinal data with high-dimensional features ($N << p$) due to the immense cost of generating and collecting longitudinal genomic data. Since the 1990s, the emergence of genomic technologies and the 'omics' revolution, the cost of genomic technologies has tremendously decreased and with their increased use, the need for developing cutting-edge data mining algorithms to identify and select a few key drivers from tens of thousands measured for a complex disease at the molecular level is imperative. However, due to the untidy confounding and multivariate dependencies in the longitudinal high-dimensional data, it is very challenging to select effective classifiers to build a parsimonious model and enhance predictability. Some work has been done by several researchers to address the variable selection problem in a linear mixed-effect model and generalized linear mixed model framework. For the linear mixed-effect model, Chen and Dunson [2003] proposed a hierarchical Bayesian model to identify any random effect having zero variance and thus performed random effects selection in linear mixed models. Vaida and Blanchard [2005] derived and compared different forms of Akaike information criterion (AIC) used for model selection in marginal and conditional representation of linear mixed-effect models where the population and cluster-specific parameters are of concern, respectively. Bondell et al. [2010] implemented the adaptive LASSO [Zou, 2006] as a shrinkage penalty on the reparameterized linear mixed-effects models for selecting fixed and random effects simultaneously. The model is fitted using the constrained EM algorithm [Larid et al., 1987] where the random effects are unobserved in the conditional expectation and the penalized likelihood is maximized at each iteration. For the generalized linear mixed model, Pan [2001] proposed a modified AIC based on quasi-likelihood with adjustment for the penalty which is suitable for model selection in Generalized Estimating Equations (GEE). Fu [2003] incorporated the bridge penalty to GEE model for variable selection when collinearity is present and the tuning parameter in the penalty is determined by quasi-Generalized Cross-Validation.

Among these algorithms mentioned above, none has provided a flexible structure for extending to analyze longitudinal high-dimensional data. Here, we further implemented the Generalized Monotone Incremental Forward Stagewise (GMIFS) algorithm discussed in Section 5.2 to analyze the longitudinal high-dimensional data. We propose a two-step algorithm: first, we concentrate on all the fixed-effects and implement the forward stagewise algorithm to perform the variable selection procedure for the high-dimensional features; second, we used the set of biased estimates to fit a random coefficient ordinal response model for classification and prediction purposes. It is worth mentioning that the variable selection procedure conducted by the forward stagewise method actually shares the same computational complexity for traditional and longitudinal data since in the ordinal random coefficient model, the normal distribution of the random component does not depend on the coefficient $\boldsymbol{\beta}$. Recall the likelihood $L(\boldsymbol{\alpha}, \boldsymbol{\beta}, \mathbf{u_i}, \mathbf{G_i}; \mathbf{x_i})$ for the ordinal random coefficient model discussed in Section 4.2 has an explicit form:

$$L(\boldsymbol{\alpha}, \boldsymbol{\beta}, u_i, \mathbf{G_i}; \mathbf{x}) = \prod_i \frac{1}{\sqrt{2\pi\sigma_{int}^2}} \exp\left(-\frac{u_i^2}{2\sigma_{int}^2}\right) \prod_{j=1}^{n_i} \prod_{c=1}^{C} \pi_c(\mathbf{x_{ij}}, u_i)^{y_{ijc}} \tag{5.3.1}$$

if assuming a random intercept model and

$$L_i(\boldsymbol{\alpha}, \boldsymbol{\beta}, \mathbf{G_i}, u_{1i}, u_{2i}; \mathbf{x_i}) = \frac{1}{2\pi\sigma_{u1}\sigma_{u2}\sqrt{1-\rho^2}} \exp\left(-\frac{1}{2(1-\rho^2)}\left(\frac{u_{1i}^2}{\sigma_{u1}^2} + \frac{u_{2i}^2}{\sigma_{u2}^2} - \frac{2\rho u_{1i}u_{2i}}{\sigma_{u1}\sigma_{u2}}\right)\right) \times \prod_{j=1}^{n_j} \prod_{c=1}^{C} \pi_c(u_{1i}, u_{2i})^{y_{ijc}}. \tag{5.3.2}$$

if assuming a random coefficient model. In both scenarios, the log-likelihood $\log L(\boldsymbol{\alpha}, \boldsymbol{\beta}, \mathbf{u_i}, \mathbf{G_i}; \mathbf{x_i})$ can be expressed as:

$$\log L(\boldsymbol{\alpha}, \boldsymbol{\beta}, \mathbf{u_i}, \mathbf{G_i}; \mathbf{x_i}) = \log g(\mathbf{u}, \mathbf{G}) + \log \left(\prod_{j=1}^{n_i} \prod_{c=1}^{C} \pi_c(\mathbf{x_{ij}}, u_i)^{y_{ijc}} \right) \tag{5.3.3}$$

where $g(\mathbf{u}, \mathbf{G})$ is the distribution of random effects $\mathbf{u}$. Since only the second part on the right side of equation (5.3.3) dependent on the coefficient $\boldsymbol{\beta}$, the gradient of the log-likelihood for the random coefficient ordinal model is the same as that of the traditional ordinal model, that is, $-\frac{\partial \log L(\boldsymbol{\alpha}, \boldsymbol{\beta}; \mathbf{x})}{\partial \beta_j} = -\frac{\partial \log L(\boldsymbol{\alpha}, \boldsymbol{\beta}, \mathbf{u_i}, \mathbf{G_i}; \mathbf{x})}{\partial \beta_j}$.

We now present our Generalized Monotone Incremental Forward Stagewise method for modeling a longitudinal ordinal response in the presence of high-dimensional data in Algorithm 2.

Algorithm 2

1. Create a negative version $-x_j$ of each predictor x_j and expand the predictor space to $\widetilde{\mathbf{X}} = (\mathbf{X}, -\mathbf{X})$. Set the initial values for the coefficients $\boldsymbol{\beta} = (\beta_1, \cdots, \beta_{2p}) = \mathbf{0}$. Obtain the estimate of intercepts under the null hypothesis where $\alpha_c = \log \frac{\sum_{c=1}^{c} P(Y \leq c)}{1 - \sum_{c=1}^{c} P(Y \leq c)}$ for $c = 1 \cdots, C-1$ and $\alpha_0 = -\infty, \alpha_C = \infty$.

2. Find the predictor $x_j, j = 1, \cdots, 2p$ with the largest negative gradient of the log-likelihood $-\frac{\partial \log L(\boldsymbol{\alpha}, \boldsymbol{\beta}, \mathbf{u_i}, \mathbf{G_i}; \mathbf{x})}{\partial \beta_j}$ evaluated at the current estimate $\boldsymbol{\beta}^{(s)}$.

3. Update the coefficient estimate of the selected predictor x_j in step 2 with $\beta_j^{(s+1)} \leftarrow \beta_j^{(s)} + \epsilon$, where ϵ is a small positive amount; a rational choice is $\epsilon = 1 \times 10^{-4}$.

4. Repeat steps 2 and 3 many times until convergence.

5. Update the intercept estimates $\hat{\boldsymbol{\alpha}}$ by fitting a series of ordinal model using the last set of features with nonzero coefficients before a new feature enters into the model. Calculate the corresponding model fitting criteria AIC and BIC for selecting the optimal set of features with nonzero coefficients.

6. A parsimonious ordinal random intercept/coefficient model

$$\log \left(\frac{P(Y_i \leq c | \mathbf{x_i}, \mathbf{u_i})}{P(Y_i > c | \mathbf{x_i}, \mathbf{u_i})} \right) = \alpha_c + \mathbf{x_i^T} \hat{\boldsymbol{\beta}}_{biased} + \mathbf{z_i u_i} \tag{5.3.4}$$

is fitted using the optimal penalized estimate $\hat{\boldsymbol{\beta}}_{biased}$ to further update the intercept estimate $\hat{\boldsymbol{\alpha}}$ and obtain the empirical Bayes estimator of the random effect $\mathbf{u_i}$, which are used for prediction and classification purposes along with $\hat{\boldsymbol{\beta}}_{biased}$.

It may be easy to notice the first five steps in the forward stagewise algorithm for ordinal response with longitudinal high-dimensional data stand exactly the same as that for traditional high-dimensional data where the beauty lies in the fact that the first-order derivative of the log-likelihood with respect to β_j for the two types of data have the exact same forms. The only modification made is the convergence criteria in step 4 to adapt to the dynamic and vulnerable properties of the longitudinal high-dimensional data and guarantees convergence. We implement the double convergence criteria: 1) the difference between two successive log-likelihood is smaller than a given value; 2) the number of features having a nonzero coefficient estimates is less than a specified value.

In addition, there is a trade-off between computational complexity and accuracy at step 5 when selecting the optimal model based on the AIC or BIC criteria. Currently, in the method described, we treat observation as independent and fit a fixed-effects only ordinal model using the last set of features with nonzero coefficients before a new feature enters into the model to obtain an approximately best model fitting criteria. Given the framework of longitudinal data, step 5 should have been done by fitting a series of penalized ordinal random intercept/coefficient model with the form of (5.3.4) using different sets of penalized estimates. However, the computational process can be extremely burdensome without implementation of parallel computing. Currently, we compromise a small amount of accuracy in return for a faster solution. We adjust this bias introduced by ignoring the within-subject correlation in step 6 by fitting a smaller number of penalized ordinal random intercept/coefficient model to further update intercept estimate $\hat{\boldsymbol{\alpha}}$ as well as model fitting criteria. Then the best parsimonious model can be selected accordingly where the performance of classification and prediction can be evaluated correspondingly. We now illustrate step 6 in more details. Suppose the model associated with optimal model fitting criteria selected in step 5 have q features with nonzero coefficients, in step 6 we fit $(2k + 1)$ penalized ordinal random intercept/coefficient models with the number of features with nonzero coefficients ranging from $q - k$ to $q + k$, where k specifies the range. For example, when $k = 0$, only one penalized ordinal random intercept/coefficient model with q nonzero coefficients is fitted; when $k = 1$, three penalized ordinal random intercept/coefficient models with $q - 1$, q and $q + 1$ nonzero coefficients are fitted, respectively.

5.4 Model Assessment and Selection

The performance of learning methods and predictive modeling are often evaluated by the consistency of variable selection and prediction accuracy. It is of ultimate importance to assess model performance using the proper criteria such that the chosen model can be generalized to a broader range of data. There are several prevalent methods for model assessment such as cross-validation and bootstrap methods. In this thesis, we use cross-validation to evaluate models for high-dimensional and longitudinal high-dimensional data. Cross-validation is probably the most widely used model assessment method for its easy interpretation and implementation. Prediction error is a good measurement of prediction accuracy and model complexity, which is defined as the average loss between observed Y and fitted $\hat{Y}$ over a given sample. A good learning method and predictive model should maintain similar prediction accuracy over any independent sample (Test sample) aside from the sample (Training sample) used to develop the final model. The prediction error estimated from the training sample is often not a good estimate of the prediction error compared to a test sample since the training error monotonically decreases as the model complexity increases, while a model with too many parameters is generally susceptible to new data and suffers from 'over-fitting'. Thus, a possible way to solve the problem is to use the holdout method by splitting the given dataset into two groups where the training set is used to fit an optimal model and the test set is used to evaluate whether the optimal model can be generalized to any independent dataset. One potential issue of the holdout method is the estimate of test error can be heavily dependent on how the data are split. An 'unfortunately' biased split can overwhelmingly mislead the estimate. Cross-validation corrects this problem by repeating the 'split-fit-evaluate' procedure many times and to obtain an average test error which is robust to the split. There are different ways to perform cross-validation, the most commonly used is K-fold cross-validation. To perform a K-fold cross-validation procedure, the data are divided into K parts where the $K-1$ parts form a training set and the remaining 1 part forms a test set. The model is fit using the training set and the average prediction error across all N observation is evaluated using equation (5.4.1).

$$CV(\hat{y}) = \frac{1}{N}\sum_{i=1}^{N} L(y_i, \hat{y}_i^{-K(i)}) \tag{5.4.1}$$

where $L(\cdot)$ is a loss function and $\hat{y}_i^{-K(i)}$ is the fitted observation when observation i is a member of the partition (test set). A typical loss function for categorical data is the 0-1 loss function where $L(y_i, \hat{y}_i^{-K(i)}) = I(y_i \neq \hat{y}_i^{-K(i)})$. This procedure is repeated K times until every partition is used as test set for exactly one time. When $K = N$, this procedure is also called N-fold or Leave-One-Out cross-validation. It is of interest to ask what the value of K should be selected to best perform the cross-validation procedure. Unfortunately, there is no single answer to this question and the choice of K is often determined by three considerations: tolerance of the bias, sparsity of the data, and computational time. Hastie et al. [2009] explored in detail the relationship between size of the training set and the prediction error. Typically, when the number of folds K increases, the bias of the test error decreases while the variance increases as well as the computational time required to achieve certain degree of accuracy. In addition, for a very sparse dataset, an N-fold cross-validation procedure is often implemented to obtain a good estimate of prediction error. There are few publications that discuss cross-validation procedures for longitudinal high-dimensional data. In this thesis, we conduct the quasi K-fold cross-validation for longitudinal data by dividing the total subjects into K partitions where the $K - 1$ partitions from a training set and the remaining 1 part forms a test set. Since the number of repeated measurements per subject can vary, the number of observations in the test set varies with each split.

There is an unavoidable issue associated with cross-validation procedures for ordinal response data when the ordinal categories are highly unbalanced. When only a few observations belong to one category, it may happen that a training dataset does not cover the full original ordinal scale. This can lead to a very high prediction error in the test dataset and yields misleading results. This problem is especially severe for longitudinal data when the extreme measurements are associated with a few subjects. To avoid this problem in K-fold cross-validation, we excluded the training sets that do not contain the full range of ordinal outcomes for both traditional and longitudinal data.

Besides the expected prediction error estimated from the cross-validation procedure, the other tool we use to evaluate the model performance is the Goodman and Kruskal's gamma. This gamma statistic, defined by Goodman and Kruskal [1954], measures the similarity of the orderings from the observed and predicted response and thus assesses the strength of association in a contingency table. It is calculated as a function of concordant and discordant pairs. According to [Agresti, 2010], a concordant pair is when the subject ranked higher on one measurement also ranks higher on the other and a discordant pair is when the subject ranked higher on one measurement ranks lower on the other. Mathematically, the probabilities of concordance (C) and discordance (D) are defined in equation (5.4.2).

$$\begin{aligned} C &= \sum_i \sum_j \pi_{ij} \big(\sum_{h>i} \sum_{k>j} \pi_{hk} \big) \\ D &= \sum_i \sum_j \pi_{ij} \big(\sum_{h<i} \sum_{k<j} \pi_{hk} \big) \end{aligned} \tag{5.4.2}$$

Correspondingly, the gamma statistics can be constructed as:

$$\gamma = \frac{C - D}{C + D} \tag{5.4.3}$$

where γ ranges from -1 to 1. When there are no discordant pairs $D = 0$ implies $\gamma = 1$ and when there are no concordant pairs $C = 0$ implies $\gamma = -1$.

5.5 Software Implementation

We developed the computational tool using the R programming environment and produced an R package called `ordinalmixed`. We attach the reference manual here which includes the detailed descriptions of the main functions in this package. In addition, two flowcharts (5.1) and (5.2) describe the working flow for the core functions.

Package ‘ordinalmixed’

November 9, 2013

Type Package

Title Fit traditional and penalized random coefficients/intercept
ordinal response model for classifying and predicting an longitudinal ordinal response.

Version 1.0

Date 2013-11-07

Author Jiayi Hou, Kellie Archer

Maintainer Jiayi Hou <houj2@vcu.edu>

Description Develop a set of classifiers using GMIFS algorithm to identify features monotonically increasing or decreasing with the ordinal response. Fit a random coefficient/intercept ordinal response model using the exact method. Incorporate the penalized estimates from GMIFS with other predictors to construct a penalized random coefficient/intercept ordinal response model for classification and prediction.

License GPL-2

Depends numDeriv, ucminf, glmmML, MASS, optimx, alabama

R topics documented:

`ordinalmixed-package` *Fit traditional and penalized random coefficients/intercept ordinal response model for classifying and predicting an longitudinal ordinal response.*

Description

Develop a set of classifiers using Generalized Monotone Incremental Forward Stagewise (GMIFS) algorithm to identify features monotonically increasing or decreasing with the ordinal response.
Fit a random coefficient/intercept ordinal response model where the parameter estimates are obtained from optimizing the marginal likelihood.
Incorporate the penalized estimates from GMIFS with other predictors to construct a penalized random coefficient/intercept ordinal response model for classification and prediction.
Implement the cross-validation procedure to evaluate the model performance.
Develop visualization tools to produce regularization path plots as well as other graphs.

Details

Package:	ordinalmixed
Type:	Package
Version:	1.0
Date:	2013-11-07
License:	GPL-2

Author(s)

Jiayi Hou, Kellie Archer

Maintainer: Jiayi Hou <houj2@vcu.edu>

References

KJ. Archer and J. Hou (2013). Generalized Monotone Incremental Forward Stagewise Method for Ordinal Response Prediction for High-dimensional datasets. Technical Report.
J. Hou (2013). Regularization Methods for Predicting an Ordinal Response Using Longitudinal High-dimensional Genomic Data. Dissertation work.

`forward.stagewise.cum` *Variable Selection by GMIFS in Cumulative Logit Ordinal Model*

Description

This function performs the Generalized Monotone Incremental Forward Stagewise (GMIFS) procedure for traditional, high-dimensional and longitudinal high-dimensional data with an ordinal response and yields a full solution of the penalized coefficient estimates $\boldsymbol{\beta}$.

Usage

```
forward.stagewise.cum(x, y, epsilon, tol, cut.prop)
```

Arguments

`x`	a matrix contains all the predictors to be identified by GMIFS.
`y`	an ordered factor represents an ordinal response
`epsilon`	an optional numeric represents a small incremental amount to update $\boldsymbol{\beta}$ at each iteration. Default is `epsilon=1e-4`.
`tol`	an optional numeric represents a convergence criterion between successive log-likelihoods. Default is `tol=1e-6`.
`cut.prop`	an optional numeric represents the maximum proportion of features with nonzero coefficients. Default is `cut.prop=1`.

Details

This function implements the GMIFS procedure for reducing dimension and building good classifiers in data with ordinal response. It is suitable in both traditional and high-dimensional data setting with no limitation on the number of covariates. The likelihood in GMIFS is constructed based on the cumulative logit ordinal model. Through the iterative procedure, the estimate of intercept $\boldsymbol{\alpha}$ remains consistent by using the initial estimation where $\alpha_c = \log \frac{P(Y_i<c)}{1-P(Y_i<c)}$.

Value

`beta.list`	a list contains the covariate with a nonzero penalized estimate at each iteration.
`num.nonzero`	a list contains the number of covariates with a nonzero penalized estimate at each iteration.

Author(s)

Jiayi Hou

References

KJ. Archer and J. Hou (2013). Generalized Monotone Incremental Forward Stagewise Method for Ordinal Response Prediction for High-dimensional datasets. Technical Report
T. Hastie, J. Taylor, R. Tibshirani, and G. Walther (2007). Forward Stagewise Regression and the Monotone Lasso. Electronic Journal of Statistics, 2007(1):1-29.

See Also

`glmpathcr` `glmnetcr`

Examples

```
### NOT RUN ###
# data <- read.csv("GSE10006.csv")
# x=data[,3:dim(data)[2]]
# y=data[,2]
# y <-factor(y, levels=c("non-smoker","smoker","early-COPD","COPD"),ordered=TRUE)
# fit <- forward.stagewise.cum(x, y, 1e-4, 1e-4, 1)
```

FSPenFixed — *Build a Penalized Fixed-effects Cumulative Logit Ordinal Model*

Description

This function identifies an active set of important features that is monotonically associated with an ordinal response and utilizes it to fit a penalized cumulative logit ordinal model for prediction and classification. It first calls `forward.stagewise.cum` to perform the GMIFS for data with ordinal response and yield a full solution of the penalized coefficient estimates of $\boldsymbol{\beta}$. At steps immediately preceding the step where a new feature enters the active set, a penalized ordinal model is fitted to update the estimate of intercept $\boldsymbol{\alpha}$ using the corresponding penalized estimates of $\boldsymbol{\beta}$. The model fitting criteria is calculated correspondingly. The optimal parsimonious model is selected based on the model fitting criteria. The predicted ordinal response is calculated for the best parsimonious model.

Usage

```
FSPenFixed(x, y, data, gene.name = NULL, beta.name = NULL,
                  epsilon = 1e-04, tol = 1e-06, cut.prop = 1,
                  criteria = "AIC", standardize = FALSE)
```

Arguments

`x`	a matrix contains all the predictors to be identified by GMIFS.
`y`	an ordered factor represents an ordinal response.
`data`	a data frame contains all variables occurring in the function.
`gene.name`	an optional vector contains the name of covariates which have a known estimate before entering into `FSPenFixed`. If you wish to use GMIFS performed by `forward.stagewise.cum` for building the active set and obtaining penalized estimates for features, use `gene.name=NULL`. Default is `gene.name=NULL`.
`beta.name`	an optional vector contains the coefficient estimates associated with `gene.name`. Default is `gene.name=NULL`.
`epsilon`	an optional numeric represents a small incremental amount to update $\boldsymbol{\beta}$ at each iteration. Default is `epsilon=1e-4`.
`tol`	an optional numeric represents a convergence criterion between successive log-likelihoods. Default is `tol=1e-6`.
`cut.prop`	an optional numeric represents the maximum proportion of features with nonzero coefficients. Default is `cut.prop=1`.
`criteria`	an optional character specifies the best parsimonious model is selected based on either AIC or BIC model fitting criteria. Default is `criteria="AIC"`.
`standardize`	an optional logic determines whether `x` needs to be standardized.

Details

The `FSPenFixed` function first calls `forward.stagewise.cum` to perform the GMIFS for ordinal response. Those results get passed to additional functions (`output.CumFS, steps, logLL`) to fit a series of penalized cumulative logit ordinal models. By using the intercept estimate $\hat{\boldsymbol{\alpha}}$ and corresponding penalized estimates $\boldsymbol{\beta}$ from the optimal parsimonious ordinal model, function `Predict` calculates the predicted ordinal response where the prediction error can be assessed by comparing it with the true ordinal response.

Value

`FSPenFixed` returns a list with the following components:

`alpha`	the intercept estimate from the optimal parsimonious ordinal model.
`beta.gene`	the penalized estimates of covariates with a nonzero coefficient identified by GMIFS and used to build the optimal parsimonious ordinal model.
`gene.name`	the name of the covariates with a nonzero coefficient identified by GMIFS and used to build the optimal parsimonious ordinal model.
`logLL`	the value of log-likelihood for the optimal parsimonious ordinal model.
`pred.table`	a contingency table compares the predicted and observed ordinal responses.
`fullpaths`	a forwardstagewise object contains the names, number and penalized estimates of covariates with a nonzero coefficient estimate at each iteration.

Author(s)

Jiayi Hou

References

J. Hou (2013). Regularization Methods for Predicting an Ordinal Response Using Longitudinal High-dimensional Genomic Data. Dissertation work.

Examples

```
### NOT RUN ###
# data <- read.csv("GSE10006.csv")
# data <- data[,-2]
# x=data[,3:dim(data)[2]]
# y=data[,2]
# y <-factor(y, levels=c("non-smoker","smoker","early-COPD","COPD"),ordered=TRUE)
# output <- FSPenFixed(x, y, data, tol=1e-5)
```

`FSPenFixedCV` *K-fold Cross-Validation for* `FSPenFixed`

Description

This function performs K-fold cross-validation to evaluate the performance of penalized cumulative logit ordinal model built by `FSPenFixed`.

Usage

```
FSPenFixedCV(x, y, data, kfold, gene.name = NULL,
                beta.name = NULL, epsilon = 1e-04,
                tol = 1e-06, cut.prop =1, criteria = "AIC",
               standardize = FALSE)
```

Arguments

`x`	a matrix contains all the predictors to be identified by GMIFS.
`y`	an ordered factor represents an ordinal response.
`data`	a data frame contains all variables occurring in the function.
`kfold`	a numeric specifies the number of partitions into which the data should be split.
`gene.name`	an optional vector contains the name of covariates which have a known estimate obtained other regularization models before entering into `FSPenFixed`. If you wish to use GMIFS performed by `forward.stagewise.cum` for building the active set and obtaining penalized estimates for features, leave `gene.name=NULL`. Default is `gene.name=NULL`.
`beta.name`	an optional vector contains the coefficient estimates associated with `gene.name`. Default is `gene.name=NULL`.
`epsilon`	an optional numeric represents a small incremental amount to update β. Default is `epsilon=1e-4`.
`tol`	an optional numeric represents a convergence criterion between successive log-likelihoods. Default is `tol=1e-6`.
`cut.prop`	an optional numeric represents the maximum proportion of features with nonzero coefficients. Default is `cut.prop=1`.
`criteria`	an optional character specifies the best parsimonious model is selected based on either AIC or BIC model fitting criteria. Default is `criteria="AIC"`.
`standardize`	an optional logic determines whether `x` needs to be standardized.

Details

It is of ultimate importance to assess model performance using the proper criteria such that the chosen model can be generalized to a broader range of data. Cross-validation performs a "split-fit-evaluate" procedure many times to obtain a robust assessment of model performance. A K-fold cross-validation performs by dividing the full dataset into K parts where the K-1 parts constructs a training set and the remaining 1 part forms a test set. The model is fitted using the training dataset and evaluated using the full dataset according to criterion such as: expected prediction error, consistency of variable selection and for ordinal response particular, Goodman and Kruskal's gamma statistic.

Value

`FSPenFixedCV` returns K lists with each having the following components:

`test.index`	the row names of observations in the test dataset.
`new.beta`	the penalized estimates of covariates with a nonzero coefficient identified by GMIFS and used to build the optimal parsimonious ordinal model in the training dataset.
`gene.name`	the name of the covariates with a nonzero coefficient identified by GMIFS and used to build the optimal parsimonious ordinal model in the training dataset.
`pred.table`	a contingency table compares the predicted and observed ordinal responses in the full dataset.

Note

For a large-scale or high-dimensional data, an alternation of this function may be appropriate to incorporate with certain parallel computing packages to accelerate the computing process.

Author(s)

Jiayi Hou

References

J. Hou (2013). Regularization Methods for Predicting an Ordinal Response Using Longitudinal High-dimensional Genomic Data. Dissertation work.
T. Hastie, R. Tibshirani, and J. Friedman (2009). The Elements of Statistical Learning: Data Mining, Inference and Prediction. Springer, New York, NY, 2nd edition.

Examples

```
### NOT RUN ###
# data <- read.csv("GSE10006.csv")
# x=data[,3:dim(data)[2]]
# y=data[,2]
# y <-factor(y, levels=c("non-smoker","smoker","early-COPD","COPD"),ordered=TRUE)
# output <- FSPenFixedCV(x, y, data, kfold=dim(data)[1])
```

`FSPenMixed` *Build a Penalized Mixed-effects Cumulative Logit Ordinal Model*

Description

This function identifies an active set of important features that is monotonically associated with an ordinal response and utilizes it to fit a penalized random coefficients/intercept ordinal response model with cumulative logit for prediction and classification. It first calls `forward.stagewise.cum` to perform the GMIFS for data with longitudinal ordinal response and yield a full solution of the penalized coeffcient estimates $\boldsymbol{\beta}$. At steps immediately preceding the step where a new feature enters the active set, a penalized fixed-effects ordinal model with cumulative logit is fitted to update the estimate of intercept $\boldsymbol{\alpha}$ using the penalized estimates of $\boldsymbol{\beta}$. The model fitting criteria is calculated correspondingly. The best parsimonious penalized fixed-effects ordinal model is then selected based on the model fitting criteria. To take the subject-specific intercept and slope into consideration, `FSPenMixed` then calls function `ordinal.mixed.model` to fit a penalized random coefficient/intercept ordinal response model using features in the active set. This set of features along with corresponding penalized estimates remain the same as those in the best penalized fixed-effects ordinal model. The estimate of intercept $\boldsymbol{\alpha}$ and model fitting criteria are then updated again. Along with the empirical Bayes estimates of the random effects, the predicted ordinal response can be calculated accordingly.

Usage

```
FSPenMixed(response.name, predictor.name, id.name, time.name, data,
                 nGauss = NULL, gene.name = NULL, beta.name = NULL,
                 link = c("Cumulative", "Adjacent", "Forward CR", "Backward CR"),
                 model = c("Random Intercept", "Random Coefficient",
                 "Indep Random Coefficient"), Adaptive = c("TRUE", "FALSE"),
                  Cholesky = c("TRUE", "FALSE"), x,  y,  epsilon = 1e-04,
                  tol = 1e-6, cut.prop =1, range = 0,
                 criteria = "AIC", standardize = FALSE)
```

Arguments

`response.name`	a character indicates the name of ordinal response in data.
`predictor.name`	a character vector indicates the names of fixed-effects in data. Note this set of predictors differ from that specified in `x` matrix. The predictors specified here do not select by GMIFS and are only used when fitting a penalized random coefficient/intercept ordinal response model. If no additional predictors are included rather than those in `x` matrix, use `predictor.name=NULL`.
`id.name`	a character indicates name of subject in data to specify the subject-specific intercept.
`time.name`	a character indicates the name of time covariate in data to specify the subject-specific slope.
`data`	a data frame contains all variables occurring in the function.
`nGauss`	an optional numeric indicates the number of abscissas from Hermite polynomial used to approximate the marginal likelihood. Default is `nGauss=10`.
`gene.name`	an optional vector contains the name of covariates which have a known estimate before entering into the `ordinal.mixed.model` function. Default is `gene.name=NULL`.
`beta.gene`	an optional vector contains the coefficient estimates associated with `gene.name`. Default is `gene.name=NULL`.
`link`	an optional character specifies link function in random coefficient/intercept ordinal response model. Currently the model with `link="Cumulative"` link can be fitted. Default is `link="Cumulative"`.
`model`	an optional character specifies the model type which defines the covariance structure of the random effect(s). Default is `model="Random Intercept"`.
`Adaptive`	an optional character determines whether an adaptive or nonadaptive Gauss-Hermite Quadrature numerical method should be used to approximate the marginal likelihood. Default is `Adaptive="TRUE"`.
`Cholesky`	an optional character determines whether a Cholesky decomposition should be used when estimating the covariance matrix of the random effect(s). Default is `Cholesky="TRUE"`.
`x`	a matrix contains all the predictor variables to be identified by GMIFS.
`y`	an ordered factor represents an ordinal response where `y=data[, response.name]`.
`epsilon`	an optional numeric represents a small incremental amount to update β. Default is `epsilon=1e-4`.
`tol`	an optional numeric represents a convergence criterion between successive log-likelihoods. Default is `tol=1e-6`.
`cut.prop`	an optional numeric represents the maximum proportion of features with nonzero coefficients. Default is `cut.prop=1`.
`range`	an optional numeric specifies `2*range+1` penalized random coefficient/intercept ordinal response models to be fitted. Default is `range=0`.
`criteria`	an optional character specifies the best parsimonious model is selected based on either AIC or BIC model fitting criteria. Default is `criteria="AIC"`.
`standardize`	an optional logic determines whether matrix `x` needs to be standardized.

Details

The Function `FSPenMixed` consists of two parts: the GMIFS part and the penalized random coefficient/intercept ordinal response model part. In the GMIFS part, it first calls function `forward.stagewise.cum` to identify an active set of important features that is monotonically associated with a longitudinal ordinal response. These results get passed to additional functions (`output.CumFS, steps, logLL`) to update the estimates of intercept α and select the best parsimonious penalized fixed-effects ordinal model. These results get passed to function `ordinal.mixed.model` where a penalized ordinal coefficient/intercept model is fitted. The estimates of intercept α, fixed-effects specified by `predictor.name` and the covariance matrix are obtained by optimizing the approximated marginal likelihood using general-purpose iterative optimization method. The empirical bayes estimate of the random effect as well as the predicted ordinal response are obtained using function `ConditionalMode`.

Value

`FSPenMixed` returns a list with the following components:

`parm`	the estimate of intercept, fixed-effects and covariance matrix.
`new.beta`	the penalized estimates of covariates with a nonzero coefficient identified by GMIFS and used to build the optimal ordinal model.
`gene.name`	the name of the covariates with a nonzero coefficient identified by GMIFS and used to build the optimal ordinal model.
`AIC`	the value of AIC for the optimal random coefficient/intercept ordinal response model.
`BIC`	the value of BIC for the optimal random coefficient/intercept ordinal response model.
`logLL`	the value of log-likelihood for the optimal parsimonious random coefficient/intercept ordinal response model.
`pred.table`	a contingency table compares the predicted and observed ordinal responses.
`fullpaths`	a forwardstagewise object contains the names, number and penalized estimates of covariates with a nonzero coefficient estimate at each iteration.

Author(s)

Jiayi Hou

References

J. Hou (2013). Regularization Methods for Predicting an Ordinal Response Using Longitudinal High-dimensional Genomic Data. Dissertation work.

Examples

```
### NOT RUN###
# data <- read.csv("gluegrant.csv")
# output1 <-  FSPenMixed(response.name="RENAL3",
#                       predictor.name="SAMPLE_STUDYSTART_DAYS",
#                       id.name="PATIENT_ID",
#                        time.name="SAMPLE_STUDYSTART_DAYS", data,
#                       x=data.matrix(data[, 15:dim(data)[2]]),
#                        y=data[,"RENAL3"],
#                       model="Random Coefficient", range=0, cut.prop=0.0015)
```

`FSPenMixedCV` *Cross-validation for* `FSPenMixed`

Description

This function performs cross-validation to evaluate the performance of penalized random coefficient/intercept ordinal response model build in `FSPenMixed`.

Usage

```
FSPenMixedCV(response.name, predictor.name, id.name, time.name, data,
                     kfold,  nGauss = NULL, gene.name = NULL, beta.name = NULL,
                     link = c("Cumulative", "Adjacent", "Forward CR", "Backward CR"),
                     model = c("Random Intercept", "Random Coefficient",
                    "Indep Random Coefficient"), Adaptive = c("TRUE", "FALSE"),
                     Cholesky = c("TRUE", "FALSE"), x, y, epsilon = 1e-04, tol = 1e-06,
                     cut.prop = 1, range = 0, criteria = "AIC", standardize = FALSE)
```

Arguments

`response.name`	a character indicates the name of ordinal response in data.
`predictor.name`	a character vector indicates the names of fixed-effects in data. Note this set of predictors differ from that specified in `x` matrix. The predictors specified here do not select by GMIFS and are only used when fitting a penalized random coefficient/intercept ordinal response model. If no additional predictors are included rather than those in `x` matrix, use `predictor.name=NULL`.
`id.name`	a character indicates name of subject in data to specify the subject-specific intercept.
`time.name`	a character indicates the name of time covariate in data to specify the subject-specific slope.
`kfold`	a numeric specifies the number of partitions into which the number of subjects in data should be split to for cross-validation.
`data`	a data frame contains all variables occurring in the function.
`nGauss`	an optional numeric indicates the number of abscissas from Hermite polynomial used to approximate the marginal likelihood. Default is `nGauss=10`.
`gene.name`	an optional vector contains the name of covariates which have a known estimate before entering into the `ordinal.mixed.model` function. Default is `gene.name=NULL`.
`beta.gene`	an optional vector contains the coefficient estimates associated with `gene.name`. Default is `gene.name=NULL`.
`link`	an optional character specifies link function in random coefficient/intercept ordinal response model. Currently the model with `link="Cumulative"` link can be fitted. Default is `link="Cumulative"`.
`model`	an optional character specifies the model type which defines the covariance structure of the random effect(s). Default is `model="Random Intercept"`.
`Adaptive`	an optional character determines whether an adaptive or nonadaptive Gauss-Hermite Quadrature numerical method should be used to approximate the marginal likelihood. Default is `Adaptive="TRUE"`.

`Cholesky`	an optional character determines whether a Cholesky decomposition should be used when estimating the covariance matrix of the random effect(s). Default is `Cholesky="TRUE"`.
`x`	a matrix contains all the predictor variables to be identified by GMIFS.
`y`	an ordered factor represents an ordinal response where `y=data[, response.name]`.
`epsilon`	an optional numeric represents a small incremental amount to update β. Default is `epsilon=1e-4`.
`tol`	an optional numeric represents a convergence criterion between successive log-likelihoods. Default is `tol=1e-6`.
`cut.prop`	an optional numeric represents the maximum proportion of features with nonzero coefficients. Default is `cut.prop=1`.
`range`	an optional numeric specifies `2*range+1` penalized random coefficient/intercept ordinal response models to be fitted. Default is `range=0`.
`criteria`	an optional character specifies the best parsimonious model is selected based on either AIC or BIC model fitting criteria. Default is `criteria="AIC"`.
`standardize`	an optional logic determines whether matrix `x` needs to be standardized.

Details

Few publications have discussed cross-validation procedures for longitudinal data. We conduct the quasi K-fold cross-validation for longitudinal data by dividing the total subjects instead of observations into K partitions where the K-1 partitions from a training set and the remaining 1 part forms a test set. Since the number of repeated measurements per subject can vary, the number of observations in the test set varies with each split. The model is then fitted using the training dataset and evaluated using the full dataset according to criterion such as: expected prediction error, consistency of variable selection and for ordinal response particular, Goodman and Kruskal's gamma statistic.

Value

`FSPenMixedCV` returns K lists with each having the following components:

`test.index`	the `id.name` of subjects in the test dataset.
`new.beta`	the penalized estimates of covariates with a nonzero coefficient identified by GMIFS and used to build the optimal parsimonious ordinal model in the training dataset.
`gene.name`	the name of the covariates with a nonzero coefficient identified by GMIFS and used to build the optimal parsimonious ordinal model in the training dataset.
`pred.table`	a contingency table compares the predicted and observed ordinal responses in the full dataset.

Note

For a large-scale or longitudinal high-dimensional data, an alternation of this function may be appropriate to incorporate with certain parallel computing packages to accelerate the computing process.

Author(s)

Jiayi Hou

References

J. Hou (2013). Regularization Methods for Predicting an Ordinal Response Using Longitudinal High-dimensional Genomic Data. Dissertation work.
T. Hastie, R. Tibshirani, and J. Friedman (2009). The Elements of Statistical Learning: Data Mining, Inference and Prediction. Springer, New York, NY, 2nd edition.

Examples

```
### NOT RUN ###
# data <- read.csv("gluegrant.csv")
# PenMixedCV <- FSPenMixedCV (response.name="RENAL3",
#                             predictor.name="SAMPLE_STUDYSTART_DAYS",
#                            id.name="PATIENT_ID",
#                             time.name="SAMPLE_STUDYSTART_DAYS" ,
#                             data, kfold=10,
#                            x=data.matrix(data[, 14:dim(data)[2]]),
#                             y=data[,"RENAL3"])
```

`ordinal.mixed.model` *Random Coefficient/Intercept Ordinal Response Model*

Description

This function can fit either a penalized or traditional random coefficient/intercept ordinal response model to capture the time-dependent trends in data. When the options `gene.name=NULL, beta.gene=NULL`, a traditional random intercept/coefficient ordinal response model is fitted using the covariates specified in `predictor.name` as the fixed effects. When `gene.name` and `beta.gene` are specified, a penalized random intercept/coefficient ordinal response model is fitted using both the covariates specified in `predictor.name` and `gene.name` as fixed effects. Only the features specified in `predictor.name` are estimated during the fitting process and the known estimates specified in `beta.gene` remain constant.

Usage

```
ordinal.mixed.model(response.name, predictor.name, id.name, time.name, data,
                              nGauss = NULL, gene.name = NULL, beta.gene = NULL,
                              link = c("Cumulative", "Adjacent", "Forward CR", "Backward CR"),
                              model = c("Random Intercept", "Random Coefficient",
                              "Indep Random Coefficient"), Adaptive = c("TRUE", "FALSE"),
                              Cholesky = c("TRUE", "FALSE"), ...)
```

Arguments

`response.name`	a character indicates ordinal response name in data.
`predictor.name`	a character vector indicates the names of fixed-effects in data. For null model, use `predictor.name=NULL`
`id.name`	a character indicates subject name in data to specify the subject-specific intercept.
`time.name`	a character indicates the time covariate in data to specify the subject-specific slope.

`data`	a data frame contains all variables occurring in the function.
`nGauss`	an optional numeric indicates the number of abscissas from Hermite polynomial used to approximate the marginal likelihood. Default is `nGauss=10`.
`gene.name`	an optional vector contains the name of covariates which have a known estimate obtained from GMIFS or other regularization models before entering into the `ordinal.mixed.model` function. Default is `gene.name=NULL`.
`beta.gene`	an optional vector contains the coefficient estimates associated with `gene.name`. Default is `gene.name=NULL`.
`link`	an optional character specifies link function in random coefficient/intercept ordinal response model. For traditional ordinal model when `gene.name=NULL, beta.gene=NULL`, there are four types of models can be fitted by specifying the link to be one of `("Cumulative", "Adjacent", "Forward CR", "Backward CR")`. For penalized ordinal model, currently the model with `link="Cumulative"` link can be fitted. Default is `link="Cumulative"`.
`model`	an optional character specifies the model type which defines the covariance structure of the random effect(s). Default is `model="Random Intercept"`.
`Adaptive`	an optional character determines whether an adaptive or nonadaptive Gauss-Hermite Quadrature numerical method should be used to approximate the marginal likelihood. Default is `Adaptive="TRUE"`.
`Cholesky`	an optional character determines whether a Cholesky decomposition should be used when estimating the covariance matrix of the random effect(s). Default is `Cholesky="TRUE"`.
`...`	additional arguments from any of the internal functions

Details

This function fits either a penalized or traditional random coefficient/intercept ordinal response model. A generalized linear mixed model approach is implemented to allow the mean response varying across subjects where the fixed-effects have a subject-specific interpretation conditional on the random effect. The estimates of the fixed-effects $\boldsymbol{\beta}$, intercept $\boldsymbol{\alpha}$ and covariance matrix are obtained by optimizing the approximate form of the marginal likelihood using the general-purpose iterative optimization methods. The estimates of random effects $\mathbf{b_i}$ are obtained through empirical Bayes method. `ordinal.mixed.model` consists of two parts: approximation and optimization. In approximation part, the set of abscissas and weights from Hermite polynomial is generated by function `ghq`. Functions `em.bayes, ghq` are used for Gauss-Hermite Quadrature methods. The data matrix generated by functions `index.all` and `linear.predictor` are expanded according to the number of quadrature points. Function `GHQ.intu1` gathers all information needed to calculate the approximated marginal likelihood. In optimization part, the approximated marginal likelihood is optimized by function `optim` or `optimx` given the initial parameter estimates specified by user or generated by function `initial.value` to obtain the estimate for the fixed-effects as well as variance components. Function `chol2var` transforms back the estimate of variance and its standard error if a Cholesky decomposition is used in the optimization process.

Value

`ordinal.mixed.model` returns a list with the first element being a data.frame contains the coefficient estimates, standard error at convergence as well as the corresponding statistical inferences; the second element contains value of the function at convergence.

Author(s)

Jiayi Hou

References

D. Hedeker and R. Gibbons (2006). Longitudinal Data Analysis. John Wiley & Sons, Inc, Hoboken, NJ.

J. Hou (2013). Regularization Methods for Predicting an Ordinal Response Using Longitudinal High-dimensional Genomic Data. Dissertation work.

See Also

`ordinal` `lme4`

Examples

```
### NOT RUN ###
# data <- read.csv("NIMH Schizophrenia.csv")
# response.name = "imps79o"
# predictor.name= c('tx','sweek','txswk')
# id.name = "id"
# time.name = 'sweek'

# Ordinal random intercept and random coefficient models with cumulative logit #
# fit1 <- ordinal.mixed.model (response.name, predictor.name, id.name,
#                              time.name, data, nGauss=15)
# fit2 <- ordinal.mixed.model (response.name, predictor.name, id.name, time.name,
#                              data, model="Random Coefficient")

# Ordinal random intercept and random coefficient models with adjacent category #
# fit1 <- ordinal.mixed.model (response.name, predictor.name, id.name, time.name,
#                              data, link="Adjacent", nGauss=5)
# fit2 <- ordinal.mixed.model (response.name, predictor.name, id.name, time.name,
#                              data, model="Random Coefficient", link="Adjacent")
```

Index

Figure 5.1: Flowchart for function `FSPenFixed` in R package `ordinalmixed`. The blue circle represents input/output and the cyan rectangle represents an R function. The `FSPenFixed` function first calls function `forward.stagewise.cum` to perform steps 1,2 and 3 described in GMIFS for ordinal response with high-dimensional data. Those results get passed to additional functions (`output.CumFS, steps, logLL`) to perform step 4. By using $\hat{\boldsymbol{\alpha}}$ and $\hat{\boldsymbol{\beta}}_{biased}$ from the optimal parsimonious ordinal model, function `Predict` calculates the predicted ordinal response where the prediction error can be estimated by comparing it with the true ordinal response.

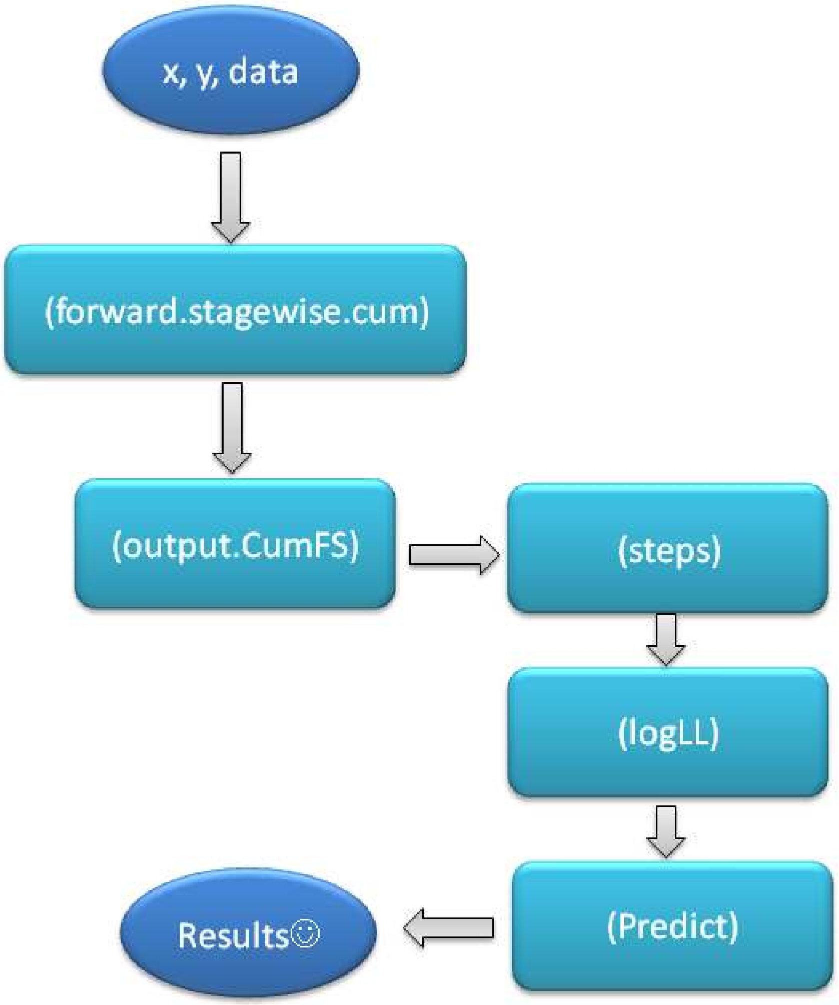
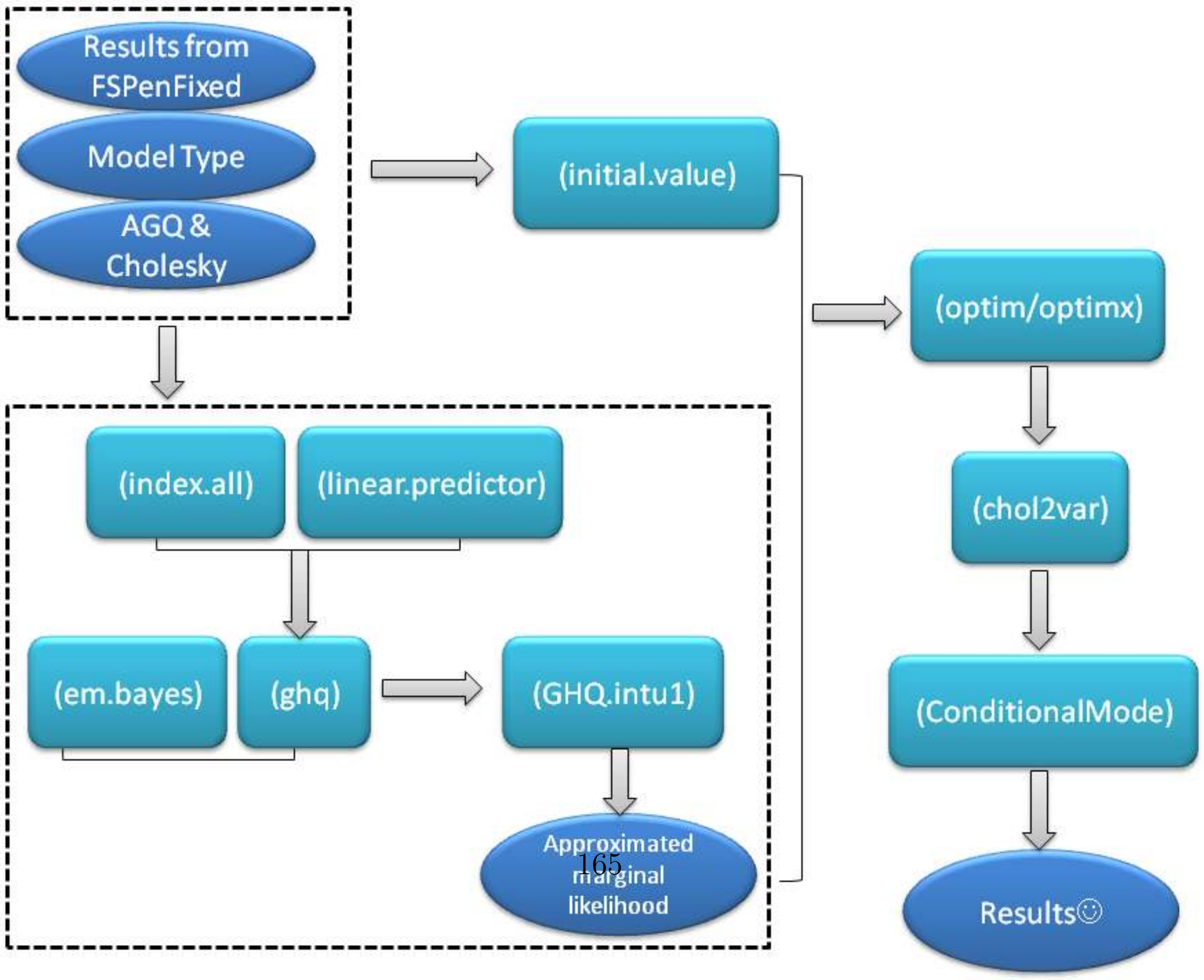


Figure 5.2: Flowchart for function `FSPenFixed` and `ordinal.mixed.model` in R package `ordinalmixed`. The blue circle represents input/output and the cyan rectangle represents an R function. Function `FSPenMixed` consists of two parts: the GMIFS part and the ordinal random coefficient/intercept model part. The ordinal random coefficient/intercept model is fitted by function `ordinal.mixed.model`, which is primarily shown in this flowchart. The GMIFS part is the same as that in function `FSPenFixed` where it first calls function `forward.stagewise.cum` to perform steps 1, 2, 3 and 4 described in GMIFS for modeling a longitudinal ordinal response in the presence of high-dimensional data. Those results get passed to additional functions (`output.CumFS, steps, logLL`) to update the intercept estimates $\hat{\boldsymbol{\alpha}}$ and decide the optimal set of features with nonzero coefficient estimates. The corresponding results get passed to function `ordinal.mixed.model` where a penalized ordinal coefficient/intercept model is fitted. This function consists of two parts: approximation and optimization. In approximation part, the set of abscissas and weights from Hermite polynomial is generated by function `ghq` for nonadaptive and functions `em.bayes, ghq` for adaptive Gauss-Hermite Quadrature methods. The data matrix generated by functions `index.all` and `linear.predictor` are expanded according to the number of quadrature points. Function `GHQ.intu1` gathers all information needed to calculate the approximated marginal likelihood. In the optimization part, the approximated marginal likelihood is optimized by function `optim` or `optimx` given the initial parameter estimates specified by the user or generated by function `initial.value` to obtain the estimate for the fixed-effects as well as variance components. Function `chol2var` transforms back the estimate of variance and its standard error if a Cholesky decomposition is used in the optimization process. The empirical Bayes estimate of the random effect as well as the predicted ordinal response are calculated in function `ConditionalMode`.



5.6 Simulations to Evaluate the Proposed Model

5.6.1 Simulation for High-dimensional Data

We simulated a high-dimensional dataset to include $n = 90$ samples and $p = 1000$ features. We generated the samples to belong to three groups where each group includes 30 samples, mimicking an ordinal response having three levels. We first generated each feature independently using a standard normal distribution, that is, $\mathbf{x_j} \sim N(0,1)$. Ten features were randomly selected among the 1000 features to have a nonzero coefficient. For each of these 10 features, we generated group 1 observations using a standard normal distribution $N(0,1)$; group 2 observations using a normal distribution with mean ± 1.5 and variance 1; and group 3 observations using a normal distribution with mean ± 3 and variance 1. We used GMIFS algorithm to reduce the dimensionality of the dataset and to construct a parsimonious model. The N-fold cross-validation procedure was conducted to evaluate model performance with respect to the following aspects: prediction accuracy, consistency, and correctness in detecting the truly important features as well as the similarity of the orderings of the observed and predicted ordinal response.

We repeated the simulation study 100 times and the results are reported in Table 5.1 from which consistently good results were observed in models selected using both AIC and BIC criteria. The important features identified has a median of 7 with ranging from 2 to 10, where the number of important features assumed is 10. In addition, the false features identified by GMIFS has a median of 2 with ranging from 0 to 18 according to AIC criteria and a median of 0 with ranging from 0 to 4 according to BIC criteria. It indicates GMIFS has the ability to detect a subset that contains the important features. We will argue in the discussion section that GMIFS serves as a method to build parsimonious model for prediction and classification rather than merely a variable selection procedure.

Table 5.1: Results from 100 simulation studies where each included 90 samples and 1000 independent features. The median and range are reported for N-fold cross-validation prediction error, N-fold cross-validation Goodman and Kruskal's gamma, and the number of true features detected.

Criteria	N-fold CV error	N-fold CV gamma	True features detected	False Features detected
AIC	6.67%[1.11%, 34.4%]	1[0.89, 1]	7[2, 10]	2[0, 18]
BIC	6.67%[1.11%, 33.3%]	1[0.89, 1]	7[2, 10]	0[0,4]

5.6.2 Simulation for Longitudinal High-dimensional Data

It is very challenging to simulate longitudinal high-dimensional data with an ordinal response due to the complex relationship between the large number of features and the small number of response categories. Rather than simulating the predictors and ordinal responses independently without considering their dependencies, we adopt the longitudinal high-dimensional features from a small time-course microarray experiment designed for investigating the expression response of human T cells to phorbol myristate acetate (PMA) and ionomicin treatment [Rangel et al., 2004] and then build the ordinal response correspondingly using the latent variable approach. This small time-course microarray dataset is available in the R package `longitudinal` [Opgen-Rhein and Strimmer, 2013]. The data contains the temporal expression of 58 genes from 10 samples measured at 10 unequally spaced time points (0, 2, 4, 6, 8, 18, 24, 32, 48, 72 hour). To simulate the ordinal response, we randomly select 6 of 58 genes to have nonzero coefficients $\boldsymbol{\beta}$ and the magnitude of the six coefficients are assigned with random numbers between -5 and 5. Then the latent response $\mathbf{y}^*$ can be constructed using $\mathbf{y}^* = \mathbf{x}^T\boldsymbol{\beta} + \epsilon$ where $\epsilon \sim N(0,1)$. The latent variable $\mathbf{y}^*$ is divided into three equal parts to form an ordinal response.

We repeated the simulation study 25 times to examine the performance of GMIFS to detect the true nonzero features. Since the number of observations is small and the ordinal response is pseudo, we were unable to fit the penalized ordinal random coefficient model. Therefore, the classification accuracy and similarity of orderings were not evaluated. However, in all these simulation studies, the number of true nonzero features detected by GMIFS has a median of 4 with the range from 1 to 6, which again demonstrates one attribute of GMIFS: it has the ability to detect a subset of features which covers a proportion of the true features with nonzero coefficients. It also demonstrates the valid usage of GMIFS on data with repeated measurements.

5.7 Some Discussion

We have proposed a learning method to construct a parsimonious model to classify and predict the ordinal outcome using traditional and longitudinal high-dimensional data. In the past, several authors have developed methodologies for analyzing time-course gene expression data. For example, Yuan and Kendziorski [2006] proposed Hidden Markov Models to detect differential expression patterns in genes over time under multiple biological conditions. Tai and Speed [2006] developed a variation of Hotelling T^2-statistics and the multivariate empirical Bayes statistics to detect temporal changes of certain genes in one- and two-sample cases where the expression level changes often motivate the experiments. Zhou et al. [2010] developed a factorial design by pooling information across the time course while accounting for multiple testing and nonnormality of the microarray data to effectively extract dynamic features. Zhang et al. [2013] proposed a novel 'optimal direction' approach to extract useful features to perform binary classification problems in time-course microarray setting. However, most of the approaches for extracting temporal patterns of differential expression requires fixed and equal number of time points between heterogeneous samples which tremendously limits its usage. Besides, most of the approaches used the average or pooled feature information across time which ignores the correlation structure and dependencies between measurements. In addition, for ordinal outcomes with more than two categories, currently there is no solution for classification and prediction using longitudinal high-dimensional data.

Our work incorporates the classical random coefficient model with the cutting-edge machine learning algorithm to provide a two-step disruptive method to address the very challenging classification and prediction problem in data with a large number of features as well as complex correlation structure. In the variable selection stage, the proposed method provides a flexible structure to accommodate a tremendous amount of features including genomic predictors, clinical information and baseline demographic variables and be able to evaluate them simultaneously. In the model fitting stage, the method utilizes the correlations and dependencies in the time-varying process and can easily adapt to a varying number of measurements collected at different time points which enormously broadens the usage. The validity and effectiveness of our model are demonstrated using simulation studies which are described in chapter 6. Currently, we use cross-validation as the only way to evaluate the model complexity and variable selection consistency. Alternative methods such as the Bootstrap[Efron and Tibshirani, 1997], which is considered as a smoothed version of cross-validation that reduces variability of a near-unbiased expected prediction error, will be used in the future.

It is worth while pointing out that we used the model fitting criteria of minimal AIC and BIC to select the final model rather than the minimal prediction error via a cross-validation procedure in the LASSO and LAR. The reason we did not select the model based on prediction error estimated from cross-validation is simply due to computational time. Since a typical forward stagewise procedure contains hundreds of thousands steps, it usually requires long computational time before converging in the R programming environment, making cross-validation unpractical for model selection for high-dimensional data. As discussed in [Tibshirani, 1996] and proved by Leng et al. [2006], the final model associated with minimal prediction error selected by LASSO may not contain all the important variables; that is, the ultimate parsimonious model is not necessarily the true model. This conclusion also holds for the forward stagewise method which is demonstrated in the simulation study. In addition, as discussed in Section 5.1, the forward stagewise method provides a greedy approximation to the proposed function which selects the features associated with immediate descent and leads to local optimal at each iteration. However, since a greedy optimization procedure does not adjust and modify its previous paths, the features that largely decrease the proposed function at an early stage may not be part of the optimal solution in a long run. Therefore, similar to LASSO and LAR, the forward stagewise method is also a way to select good classifiers for building a parsimonious model for classification and prediction purposes but may not serve as an optimal variable selection method.

Predictive modeling is so important in areas such as computational biology, bioinformatics and translational medicine for finding potential biomarkers. In this thesis, we primarily applied the proposed method to microarray gene expression data, but it can be easily applied to other types of biological data, such as sequencing and GWAS data.

Chapter 6

Application of Proposed Methodology

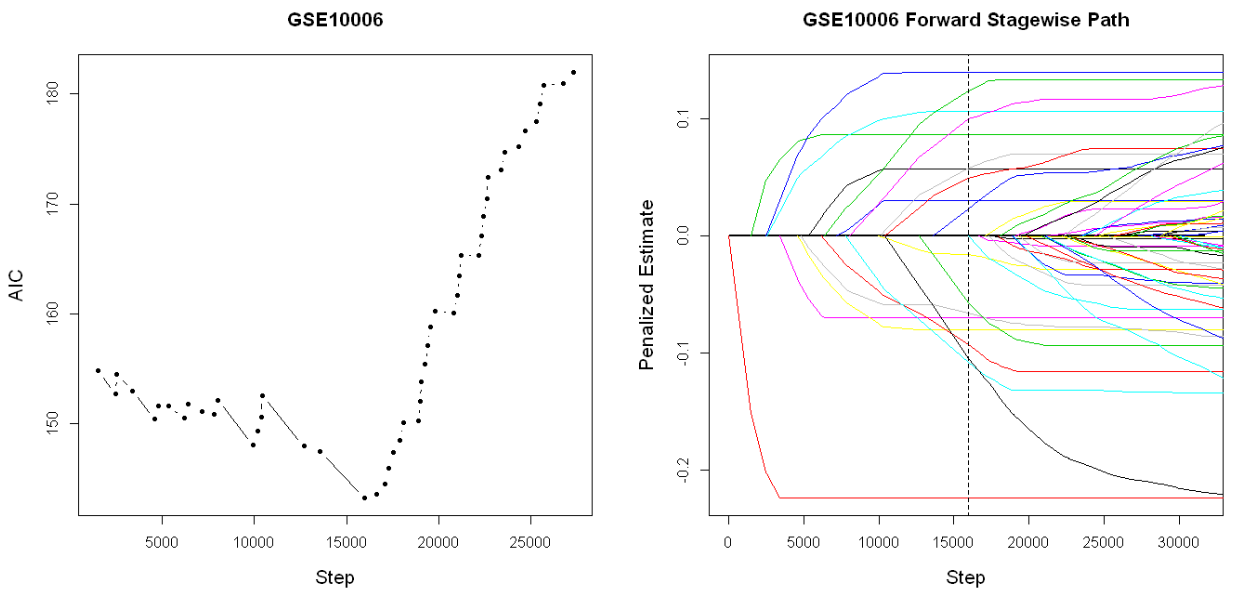
In Chapter 5, we introduced the Generalized Monotone Incremental Forward Stagewise (GMIFS) method for building a parsimonious ordinal response model using high-dimensional data. In addition, we incorporated the random coefficient model with the GMIFS method to build a penalized ordinal model with random effects using longitudinal high-dimensional data. The proposed models were applied to two simulated datasets. In this chapter, we apply the proposed methods to two real microarray datasets: an Affymetrix gene expression dataset that included 58 subjects in a COPD study (GSE10006) and the Glue Grant data from a burn injury study. The former is a study consisting of gene expression data at one time point whereas the latter is a longitudinal study where samples were procured from each patient over time. Specifically, in Section 6.1 we describe the results from the COPD study which was performed to identify a subset of important genes associated with smoking behavior and useful for classifying COPD stage. In Section 6.2, we present the results from the burn injury study where a set of good classifiers are detected to capture the ordinal trend developing along with time. For both examples, the model performance is examined using cross-validation. For the probe sets having nonzero coefficients, probe annotation data was used to map probeset ID to gene symbol and to identify corresponding information in public repositories such as protein products, associated diseases, and existing drugs that target the gene to make our discoveries meaningful and interpretable to clinical researchers. Additional discussion and conclusions are included in Section 6.3.

6.1 Application to the Smoking Study

Chronic Obstructive Pulmonary Disease (COPD) is a lung disease most commonly caused by tobacco smoke and is the third leading cause of death in the U.S. However, the mechanism by which smoke causes impairment and breakdown of lung tissue is still unclear and in fact, only a proportion of smokers ($10\% - 15\%$) eventually develop COPD [Mayer and Newman, 2001]. It is widely accepted that the genetic susceptibility combined with environmental exposure which varies by individual contribute to this difference. A study conducted by researchers at Weill Cornell Medical College assumed cigarette smoking may influence the lectin gene expression in the small airway epithelium which increases the risk of bacterial infection. A microarray experiment was designed to test this hypothesis. A total of 58 patients from whom airway epithelial cells were collected including 13 healthy nonsmoker, 18 smoker, 13 early COPD patients, and 14 COPD patients, were included in this study. The samples were hybridized to Affymetrix Human Genome U133 Plus 2.0 microarrays. For each sample, 54657 probe set expression levels were obtained. The data from this study has been deposited into Gene Expression Omnibus, GSE10006 (`http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE10006`). The quality of the microarrays was assessed by the $3' : 5'$ ratios of housekeeping gene GAPDH. No sample revealed quality issues as all ratings were close to 1 thus all 58 samples were used in the analysis.

We first applied the GMIFS algorithm to the full dataset collected from the smoking study (GSE10006 data) setting the two parameters of increment amount $\epsilon = 1 \times 10^{-4}$ and the convergence criteria to $\delta = 1 \times 10^{-4}$. All features were standardized to have mean 0 and unit norm. The initial intercept $\boldsymbol{\alpha}$ was taken to be $\alpha_c = \log \frac{\sum_{c=1}^{C} P(Y \leq c)}{1 - \sum_{c=1}^{C} P(Y \leq c)}$. Upon convergence, 83 probe sets with nonzero coefficient estimates were included in the model. To update the estimate of intercept $\boldsymbol{\alpha}$ and get a more accurate model fitting criteria for selecting the best parsimonious model, we then estimated $\boldsymbol{\alpha}$ for those steps immediately preceding the step where a new feature entered the active set of predictor. It is worth emphasizing that only $\boldsymbol{\alpha}$ is updated for these steps while the penalized estimate of $\boldsymbol{\beta}$ is fixed and used to update $\boldsymbol{\alpha}$. Therefore, a total of 83 penalized ordinal models were fitted and the corresponding model fitting criteria were also calculated. The model with optimal fitting criteria (*e.g.* minimal AIC) was selected as the best parsimonious model and the corresponding parameter estimates $\hat{\boldsymbol{\alpha}}$ and $\hat{\boldsymbol{\beta}}_{biased}$ were used for calculating the fitted ordinal response. Figure 6.1 presents this selection process: on the left panel, the model fitting criteria AIC reaches the minimal at 15965 steps; on the right panel, each colored line represents one probe with nonzero coefficient and the dashed vertical line represents the cutoff at around 16000 step. The probe sets associated with the colored line left of the dashed vertical line have nonzero penalized estimates at the threshold. In this scenario, 19 probe sets enter into the active set and are considered as 'good classifiers'. These probe sets were then used to classify stage of COPD under an ordinal model framework. These probe sets performed well in distinguishing the non-smokers from the COPD patients as shown in Table 6.1. Also from the contingency table, a large proportion of misclassifications were observed in the early COPD and smoker groups which implies the expression levels of the selected probe sets are similar in these two groups. As mentioned earlier, one prominent advantage in analyzing data a multi-category outcome under the ordinal model framework rather than using a pairwise comparison approach is its superiority in detecting important features associated with monotonic trends. Here, our proposed model identified a set of probe sets whose expression levels were monotonically associated with tobacco usage.

Figure 6.1: The model fitting paths (left) and the regularization profile (right) plots for the GSE10006 data. In each plot, the horizontal-axis is the step the forward stagewise algorithm has undertaken and the vertical-axes are the model fitting criteria AIC (left) and the penalized estimate for the coefficients (right).



We also performed N-fold cross-validation to obtain a robust estimate of the prediction

Table 6.1: Contingency Table for the observed and predicted COPD category using GSE10006 data. The predicted ordinal response was calculated using the penalized estimates obtained at 15965 steps.

Pred/Obs	COPD	EarlyCOPD	Smoker	Non-smoker
COPD	11	0	0	0
EarlyCOPD	2	18	8	0
Smoker	0	0	2	2
Non-smoker	0	0	3	12

error, variable selection as well as the Goodman and Kruskal's gamma. From the cross-validation procedure, an expected prediction accuracy of 74.1%(55.2%−87.9%) was obtained where a large proportion of misclassifications was due to the confusion between the smoker and early COPD groups. A high Goodman and Kruskal's gamma 0.99(0.97 − 1.00) was also obtained which indicates strong association between the observed and predicted ordinal scales. We present the important probe sets having a non-zero coefficient in at least one of the cross-validation fits in descending order of CV% in Table 6.2. CV% represents the percentage of times a probe set was identified by GMIFS in the cross-validation models. Each probe set name was matched to its gene symbol, chromosomal location, molecular function, associated disease, and drugs or chemical compounds either targeted or related. This information was mainly obtained from two public repositories: The Universal Protein Resource database and the GeneCard database. Several genes detected have been reported to be associated with lung cancer, inflammatory and other severe diseases. In a study to examine the association of 4-OHEs (estrogen metabolites) level and effect of tobacco smoke exposure, [Peng et al., 2013] concluded that tobacco smoke accelerates the level of 4-OHEs which is primarily produced by CYP1B1 and thus inhibition of CYP1B1 can be a promising strategy for the prevention and treatment of lung cancer. The SPRR1B gene is related to cell differentiation which leads to non-small-cell lung cancer. This was discovered in a microarray study by Woenckhaus et al. [2006] when comparing samples from surgically resected and microdissected of non-small-cell lung cancers, matched normal bronchial epithelium and peripheral lung tissue among 22 smokers and 5 non-smokers.

Table 6.2: Probe sets having non-zero coefficients in the cumulative logit models by GMIFS from N-fold cross-validation procedure using GSE10006 Data. The probe sets on the Affymetrix Human Genome U133 Plus 2.0 Array were mapped to Gene Symbol using R package `hgu133a2.db`. Each gene's molecular function information was obtained from the The Universal Protein Resource database (`http://www.uniprot.org/`). The associated disease and drug information were obtained from GeneCard database (`http://www.genecards.org`).

No.	AFFYprobe	SYMBOL	CV%	Location	Type	Disease	Drugs
1	205281_s_at	PIGA	100%	Xp22.1	glycosyltransferase transferase	hemoglobinuria anemia lymphoma, etc	3-Dehydrosphinganine aerolysin n-acetylglucosamine,etc
2	202435_s_at	CYP1B1	91.4%	2q22.2	monooxygenase oxidoreductase	congenital glaucoma breast cancer, etc	estrone
3	202436_s_at	CYP1B1	89.7%	2q22.2	monooxygenase oxidoreductase	congenital glaucoma breast cancer, etc	estrone
4	209331_s_at	MAX	87.9%	14q23	activator repressor	lung cancer prostate cancer, etc	leucine lysine
5	219563_at	LINC00341	87.9%	14q32.13	RNA Gene		
6	201387_s_at	UCHL1	87.9%	4p13	hydrolase ligase protease	Parkinson's disease trauma leukemia,etc	dopamine lysine tyrosine,etc
7	205064_at	SPRR1B	86.2%	1q21-q22	protein binding	ichthyosis lung carcinoma skin disease, etc	retinoic acid proline diethylstilbestrol,etc
8	211220_s_at	HSF2	86.2%	6q22	activator	carcinoma leukemia,etc	mg 132 leucine,etc
9	205513_at	TCN1	84.5%	11q11-q12	cobalamin binding	lymphoma oral cancer anemia,etc	cobalamin cobalt methylmalonic acid,etc
10	202254_at	SIPA1L1	70.7%	14q24.1	GTPase activation	papilloma	
11	218980_at	FHOD3	56.9%	18q12	protein coding	leukemia,etc	
12	217997_at	PHLDA1	6.9%	12q15	phospholipid binding	breast cancer melanoma,etc	
13	823_at	CX3CL1	6.9%	16q13	chemokine activity	leukemia,etc	tarc,etc
14	211998_at	H3F3A	5.2%	1q42.12	DNA binding	leukemia, etc	
15	211998_at	H3F3B	5.2%	17q25.1	DNA binding	lupus, etc	
16	202341_s_at	TRIM2	3.4%	4q31.3	ion binding	schizophrenia,etc	
17	204427_s_at	TMED2	1.7%	12q24.31	protein binding	pancreatitis	
18	204427_s_at	RRAS2	1.7%	11p15.2	GTP binding	carcinoma,etc	tetrapeptide, etc
19	213069_at	HEG1	1.7%	3q21.2	ion binding	malformation	
20	213989_x_at	SETD4	1.7%	21q22.13	protein binding	down syndrome	
21	215253_s_at	RCAN1	1.7%	21q22.1-q22.2	DNA binding	down syndrome leukemia,etc	ts 16 tacrolimus, etc
22	213069_at	DENND2A	1.7%	7q34	Rab guanyl-nucleotide exchange		

6.2 Application to the Glue Grant Study

The Inflammation and the Host Response to Injury is a large-scale collaborative research program supported by the National Institute of General Medical Sciences which began in 1998. It aims to better understand the human body's response to serious injury using a discovery-driven approach. A large amount of biomedical, genomic, and proteomic data are being successively collected for analysis. One of the missions of this research program is to provide improved understanding of novel genomic technologies applied to translational medicine as well as to clinical practice. For example, one goal is to identify gene sets having high predictability of multiple organ failure, which would be of tremendous value. As part of this program, one study focused on the body's response to burn injury. More detailed information regarding this program can be found at `www.gluegrant.org`.

The original multi-center study recruited 2002 burn injury patients and samples from different tissues were hybridized on either high-throughput microarrays or were sequenced for genomic research purposes. We demonstrate the validity of our proposed model using 869 buffy coat samples hybridized to Affymetrix Human Genome U133 Plus 2.0 Arrays and normalized and summarized using the dChip method [Li and Wong, 2001]. After removing observations with missing outcomes, 657 samples from 169 burn injury patients were used in our analysis. Originally, for each sample, there are 54675 probe sets for each sample. We further reduced the dimensionality by and filtering probe sets according to the Affymetrix Detection Call algorithm which provides an assessment of the reliability of each transcript. After filtering probe sets that are Absent on all arrays, there were 48093 probe sets remaining for statistical analysis. The severity of illness of the burn injury patients was assessed using the Marshall Multiple Organ Dysfunction Score [Marshall, 1995], which is considered to be a comprehensive and effective measurement system for critical illness condition and has been demonstrated to be strongly associated with risk of ICU and hospital mortality. The Marshall Score evaluates the dysfunction level of six systems: 1) the respiratory system (Po2/FIO2 ratio); 2) the renal system (serum creatinine concentration); 3) the hepatic system (serum bilirubin concentration); 4) the hematologic system (platelet count); 5) the central nervous system (Glasgow Coma Scale) and 6) the cardiovascular system (pressure-adjusted heart rate). The assessment of each organ results in an ordinal outcome ranging from 0 to 4 with 0 indicating normal and 4 indicating severe dysfunction. In addition, an aggregated Marshall score was derived by adding the individual organ scores. In our analysis, we modified the ordinal outcome by combining some categories together so that the observations in each category are more balanced. We applied the proposed model to classify the severity of illness according to all seven Marshall score measurements. The results from the Marshall score assessed on the renal system, central nervous system, and the aggregated Marshall score are mainly discussed.

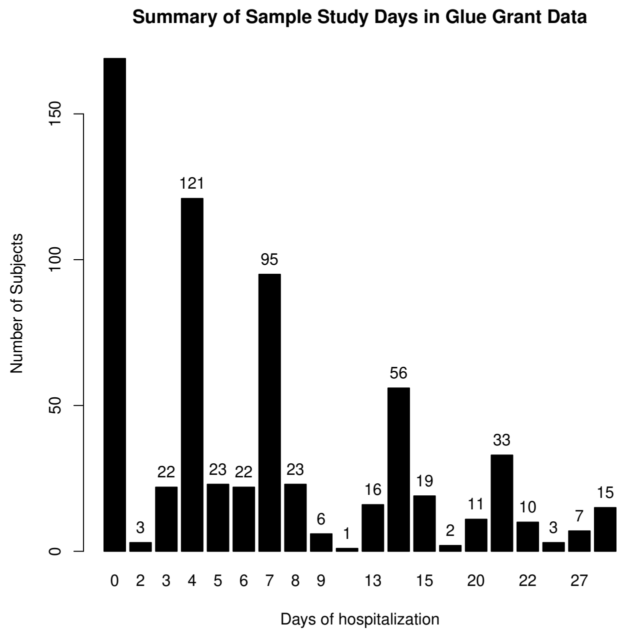
According to [Ibrahim et al., 2013], acute kidney injury (AKI) is known to be a major complication leading to mortality in burn injury patients. However, the treatment for this condition is still not well defined. Therefore, early diagnosis and prevention are of ultimate importance for preventing aggressive progression that incurs irreversible tissue damage. In other aspects, mental health and quality of life of severe burn injury patients are also of great concern. There is growing evidence that psychological health problems during the acute care setting have long-term consequences and influence outcome of burn injury [Renneberg et al., 2013]. Post-traumatic stress disorder (PTSD) is a common mental disorder that has been seen in up to 43% of burn injury patients 1 or more years after hospitalization [McKibben et al., 2008]. This aftermath psychological depression, if not identified and treated properly at early stages, could markedly change the survivors' quality of life. Besides these two specific system outcomes, the aggregated Marshall score also provides an insightful overall assessment of critical illness which is of tremendous value in making treatment decision.

The original time covariate in the data was recorded on the hour scale; we converted it to a day scale for convenience. Figure 6.2 illustrates the distribution of the sample collection times where the majority of samples were collected at four scheduled time points: baseline, day 4, day 7 and day 14. However, apparently a large number of samples were collected between the scheduled times which implied a very complex correlation structure among repeated measurements. In our analysis, the time covariate was standardized to have unit norm to correspond with the gene expression features which were also standardized. The set of important features were selected according to the AIC model fitting criteria. All analyses were conducted using the R programming environment 3.0.0.

Table 6.3: The original and modified Marshall score for the renal system

Original	Normal(0)	Mild(1)	Moderate(2)	Markedly(3)	Severe(4)
	236	395	17	4	5
Modified	Normal(0)	Mild(1)	Moderate+(2+)		
	236	395	26		

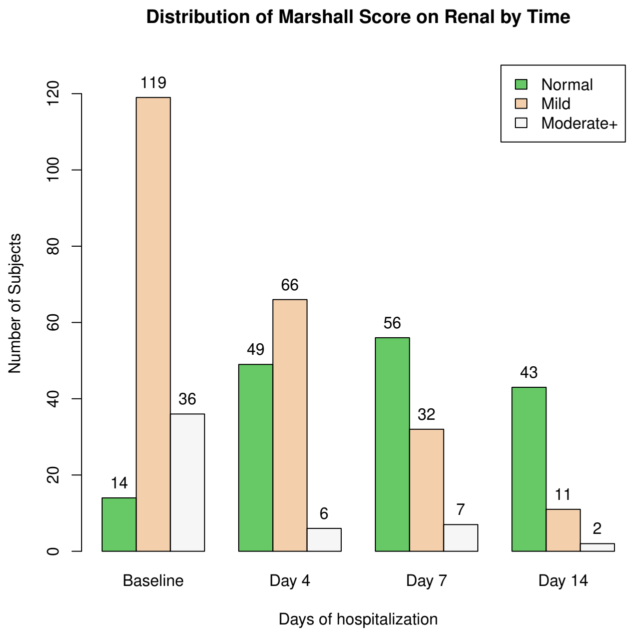
Figure 6.2: Counts of the days when the buffy coat samples were collected and hybridized to Affymetrix HG U133 Plus 2.0 array from 169 burn injury patients during their hospitalized days.



6.2.1 Marshall score for the renal system

We first present the results when applying the proposed model to the longitudinal high-dimensional burn injury data using the renal system Marshall score as our ordinal outcome. All features, including the time covariate, were standardized to have a mean of 0 and unit variance. The original Marshall score varied from 0 to 4 with the number of observations in each category displayed in the first row of Table 6.3. We combined the Moderate, Markedly, and Severe illness groups to create a modified three-category ordinal outcome as shown in the second row of Table 6.3. Figure 6.3 presents the distribution of the three-category renal system Marshall score evaluated at four time points: baseline, day 4, day 7 and day 14,

Figure 6.3: Distribution of three-category Marshall score assessed on the renal system at four time points: baseline, day 4, day 7 and day 14.



where an apparent decreasing trend is observed for groups with mild and more severe kidney dysfunction. In addition, an increase in the number of patients with normal kidney function (Marshall score of 0) indicates the effectiveness of treatment. We applied the GMIFS algorithm to the Glue Grant data setting the three parameters of increment amount to $\epsilon = 1 \times 10^{-4}$, convergence criteria to $\delta = 1 \times 10^{-4}$ and the maximal proportion of important probe sets detected to be 0.0015 (which corresponds to 72 probe sets inclusion of the total 48093). Upon convergence, 72 probe sets with nonzero coefficients entered the active set. We then updated the $\boldsymbol{\alpha}$ estimates at those steps immediately preceding the step where a new feature entered the active set of predictors. The model with optimal fitting criteria was then selected. In this scenario, the best parsimonious model associated with the minimal AIC was at step 19235 where the active set consisted of 34 probe sets with non-zero coefficient estimates. Figure 6.4 presents this selection process: on the left panel, the model fitting criteria AIC reaches its minimum at step 19235. On the right panel, each colored line represents one probe set having nonzero coefficient and the dashed vertical line represents the cutoff at the 19235th step. The probe sets associated with the colored line left of the dashed vertical line are considered to be useful for classification. Because the burn injury data is a longitudinal high-dimensional data set where the within-subject correlation is not negligible, a random coefficient/intercept ordinal response model was further fit using the 34 probe sets having non-zero coefficient estimates to update intercept estimate $\hat{\boldsymbol{\alpha}}$ and model fitting criteria again. After obtaining the updated estimates of the intercepts and the empirical Bayes estimate of the random effects, the fitted ordinal response can be calculated correspondingly. Tables 6.4 and 6.5 present the contingency table for the observed and predicted renal system Marshall score from the random intercept and random coefficient ordinal model, respectively. From both tables, a high classification accuracy (88.1% and 95.3%, respectively) was obtained. It is not surprising that the random coefficient model had better classification performance than the random intercept model, as it takes both the heterogeneity at baseline and variations in time trend into consideration. However, the computational time required for fitting a random coefficient model can be much longer than for fitting a random intercept model, especially when the data are relatively dense and the number of features having a nonzero coefficient is large. It is also of interest to compare the prediction accuracy when treating the data as longitudinal versus traditional. Table 6.6 presents the contingency table of the observed and predicted renal system Marshall score when ignoring the correlation between repeated measurements. The prediction accuracy drops to 80.7% and no observation in the extreme category is correctly classified.

To further validate the model performance and assess whether the result can be generalized to an independent data set, we also performed the cross-validation on the burn injury data. In each split, the test data consists of five subjects so that the size of test dataset varied as the number of repeated measurements per subject differs. In addition, because the number of observations in the 'moderate and beyond' group is significantly smaller than the other groups, to ensure the expected prediction error from the test data is not misleading due to lack of observations from the extreme category in the training data, we omitted all splits from the cross-validation procedure where the training data only contained observations from the normal and mildly ill groups. By implementing the proposed model on the training dataset and using the corresponding parameter estimates for calculating the predicted ordinal response for the full dataset, for the penalized random intercept ordinal response model, a high prediction accuracy with median 88.3% and range from 87.7% to 88.9%; and for the penalized random coefficient ordinal response model, a higher prediction accuracy with median 95.3% and range from 95.1% to 95.4% were obtained, which implied a consistently commendable performance of the proposed model on an independent dataset. A high Goodman and Kruskal' gamma 0.9851 [0.9828-0.9877] for random intercept model and 0.9984 [0.9983-0.9985] for random coefficient model were also observed indicating strong ordering similarities between the observed and predicted ordinal outcomes.

We presented the probe set having a non-zero coefficient in at least one of the cross-validation fits in a descending order of CV% in Table 6.7. CV% represents the percentage of times a probe set was identified by GMIFS in the cross-validation models. Five genes: HLA-DMB, DDAH2, BTN3A1, FUT3 and SOX13 were robustly identified by GMIFS in all partitions of cross-validation. Figure 6.5 illustrate the gene expression change of the first four genes along with days of hospitalization in three ordered renal impairment groups. For specific gene, the dot represents the observed gene expression from one microarray sample and the solid curve represents the average gene expression level in an ordered category fitted by locally weighted scatterplot smoothing (LOESS) technique [Cleveland, 1979]. By visualizing these graphs, three ordered groups can be clearly distinguished by the average gene expression level where the corresponding solid curves are approximately parallel and monotonically associated with the ordinal scale. The two characteristics depict the specific gene a good classifier. Also from these graphs, a larger bandwidth (gray shadow) was observed in the extreme category due to a significantly smaller number of observations in that category yields a much larger variation. Among the robust good classifiers, some have been previously reported to be associated with hypertension, *e.g.,* DDAH2, SOX13, which is important as hypertension is considered to be a major cause of kidney disease. A large amount of research has concentrated on understanding the biological mechanism behind hypertension. For example, Pullamsetti et al. [2005] investigated the role of the metabolizing enzyme DDAH in the course of Idiopathic pulmonary arterial hypertension (IPAH). Two isoforms of DDAH (DDAH1 and DDAH2) have been found in mammals. When comparing the tissue samples from healthy donors and IPAH patients, a significant reduction of DDAH2 immunoreactivity was observed while no significant difference in DDAH1 immunostaining intensity, which demonstrated the change in expression of DDAH in IPAH lungs.

Figure 6.4: The model fitting paths (left) and the regularization profile (right) plots for the Glue Grant burn injury data. In each plot, the horizontal-axis is the steps the forward stagewise algorithm has undertaken and the vertical-axes are the model fitting criteria AIC (left) and the penalized estimate for the coefficients (right).

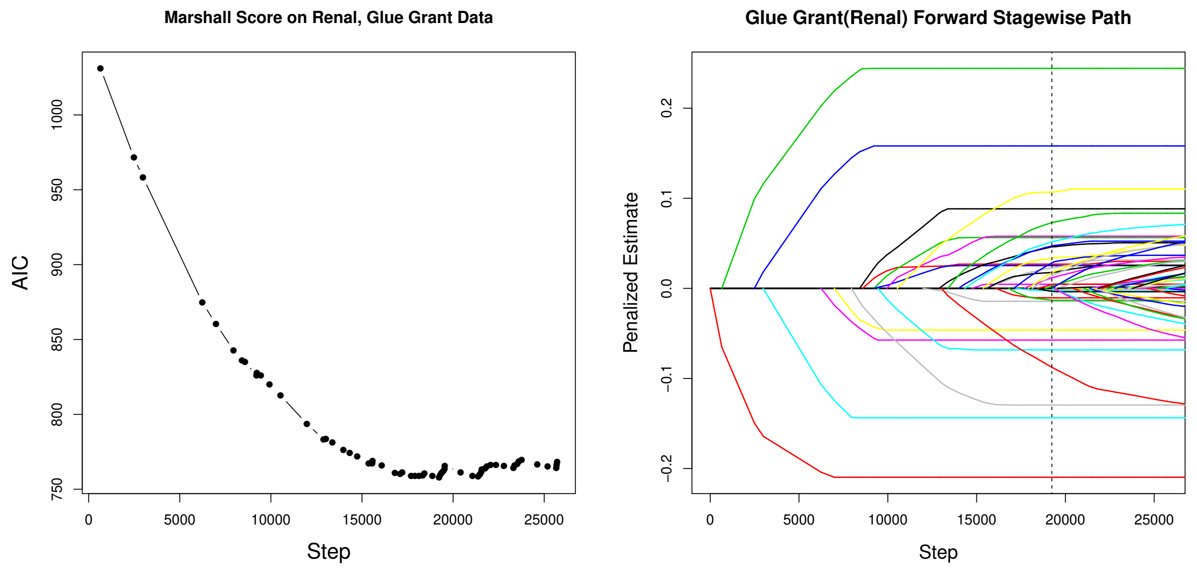


Table 6.4: Contingency table of the observed and predicted three-category Marshall score assessed on the renal system. The predicted ordinal outcome is calculated from penalized random intercept ordinal response model.

Pred/Obs	Normal(0)	Mild(1)	Moderate+(2+)
Normal(0)	196	14	0
Mild(1)	40	377	20
Moderate+(2+)	0	4	6

Table 6.5: Contingency table of the observed and predicted three-category Marshall score assessed on the renal system. The predicted ordinal outcome is calculated from penalized random coefficient ordinal response model.

Pred/Obs	Normal(0)	Mild(1)	Moderate+(2+)
Normal(0)	222	5	0
Mild(1)	14	389	11
Moderate+(2+)	0	1	15

Table 6.6: Contingency table of the observed and predicted three-category Marshall score assessed on renal system when ignoring the correlation between repeated measurements from the same subject.

Pred/Obs	Normal(0)	Mild(1)	Moderate+(2+)
Normal(0)	166	31	1
Mild(1)	70	364	25
Moderate+(2+)	0	0	0

Table 6.7: A subset of Affymetrix probe sets having a non-zero coefficient estimate from the GMIFS applied to the Glue Grant burn injury data when using the modified Marshall score assessed on the renal system as ordinal outcome. The probe set having a non-zero coefficient is ordered in a descending order of CV% where CV% represents the percentage of times a probe set was identified by GMIFS in the cross-validation models. The test dataset in each cross-validation model includes five subjects. Only the probe set can be mapped to a Gene Symbol is listed here and the full list of probe sets identified can be found in Appendix H.

No.	**AFFYprobe**	**SYMBOL**	**CV%**	**Est**	**Location**	**Type**	**Disease**	**Drugs**
1	203932_at	HLA-DMB	100%	0.244	6q21.3		lupus diabetes arthritis, etc	oligonucleotide
2	209770_at	BTN3A1	100%	0.158	6q22.1	T cell receptor	schizophrenia ovarian cancer, etc	
3	212033_at	RBM25	100%	0.058	14q24.3	mRNA binding	Alzheimer's disease thyroiditis	
4	214088_s_at	FUT3	100%	0.158	19p13.3	protein-binding	cancer cystic fibrosis amnesia	GDP-L-fucose tetrasaccharide glycolipid, etc
5	214909_s_at	DDAH2	100%	-0.210	6q21.3	amino acid binding	lupus hypertension kidney disease, etc	Citrulline Dimethylamine adma, etc
6	38918_at	SOX13	100%	0.073	1q32	DNA binding	cirrhosis hypertension neuronitis,etc	leucine
7	209514_s_at	RAB27A	97.0%	-0.057	15q15-q21.1	GDP/GTP binding	Griscelli syndrome albinism,etc	gtp gdp,etc
8	214025_at	DDX28	97.0%	-0.046	16q22.1	ATP binding	prostatitis	
9	39817_s_at	C6orf108	97.0%	0.047	6p21.1	protein-binding		
10	218191_s_at	LMBRD1	93.9%	0.028	6q13	cobalamin binding	anemia hepatitis metabolic disorders	
11	216336_x_at	MT1E	81.8%	0.025	16q13	ion binding	cancer carcinoma, etc	cadmium glyceraldehyde
12	207539_s_at	IL4	69.7%	0.014	5q31.1	growth factor activity	lymphoma asthma leukemia, etc	ionomycin il 10 rantes, etc
13	204970_s_at	MAFG	63.2%	-0.014	17q25.3	DNA binding	fibrosarcoma lung cancer	tbhq leucine
14	201968_s_at	PGM1	51.5%	-0.002	1p22.1	ion binding	liver cirrhosis tuberculosis, etc	D-Glucose Magnesium
15	220924_s_at	SLC38A2	51.5%	-0.003	12q	protein binding	neuronitis, etc	glutamate, etc
16	209762_x_at	SP110	36.4%	0.004	2q37.1	DNA binding	tuberculosis leukemia	

6.2.2 Marshall score for the central nervous system

We now present the results when applying the proposed model to the burn injury data when using Marshall score on the central nervous system as the ordinal outcome. We modified the original five-category ordinal scale to a more balanced three-category ordinal scale as shown in Table 6.8. We applied the forward stagewise algorithm to the full dataset setting the parameters to increment amount to $\epsilon = 1 \times 10^{-4}$, convergence criteria to $\delta = 1 \times 10^{-4}$, and the maximal proportion of important probe sets detected to 0.0015. Upon convergence, 72 probe sets had nonzero coefficients. We then fitted 72 penalized cumulative logit ordinal models at each of the turning point where a new feature enters into the active set in the following one step to update the estimate of $\boldsymbol{\alpha}$ and model fitting criteria. The model with optimal fitting criteria was then selected. In this case, the best parsimonious model associated with minimal AIC was at step 27128 where the active set consisted of 64 probe sets with non-zero coefficient estimates. We then used the coefficient estimates of the 64 probe sets to fit a penalized random coefficient/intercept ordinal response model for classification purpose. After obtaining the updated estimates of the intercepts and the empirical Bayes estimates of the random effects, the fitted ordinal outcome was calculated and the contingency table of the observed and predicted ordinal outcomes was constructed correspondingly. Classification accuracy of 71.7% and 92.1% were obtained for the penalized random intercept (Table 6.9) and penalized random coefficient (Table 6.10) models, respectively. The classification accuracy from the penalized ordinal random intercept model was similar to that from the traditional ordinal model when ignoring the intra-subject correlation (70.5% as shown in Table 6.11). However, when taking the time trend into consideration and quantify the correlation between measurements, the accuracy in classification was tremendously improved where the mental disorder severity of only a few observations were misclassified.

Table 6.8: The original and modified central nervous system Marshall score.

Original	Normal(0)	Mild(1)	Moderate(2)	Markedly(3)	Severe(4)
	210	48	125	84	190
Modified	Normal/ Mild(0,1)	Moderate/ Markedly(2,3)	Severe(4)		
	258	209	190		

A subset of 64 probe sets having a non-zero coefficient included in the penalized random coefficient/intercept ordinal response model were listed in Table 6.12, where each probe annotation was mapped to the corresponding gene symbol. There are several genes that have been reported to be associated with mental disorders and psychosocial impairment, such as Alzheimer's disease, schizophrenia, and neuritis. For example, RGS10 protein is known to have a predominant location in the cytosol [Rivero et al., 2010] and its movement between the cytoplasm and the nucleus is considered as a possible mechanism of regulating intra-cellular signaling [Burgon et al., 2001]. Rivero et al. [2013] designed an experiment to evaluate the treatment effect of an antipsychotic or antidepressant on schizophrenic, non-diagnosed suicide, and control groups where cytosolic RGS10 protein immunoreactivity has been involved in. However, no significant difference in RGS10 protein expression level was detected which implies new treatment and further studies are needed to elucidate its function. The reelin signaling is involved in the etiology of neurodevelopmental and psychiatric disorders such as: schizophrenia, bipolar, depression and autism. One of the two reelin receptors in the pathway is related to LRP8 gene [Fatemi, 2001]. Sequeira et al. [2012] reviewed nine most recent microarray studies of peripheral blood gene expression in schizophrenia where little agreement was reached in terms of specific genes identified enriched biological pathways. This could be due to different experimental designs and preparations, heterogeneity in subjects, large proportion of noise and high false positive rate. Ingenuity Pathway Analysis (IPA) was run on the gene symbols from the nine peripheral blood microarray experiments. However, the comprehensive understanding of the mechanism and biological function of most mental disorder still remains vague. To acquire a solid and precise understanding requires advanced knowledge in systems biology, a collection of a larger number of samples for examining multiple brain regions, and more stable methods such as RNA-sequencing.

Table 6.9: Contingency table of the observed and predicted three-category Marshall score assessed on central nervous system. The predicted ordinal outcome was calculated from the penalized random intercept ordinal response model.

	Mild(0,1)	Moderate(2,3)	Severe(4)
Mild(0,1)	188	37	5
Moderate(2,3)	70	144	46
Severe(4)	0	28	139

Table 6.10: Contingency table of the observed and predicted three-category Marshall score assessed on central nervous system. The predicted ordinal outcome was calculated from the penalized random coefficient ordinal response model.

	Mild(0,1)	Moderate(2,3)	Severe(4)
Mild(0,1)	236	5	0
Moderate(2,3)	22	188	9
Severe(4)	0	16	181

Table 6.11: Contingency table of the observed and predicted three-category Marshall score assessed on central nervous system when ignoring the correlation between repeated measurements from the same subject.

	Mild(0,1)	Moderate(2,3)	Severe(4)
Mild(0,1)	210	75	4
Moderate(2,3)	40	108	41
Severe(4)	8	26	145

Figure 6.5: Average gene expression levels in three ordinal categories change along with patient days in hospital for four genes: HLA-DMB (top left), DDAH2(top right), BTN3A1(bottom left) and FUT3(bottom right). These genes are 100% identified by GMIFS in cross-validation assessment procedure. For a specific gene, the dot represents the observed gene expression from one microarray platform and the solid curve represents the average gene expression level in an ordered category fitted by LOESS smoothing technique.

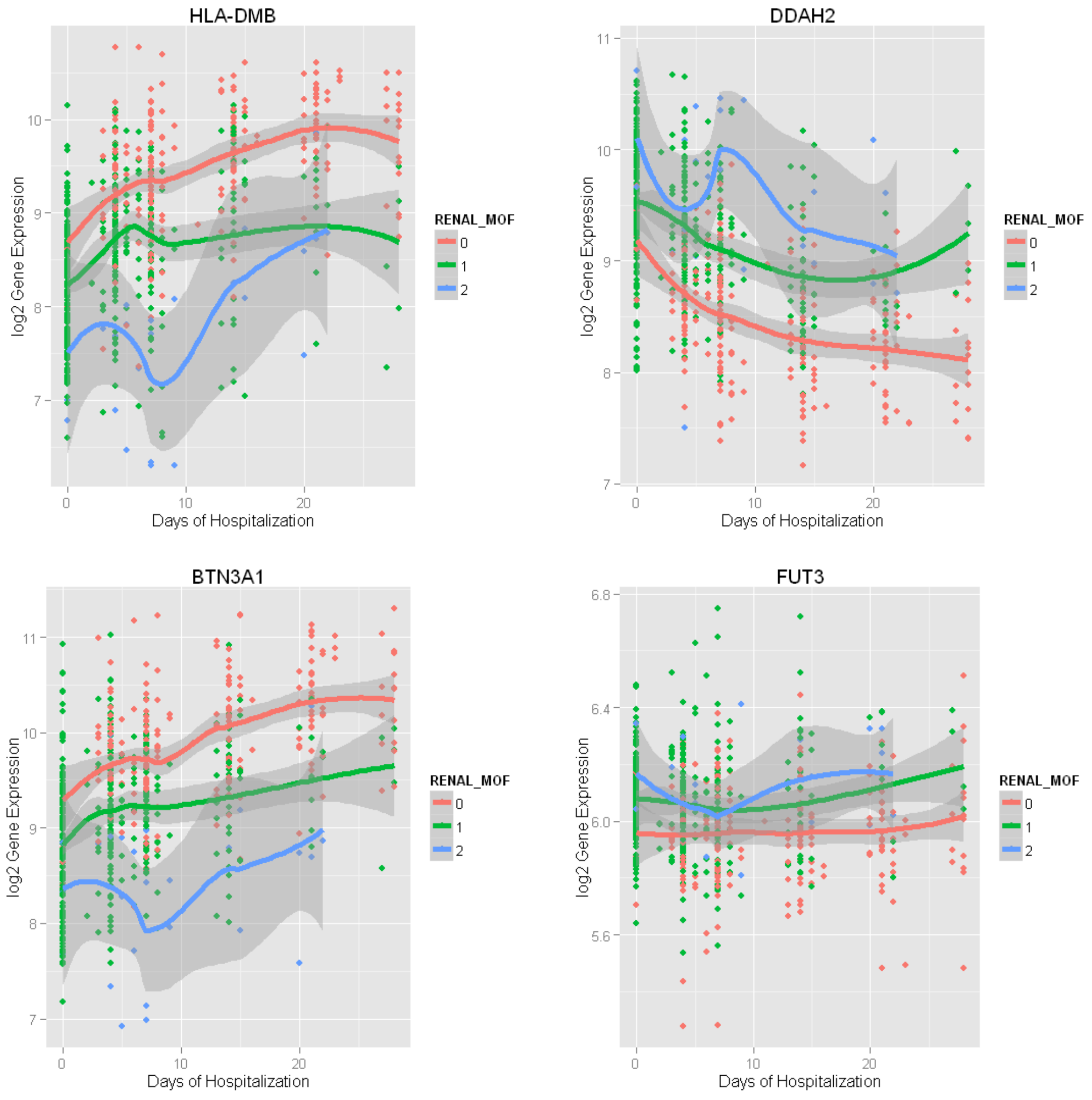


Table 6.12: A subsetof Affymetrix probe sets having a non-zero coefficient estimate in the GMIFS from Glue Grant burn injury dataset when using three-category central nervous system Marshall score as ordinal outcome mapped to Gene Symbol.

No.	**AFFYprobe**	**SYMBOL**	**Est**	**Location**	**Type**	**Disease**	**Drugs**
1	207564_x_at	OGT	0.16	Xq13	enzyme activator	Alzheimer diabetes,etc	Lactosamine threonine,etc
2	214909_s_at	DDAH2	-0.14	6p21	amino acid binding	hypertension kidney disease,etc	Citrulline nitric oxide, etc
3	218078_s_at	ZDHHC3	-0.12	3p21.31	ion binding	ataxia	
4	203633_at	CPT1A	0.08	11q13.2	protein binding	diabetes metabolic disorder	Coenzyme A Glycerol, etc
5	204316_at	RGS10	0.08	10q25	GTPase activator	nervosa schizophrenia neuronitis	gnrh cysteine
6	204970_s_at	MAFG	-0.06	17q25.3	DNA binding	fibrosarcoma	
7	205686_s_at	CD86	0.06	3q21	coreceptor	immunodeficiency neuritis, etc	Abatacept ctla4-ig, etc
8	205517_at	GATA4	-0.06	8p23.1-p22	DNA binding,etc	lymphoma heart disease cancer, etc	azathioprine zinc, etc
9	205425_at	HIP1	-0.06	7q11.23	clathrin binding	Huntington's disease neuropathy, etc	inositol glutamine,etc
10	208433_s_at	LRP8	-0.05	1p32.3	ion binding, etc	depression neuronitis, etc	tyrosine phosphotyrosine, etc
11	206110_at	HIST1H3H	0.05	6p22.1	DNA binding	erythematosus	
12	219452_at	DPEP2	0.05	16q22.1			Leukotriene E4
13	206114_at	EPHA4	-0.05	2q36.1	ATP binding	Alzheimer neuronitis, etc	tyrosine ADP, etc
14	209553_at	VPS8	0.04	3q27.2	zinc ion binding		
15	209553_at	LOC100505729	0.04	-	RNA gene		
16	211998_at	H3F3A	0.04	1q42.12	DNA binding	lupus immunodeficiency, etc	
17	211998_at	H3F3B	0.04	17q25.1	DNA binding	lupus immunodeficiency, etc	
18	201978_s_at	KIAA0141	0.04	5q31	protein coding	thyroiditis	
19	205781_at	C16orf7	-0.04	16q24.3	GTPase activator		
20	211743_s_at	PRG2	-0.03	11q12	carbohydrate binding	cancer rhinitis, etc	estromustine estracyt, etc
21	203932_at	HLA-DMB	0.03	6q21.3		lupus diabetes arthritis, etc	oligonucleotide
22	220614_s_at	C6orf103	-0.02	6q24.2	ion binding, etc	leprosy	
23	60474_at	FERMT1	0.02	20p12.3	phospholipid binding	periodontitis, etc	
24	213624_at	SMPDL3A	-0.02	6q22.32	protein binding	histiocytoma, etc	
25	206705_at	TULP1	-0.02	6p21.3	protein binding	neuronitis, etc	
26	202381_at	ADAM9	-0.02	8p11.23	collagen binding, etc	myeloma cancer, etc	Clotrimazole
27	222287_at	TRDN	0.02	6q22.31	protein binding	bipolar, etc	ryanodine, etc
28	201810_s_at	LOC100505696	-0.02	-	-	-	-
29	201810_s_at	SH3BP5	-0.02	3p24.3	kinase inhibitor	schizophrenia,etc	tyrosine, etc
30	202673_at	DPM1	0.01	20q13.1	alcohol binding, etc	tuberculosis alcoholism	Dolichol-20, etc
31	200899_s_at	MGEA5	0.01	10q24.1-3	protein binding	diabetes meningioma	
32	202925_s_at	PLAGL2	-0.01	20q11.21	DNA binding, etc	anemia, etc	
33	203389_at	KIF3C	-0.01	2p23	ATP binding	schizophrenia	
34	204611_s_at	PPP2R5B	-0.01	11q12	protein binding	neuronitis, etc	
35	205098_at	CCR1	-0.01	3p21	protein binding	hematopoiesis leukemia, etc	J 113863 bx471, etc
36	214469_at	HIST1H2AE	0.0033	6p22.1	DNA binding	immunodeficiency	
37	214469_at	HIST1H2AB	0.0017	6p22.1	DNA binding	immunodeficiency	
38	213298_at	NFIC	-0.0013	19p13.3	DNA binding	neuronitis, etc	diamide

6.2.3 Aggregated Marshall score

Besides using the assessments on a specific biological system to classify the severity of illness, it is also valuable to get an overall assessment of the medical condition using the aggregated Marshall score. Theoretically, the aggregated Marshall score ranges from 0 to 24 because on each of the six system scores can take on values from 0 to 4. However, it is rare for a patient to be evaluated as 'markedly' or 'severely' ill in all aspects. In fact, the aggregated Marshall score observed in the burn injury dataset varies from 0 to 15 with a heavy concentration of values falling between 4 and 8. In order to use the proposed method and treat the outcome as an ordinal scale, we compressed the original aggregated Marshall score (Figure 6.6) to a modified measurement with three categories as shown in Table 6.13.

We the applied the forward stagewise algorithm to the full dataset setting the parameters increment amount to $\epsilon = 1 \times 10^{-4}$, convergence criteria to $\delta = 1 \times 10^{-4}$ and the maximal proportion of important probes detected to be 0.0015. Upon convergence, 72 probes had non-zero coefficient entered the active set. We then fitted 72 penalized cumulative logit ordinal models at each of the turning point where a new feature enters into the active set in the following one step. In this case, the best parsimonious model associated with minimal AIC was at step 16813 where the active set consisted of 35 probe sets with non-zero coefficient estimates. We then used the coefficient estimates of the 35 probe sets to fit a penalized random coefficient/intercept ordinal response model for classification purpose. After obtaining updated estimates of the intercepts and the empirical Bayes estimates of the random effects, the fitted ordinal scale can be calculated correspondingly. High concordance between the observed and fitted response was observed with the classification accuracies of 85.5% and

87.2% for the penalized random intercept and random coefficient ordinal response models, respectively (Tables 6.14 and 6.15). An improvement in classification accuracy was also observed when treating the data as longitudinal compared to treating it as traditional, where the classification accuracy dropped to 72.8% as shown in Table 6.16.

A subset of the 35 probe sets having a non-zero coefficient included in the penalized random coefficient/intercept ordinal response model were presented in Table 6.18, where each probe annotation was mapped to the corresponding gene symbol. It is of interest to evaluate the concordance between the set of genes included in the aggregated Marshall score model and the set of genes included in the specific Marshall score models. Three genes: HLA-DMB, MAFG and DDAH2 were detected to be the robust classifiers under all three scenarios (Table 6.17).

Table 6.13: Modified three-category aggregated Marshall score

Categories	Mild(0-4)	Moderate(5-9)	Markedly(10-15)
No.obs	287	298	72

Table 6.14: Contingency table of the observed and predicted three-category aggregated Marshall Score. The predicted ordinal outcome is calculated from penalized random intercept ordinal response model.

Pred/Obs	Mild(0-4)	Moderate(5-9)	Severe(10-15)
Mild(0-4)	261	26	2
Moderate(5-9)	26	265	34
Severe(10-15)	0	7	36

Figure 6.6: Distribution of raw aggregated Marshall Score (observed range from 0 to 15) in burn injury dataset.

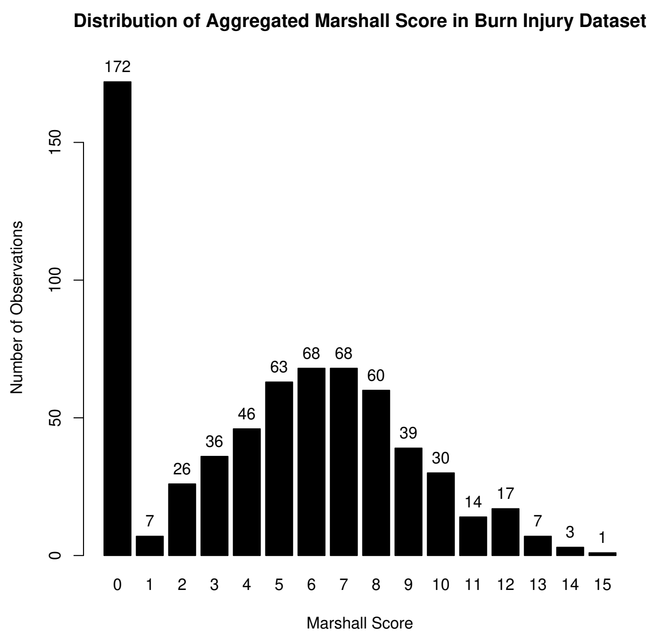


Table 6.15: Contingency table of the observed and predicted three-category aggregated Marshall Score. The predicted ordinal outcome is calculated from penalized random coefficient ordinal response model.

Pred/Obs	Mild(0-4)	Moderate(5-9)	Severe(10-15)
Mild(0-4)	254	13	0
Moderate(5-9)	33	280	33
Severe(10-15)	0	5	39

Table 6.16: Contingency table of the observed and predicted three-category aggregated Marshall score when ignoring the correlation between repeated measurements from the same subject.

Pred/Obs	Mild(0-4)	Moderate(5-9)	Severe(10-15)
Mild(0-4)	227	53	1
Moderate(5-9)	60	244	64
Severe(10-15)	0	1	7

Table 6.17: List of overlapping genes detected among the three classifications: the modified Marshall score on the renal system, the modified Marshall score on the central nervous system and the modified aggregated Marshall score.

Groups	Number of Probes	Gene Symbol
Renal vs Neuro	5	HLA-DMB, MAFG, DDAH2
Renal vs Aggregated	11	HLA-DMB, MAFG, RAB27A BTN3A1, RBM25, DDX28 DDAH2, SLC38A2
Neuro vs Aggregated	13	ADAM9, HLA-DMB, MAFG HIST1H3H, SMPDL3A, DDAH2
Renal vs Neuro vs Aggregated	5	HLA-DMB, MAFG, DDAH2

Table 6.18: A subsetof Affymetrix probe sets having a non-zero coefficient estimate in the GMIFS from Glue Grant burn injury dataset when using three-category aggregated Marshall score as ordinal outcome mapped to Gene Symbol.

No.	**AFFYprobe**	**SYMBOL**	**Est**	**Location**	**Type**	**Disease**	**Drugs**
1	214909_s_at	DDAH2	-0.23	6p21	amino acid binding	hypertension kidney disease,etc	Citrulline nitric oxide, etc
2	203932_at	HLA-DMB	0.18	6q21.3		lupus diabetes arthritis, etc	oligonucleotide
3	202381_at	ADAM9	-0.08	8p11.23	collagen binding, etc	myeloma cancer, etc	Clotrimazole
4	213624_at	SMPDL3A	-0.07	6q22.32	protein binding	histiocytoma, etc	
5	212033_at	RBM25	0.05	14q24.3	mRNA binding	Alzheimer's disease thyroiditis	
6	209375_at	XPC	0.05	3p25	DNA binding	pigmentosum, etc	cisplatin, etc
7	204970_s_at	MAFG	-0.05	17q25.3	DNA binding	fibrosarcoma	
8	202973_x_at	FAM13A	0.05	4q22.1	GTPase activator	COPD, etc	
9	209514_s_at	RAB27A	-0.04	15q15-q21.1	GDP/GTP binding	Griscelli syndrome albinism,etc	gtp gdp,etc
10	220924_s_at	SLC38A2	-0.04	12q	protein binding	neuronitis, etc	glutamate, etc
11	214025_at	DDX28	-0.02	16q22.1	ATP binding	prostatitis	
12	218380_at	LOC728392	0.02	17p13.2	-	Parkinson's disease	
13	213199_at	C2CD3	0.01	11q13.4	protein binding	nephronophthisis	
14	206110_at	HIST1H3H	0.01	6p22.1	DNA binding	erythematosus	
15	219681_s_at	RAB11FIP1	0.01	8p11.22	protein binding	neuronitis, etc	
16	209770_at	BTN3A1	0.01	6q22.1	T cell receptor	schizophrenia ovarian cancer, etc	
17	202116_at	DPF2	0.01	11q13.1	ion binding	nephropathy, etc	
18	220112_at	ANKRD55	-0.00	5q11.2	-	arthritis, etc	

6.3 Discussion

In this chapter, we applied the proposed model to solve the challenging classification and prediction problem in the traditional and longitudinal high-dimensional data setting for a multi-category ordinal response. The proposed model combines the novel machine learning algorithm, generalized monotone incremental forward stagewise (GMIFS), and the classical generalized linear mixed model (GLMM) as a statistical model capable of classifying an ordinal outcome using high-dimensional data with a complex correlation structure. The proposed model has a promising performance given its high prediction accuracy, consistency of variable selection, and concordance of orderings demonstrated by different microarray datasets. Demonstrated by the cross-validation procedure, the proposed model performed well on the training dataset and can be generalized to an independent test dataset and still remains consistently superior performance. In addition, the set of important features detected by GMIFS shares a large proportion of overlapping for different combinations of training and test datasets.

From the examples presented, it is clear the proposed model works best if 1) the number of ordered categories is relatively small and the number of observations in each category is balanced; and 2) the number of features having a nonzero coefficient is small, that is, the degree of sparseness is high. In fact, some theoretical and simulation work has shown the ordered variable with 3 or 4 categories often carries the maximal information and if the number of categories goes beyond 5, it can be treated as continuous instead [Pasta, 2009]. In addition, [Friedman et al., 2004] discussed that the model with an L_1 penalty follows the 'bet on sparsity' principle which reflects a fact that it has a good performance in sparse data with a small to moderate size of important features. We conclude this principle also holds for the GMIFS method. When comparing the set of nonzero coefficients from the data which uses the Marshall score on the renal system as the ordinal outcome versus that which uses the Marshall score on the central nervous system, a much larger number of features are identified from the central nervous system than the renal system. It has been discussed in several publications, *e.g.,*Karssen et al. [2006], that a relatively large set of modest changes in gene expression rather than a small set of strong signals are often identified to incur mental disorders with high variability, complexity, and heterogeneity. Correspondingly, since the real change in expression levels are often obscured with extraneous source of variation even under a carefully planned experiment, the set of features having nonzero coefficients can be a mixed of features that are good classifiers as well as noise features, which influences the predictive power and explains why the model for Marshall score on the renal system achieves higher prediction accuracy than that on the central nervous system.

The other potential improvement for GMIFS is the convergence criteria we implemented as well as the model fitting criteria. Currently, the two convergence criteria we used are: 1) the difference between successive log-likelihoods is smaller than a given tolerance; 2) the proportion of features with nonzero coefficients reaches a pre-specified number. Criterion 1) can often be satisfied in high-dimensional data with independence assumption. However, criterion 1) is seldom satisfied in longitudinal data due to inter-subject heterogeneity and complicated intra-subject correlation, which have the potential to influence to the fixed effect estimate and we choose to put it aside when obtaining the penalized estimate. Instead, the longitudinal high-dimensional model is often converged according to criterion 2). Although the choice of cutoff value in criterion 2) is somewhat arbitrary, based on the ‘bet on sparsity’ principle, only a small or moderate sized feature set with nonzero coefficients should be in the final model, thus a large enough value can cover all the important features.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

Health status and disease-related ordinal measurements play an irreplaceable role for assessment of severity of illness in a hospital setting as well as in translational medical research. With the emergence of novel genomic technologies being actively applied in diagnostic and therapeutic areas, a proper statistical methodology that is able to capture the strong signals from high-dimensional data for classifying and predicting an ordinal outcome is a great need. In addition, repeated measurements are common in clinical practice for tracking and monitoring the progression of disease, therefore we also developed a statistical model that is capable of analyzing correlated longitudinal high-dimensional data with an ordinal outcome. In Chapter 1, we reviewed the classical statistical model for analyzing data with an ordinal response in the traditional setting where enough degrees of freedom can be allocated for parameter estimation. Primarily, we reviewed four types of ordinal models: cumulative logit, adjacent category, backward continuation ratio, and forward continuation ratio models under the proportional odds assumption. The parameter estimates via the Maximum Likelihood approach as well as the corresponding software implementations were also included. In Chapter 2, we reviewed regularization methods that are often implemented to use a few important features to predict outcome in a high-dimensional dataset, where the number of features is much larger than the number of samples. Three prevalent methods: least absolute shrinkage and selection operator (LASSO), generalized monotone incremental forward stagewise (GMIFS), and least angle regression (LAR) were reviewed when interest lies in predicting a continuous or dichotomous response. In Chapter 3, we reviewed the statistical models to analyze a broad range of longitudinal data with different distributions in the responses. Namely, we reviewed the linear mixed models (LMM), nonlinear mixed model (NLMM), and generalized linear mixed model (GLMM) where the response has a normal, non-normal, or discrete distribution, respectively. The model fitting techniques for each type of model were discussed in detail afterwards. Specifically, for the linear mixed models, the Expectation-Maximization (EM) algorithm was implemented to obtain the estimate of the variance component and both Maximum Likelihood (ML) and Restricted Maximum Likelihood (REML) approaches were used to obtain the estimates of the fixed and random effects; for the nonlinear mixed models, the marginal likelihood and its numerical approximation form was optimized to obtain estimates of the fixed effects and variance components. The random effects were estimated using the Empirical Bayes approach; for the generalized linear mixed models, generalized estimating equations (GEE) and penalized quasi-likelihood (PQL) were briefly reviewed to obtain biased estimates of the fixed effects and the correlation structure. In Chapter 4, we introduced the ordinal random intercept and random coefficient models which were suitable for analyzing longitudinal data with an ordinal response. The models were fitted using the marginal likelihood approach whose closed-form was approximated by both the nonadaptive and adaptive Gauss-Hermite Quadrature methods. The fixed effects were obtained by optimizing the marginal likelihood and the random effects were obtained through empirical Bayes method. In Chapter 5, we first stated the question of interest, that is, to identify a subset of important features that is monotonically associated with the ordinal response that can be utilized to build a parsimonious model for predication and classification in a high-dimensional or longitudinal high-dimensional setting. We extended the GMIFS method for a binary outcome to solve the prediction and classification problem for high-dimensional data with an ordinal outcome. In addition, we combined the novel GMIFS method with the classical ordinal random coefficient model to create an innovative penalized random coefficient/intercept ordinal response model for solving the challenging classification and prediction problem for the longitudinal high-dimensional setting. The model assessment and selection criteria for the proposed models were also discussed. In Chapter 6, we applied the proposed model to two real microarray datasets to demonstrate its usage and effectiveness in analyzing this novel type of data. The model performance was assessed using cross-validation to estimate primarily three characteristics: prediction accuracy, consistency of variable selection, and similarity of ordering between the observed and predicted ordinal outcomes. In the COPD study, a high-dimensional dataset consisting of 58 samples, 54657 features, and four ordered response categories, our method detected several genes, *e.g.,* CYP1B1, SPRR1B, which are known to be associated with progression of lung cancer, inflammation and other diseases. In the burn injury study, 657 longitudinal high-dimensional microarray samples were collected from 169 patients with the comprehensive assessment of the severity of illness using the Marshall Multiple Organ Dysfunction

Score on six organs. The results presented primarily focused on using the Marshall score assessed on the renal and central nervous system as the ordinal outcomes. One set of genes, *e.g.,* DDAH2, SOX13, have been reported in literature to be associated with hypertension and kidney disease was also detected by our method. In addition, another set of gene, *e.g.,* RGS10 and LRP8 which have been previously associated with mental disorders were also detected by our proposed method when using the Marshall score on the central nervous system as the ordinal outcomes. In both scenarios, the set of genes detected serves as good classifiers to classify and predict the progression of disease severity. In fact, a very high prediction accuracy of 95.3% was observed from the cross-validation procedure when using the Marshall score on the renal system as the outcome.

7.2 Future Work

7.2.1 Variable Selection using LAR type Algorithm

The proposed model incorporates the GMIFS method for selecting important features which are monotonically associated with the ordinal scale. For each step, the coefficient associated with the largest negative gradient of the likelihood is updated with a very small incremental amount. Thus, this procedure usually takes hundreds of thousands iterations before reaching convergence criteria. Efron et al. [2004] introduced the Least Angle Regression (LAR) method for high-dimensional data where the response is continuous, which has been considered as a 'democratic' version of the Forward Stagewise algorithm. LAR is a less greedy version of forward stagewise and Efron et al. [2004] showed with a small modification on LAR, the penalized estimates from LAR and forward stagewise agree in general. The other striking property of LAR is it only requires p steps for the full solution, where p is the number of features. This property of LAR is appealing since it can tremendously reduce the computational time. Madigan and Ridgeway [2004] briefly discussed the possibility of extending the LAR algorithm to generalized linear models and illustrated using logistic regression. Suppose π_i is the probability observation i falls into category 0 and y_i is the corresponding indicator, the log-likelihood of N observations having a binary outcome can be written as:

$$\log L = \sum_i^N y_i \log \frac{\pi_i}{1-\pi_i} - \log(1-\pi_i). \tag{7.2.1}$$

Using the logit link for π_i, the linear expression $\alpha + \mathbf{x_i^T}\boldsymbol{\beta}$ demonstrates the log-likelihood is a function of α and $\boldsymbol{\beta}$, which can be denoted as $\log L(\alpha, \beta; \mathbf{x})$. Similar to the GMIFS method, the LAR algorithm starts with all coefficients $\boldsymbol{\beta} = 0$. At the first step, $\beta^*_{j_1}$ associated with the maximum gradient of $-\log L(\alpha, \boldsymbol{\beta}; \mathbf{x})$ is selected to update and thus j_1^* is a member belonging to the active set, that is $j_1^* \in A$.

$$\begin{aligned} j_1^* &= \arg\max_j \left| - \frac{\partial \log L(\alpha, \boldsymbol{\beta}; \mathbf{x})}{\partial \beta_j} \right| = \arg\max_j \big| - \mathbf{x_j^T} \sum_i (y_i - \pi_i) \big| \\ &= \arg\max_j \left| - \mathbf{x_j^T} \sum_i (y_i - \frac{\exp(\alpha + \mathbf{x_i^T}\boldsymbol{\beta})}{1 + \exp(\alpha + \mathbf{x_i^T}\boldsymbol{\beta})} \right| \end{aligned} \tag{7.2.2}$$

In the following iterations, suppose β_{j_2} is the coefficient associated with the second largest gradient and $j_2 \in A^c$. An inequality constraint (7.2.3), where s_j indicates the sign of the first-order derivative in the LAR development, is required to hold until the absolute magnitude of gradient in direction $\mathbf{x_{j_2}}$ exceeds that in direction $\mathbf{x_{j_1^*}}$, and consequently j_2 enters the active set A, that is $j_2^* \in A$.

$$\begin{aligned}
&\left| -\frac{\partial \log L(\alpha, \boldsymbol{\beta}; \mathbf{x})}{\partial \beta^*_{j_1}} \right| - \left| -\frac{\partial \log L(\alpha, \boldsymbol{\beta}; \mathbf{x})}{\partial \beta_{j_2}} \right| \geq 0 \\
&(s_{j_1^*}\mathbf{x_{j_1^*}} - s_{j_2}\mathbf{x_{j_2}})^T \left(\sum_i (y_i - \frac{\exp(\alpha + \mathbf{x_i^T}\boldsymbol{\beta})}{1 + \exp(\alpha + \mathbf{x_i^T}\boldsymbol{\beta})} \right) \geq 0. \qquad (7.2.3)
\end{aligned}$$

Now the active set A contains two elements j_1^* and j_2^* which construct a new direction between $\mathbf{x_{j_1^*}}$ and $\mathbf{x_{j_2^*}}$. Suppose β_{j_3} is the coefficient associated with the third largest gradient and $j_3 \in A^c$, the inequality (7.2.4) needs to be held until the absolute magnitude of the gradient in direction $\mathbf{x_{j_3}}$ exceeds those in directions $\mathbf{x_{j_1^*}}$ and $\mathbf{x_{j_2^*}}$,

$$\min_{j_1^*, j_2^*} \left(\left| -\frac{\partial \log L(\alpha, \boldsymbol{\beta}; \mathbf{x})}{\partial \beta^*_{j_1}} \right|, \left| -\frac{\partial \log L(\alpha, \boldsymbol{\beta}; \mathbf{x})}{\partial \beta^*_{j_2}} \right| \right) - \left| -\frac{\partial \log L(\alpha, \boldsymbol{\beta}; \mathbf{x})}{\partial \beta_{j_3}} \right| \geq 0. \qquad (7.2.4)$$

Similar logic applies to the following iterations which yields the full solution. The extension to the ordinal model should be followed accordingly.

7.2.2 Variable Selection with Consideration of the Correlations between Features

In either GMIFS or LAR methods, a strong assumption, which in fact is seldom mentioned, is the independence between features. In practice, this assumption may never be held for high-dimensional data and for the genomic data, particularly. It is well known a group of genes often works together to complete one or several complex biological processes. Pathway analysis is an approach to identify groups of related genes according to their underlying molecular functions. It has been applied to the analysis of Gene Ontology terms (`http://www.geneontology.org/`) for finding physical interaction networks between genes and gene products. It also helps to exploit pathway knowledge in the public repository Kyoto Encyclopedia of Genes and Genomes (KEGG) (`http://www.genome.jp/kegg/`), which records networks of molecular interactions in cells and their variants specific to particular organisms. A brief review of current approaches and outstanding challenges of pathway analysis can be found in Khatri et al. [2012].

To select a set of correlated features using a penalized model, modifications to existing methods are essential. In fact, Yuan and Lin [2006] introduced three modified models: group LASSO, group LAR, and group non-negative garrottee to address the correlation issue when the response is continuous. We briefly explain the mechanism of group LAR here. Suppose a vector r is of length n, where n is the number of observations in sample and p is the number of features denoted by $\mathbf{x_1}, \cdots, \mathbf{x_p}$. The angle $\theta(r, \mathbf{x_j})$ is determined by vector r and feature $\mathbf{x_j}$. It follows $\cos^2\theta(r, \mathbf{x_j}) = \|\mathbf{x_j^T} r\|^2/\|r\|^2$ is the proportion of the total variation in direction r explained in direction $\mathbf{x_j}$. The group LAR starts with all coefficients $\boldsymbol{\beta} = (\beta_1, \cdots, \beta_p) = \mathbf{0}$. It first finds the feature $\mathbf{x_j}$ that has the smallest angle θ with response Y, that is equivalent to say, direction $\mathbf{x_j}$ explains the largest proportion of variation in the response Y at the current iteration. The group LAR proceeds in the direction of Y until another feature $\mathbf{x_2}$ has the same angle with the current residual r, that is, $\|\mathbf{x_1^T} r\| = \|\mathbf{x_2^T} r\|$. The group LAR marches in the direction of current residual r determined by $\mathbf{x_1}$ and $\mathbf{x_2}$, until the third feature $\mathbf{x_3}$ enters the active set where the angle between $\mathbf{x_3}$ and r is the same as $\theta(r, \mathbf{x_1})$ and $\theta(r, \mathbf{x_2})$. The group LAR now moves in a direction of the updated residual r, which is determined by $\mathbf{x_1}, \mathbf{x_2}$ and $\mathbf{x_3}$, until the fourth feature $\mathbf{x_4}$ enters the active set and so on and so forth. Extending the group LAR to a generalized linear model, or in particular the ordinal model may need further modification, since the residual is not a good measurement in data having discrete response.

7.2.3 Application to Other Genomic and Medical Data

In this dissertation, we applied the proposed model to microarray dataset for detecting important genes associated with the progression of disease. The usage of our proposed model certainly should not be limited to microarray data, other types of genomic data, such as SNP or high-throughput sequencing data could also be analyzed using our proposed model provided the response is ordinal. A single-nucleotide polymorphism (SNP) refers to a variation in a single DNA base. Such variations have been identified as causal for certain diseases and thus are critically important for personalized medicine in biomedical research. Genome-wide association studies (GWAS), are conducted to examine common genetic variants among individuals to determine if certain variants are associated with a disease. The proposed model can be applied to this type of data for feature selection as well as for trait classification purposes. Another popular technology for measuring gene expression is next-generation sequencing. This novel technology provides a solution for low-cost sequencing that parallelizes the sequencing process and results in millions of sequences reads for a single sample. Although a huge amount of data processing still remains an ineluctable challenge even with the rapid development of high-performance computing capacity, gene expression studies have been moving gradually from microarray technologies to RNA sequencing as it promises higher resolution, lower biases, and ability to discover novel transcripts and mutations [Soon et al., 2013]. In fact, some striking advancements in personalized medicine include monitoring certain types of diseases are using high-throughput methods. For example Chen et al. [2012] presented an integrative personal omics profile (iPOP), which performs a comprehensive analysis of longitudinal omic information (combination of genomic, transcriptomic, proteomic, metabolomic and autoantibody) from a single individual over 14 months. RNA-seq samples at 20 time points were generated resulting over 2.67 billion uniquely mapped 101b paired-end reads along with other approaches. By analyzing this wealth of omics information to search for correlated patterns over time and single unusual events can reveal multiple medical risks, including type 2 diabetes and coronary artery disease in this individual. Our proposed model has the potential to analyze a much larger longitudinal study for obtaining more general conclusions.

In addition, the proposed model has the potential to provide tremendous benefits to personalized medicine from non-genomic data, such as using large-scale medical record databases to predict health outcomes. A revealing example was provided by [McCormick et al., 2011] where a hierarchical model was proposed from the Bayesian perspective to mine the associations between past, current, and future events. A simple example is that a patient experiencing dyspepsia and epigastirc pains is likely reporting heartburn in future doctor visits. Because most patients experience only a few among a massive set of possible symptoms and patients often visit the doctors' office only periodically, a longitudinal sparse dataset results which remains a challenge for existing statistical models. However, our proposed model may be a possible solution to predict disease progression using large-scale medical record data, which often has a longitudinal structure.

Bibliography

M. Abramowitz and I.A. Stegun. *Handbook of Mathematical Functions.* Dover Publications, Inc, New York, NY, 1972.

A. Agresti. *Analysis of Ordinal Categorical Data.* John Wiley & Sons, Inc, Hoboken, NJ, 2 edition, 2010.

H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19:716–723, 1974.

D.B. Rubin A.P. Dempster and R.K. Tsutakawa. Estimation in covariance components models. *Journal of the American Statistical Association*, 76:341–353, 1981.

D. Bates. Computational methods for mixed models. Available at **http://cran.r-project.org/web/packages/lme4/vignettes/Theory.pdf**, 2011.

R.D. Bock. *Multivariate Statistical Methods in Behavioral Research.* McGraw-Hill, New York, NY, 1975.

H.D. Bondell, A. Krishna, and S.K. Ghosh. Joint variable selection for fixed and random effects in linear mixed-effects models. *Biometrics*, 66:1069–1077, 2010.

N.E. Breslow and D.G. Clayton. Approximate inference in generalized linear mixed models. *Journal of the American Statistical Association*, 88:9–25, 1993.

H. Brown and R. Prescott. *Applied Mixed Models in Medicine.* John Wiley & Sons, Ltd, West Sussex, England, 2nd edition, 2006a.

H. Brown and R. Prescott. *Applied Mixed Models in Medicine.* John Wiley & Sons, Inc., West Sussex, England, second edition, 2006b.

CG. Broyden. The convergence of a class of double-rank minimization algorithms. *Journal of the Institute of Mathematics and its Applications*, 6:76–90, 1970.

P. Burgon, W. Lee, A. Nixon, E. Peralta, and P. Casey. Phosphorylation and nuclear translocation of a regulator of g protein signaling (rgs10). *Journal of Biological Chemistry*, 276(35):32828–32834, 2001.

R. Chen, G.I. Mias, J. Li-Pook-Than, L. Jiang, H.Y.K. Lam, R. Chen, et al. Personal omics profiling reveals dynamic molecular and medical phenotypes. *Cell*, 148(6):1293–1307, 2012.

Z. Chen and D.B. Dunson. Random effects selection in linear mixed models. *Biometrics*, 59: 762–769, 2003.

R.H.B Christensen. ordinal: Regression models for ordinal data. Available at **http://cran.r-project.org/web/packages/ordinal/index.html**, 2012.

W.S. Cleveland. Robust locally weighted regression and smoothing scatterplots. *Journal of the American statistical association*, 74(368):829–836, 1979.

M. Maechler D. Bates and B. Bolker. lme4: Linear mixed-effects models using s4 classes. Available at **http://cran.r-project.org/web/packages/lme4/index.html**, 2011.

M. Davidian and A.R. Gallant. The nonlinear mixed effects model with a smooth random effects density. *Biometrika*, 80:475–888, 1993.

P.J. Davis and P. Rabinowitz. *Methods of Numerical Integration.* Academic Press, New York, NY, 1975.

A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39: 1–38, 1977.

JE. Dennis and HHW. Mei. Two new unconstrained optimization algorithms which use function and gradient values. *Journal of optimization theory and applications*, 28:1453–482, 1979.

P.J. Diggle, K.Y. Liang, and S.L. Zeger. *Analysis of Longitudinal Data.* Oxford University Press, Oxford, UK, 1994.

M. Chen D.K. Dey and H. Chang. Bayesian approach for nonlinear random effects models. *Biometrics*, 53:1239–1252, 1997.

D. Donoho and I. Johnstone. Adapting to unknown smoothness via wavelet shrinkage. *Journal of American Statistical Association*, 90:1200–1224, 1995.

N.R. Draper and H. Smith. *Applied Regression Analysis.* John Wiley & Sons, Inc., New York, NY, second edition, 1981.

R. Hall E. Berndt, B. Hall and J. Hausman. Estimation and inference in nonlinear structural models. *Annals of Economic and Social Measurement*, 3:653–665, 1974.

B. Efron and R. Tibshirani. Improvements on cross-validation: The 632+ bootstrap method. *Journal of the American Statistical Association*, 92:548–560, 1997.

B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *Annals of Statistics*, 32:407–499, 2004.

F. Ezzet and J. Whitehead. A random effects model for ordinal responses from a crossover trial. *Statistics in Medicine*, 10:901–907, 1991.

J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of American Statistical Association*, 96:1348–1360, 2001.

S. Fatemi. Reelin mutations in mouse and man: from reeler mouse to schizophrenia, mood disorders, autism and lissencephaly. *Molecular Psychiatry*, 6:129–133, 2001.

R.A. Fisher. The correlation between relatives on the supposition of mendelian inheritance. *Transactions of the Royal Society of Edinburgh*, 52:399–433, 1918.

R. Fletcher. A new approach to variable metric algorithms. *The computer journal*, 13: 317–322, 1970.

R. Fletcher and CM. Reeves. Function minimization by conjugate gradients. *Computer Journal*, 7:148–154, 1964.

J. Friedman, T. Hastie, S. Rosset, R. Tibshirani, and J. Zhu. Discussion of boosting papers. *Annals of Statistics*, 32:102–107, 2004.

J. Friedman, T. Hastie, H. Hofling, and R. Tibshirani. Pathwise coordinate optimization. *The Annals of Applied Statistics*, 1:302–332, 2007.

J. Friedman, T. Hastie, and R. Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33:1–22, 2010.

J. Friedman, T. Hastie, and R. Tibshirani. glmnet: Lasso and elastic-net regularized generalized linear models. Available at **http://cran.r-project.org/web/packages/glmnet/glmnet.pdf**, 2013.

W. Fu. Penalized estimating equations. *Biometrics*, 59:126–132, 2003.

J. Geweke. Bayesian inference in econometric models using monte carlo integration. *Econometrica*, 57:1317–1339, 1989.

L.A. Goodman and W.H. Kruskal. Measures of association for cross classifications. *Journal of the American Statistical Association*, 49:732–764, 1954.

P.J. Green. Iteratively reweighted least squares for maximum likelihood estimation, and some robust and resistant alternatives. *Journal of Royal Statistical Society, Series B*, 46: 149–192, 1984.

P.J. Green. Penalized likelihood for general semi-parametric regression models. *International Statistical Review*, 55:245–259, 1987.

H.O. Hartley and J.N.K Rao. Maximum-likelihood estimation for the mixed analysis of variance model. *Biometrika*, 54:93–108, 1967.

D. Harville. Bayesian inference for variance components using only error contrasts. *Biometrika*, 61:383–385, 1974.

D. Harville. Maximum likelihood approaches to variance component estimation and to related problems. *Journal of the American statistical association*, 72:320–338, 1977.

D. Harville and R. Mee. A mixed-model procedure for analyzing ordered categorical data. *Biometrics*, 40:393–408, 1984.

T. Hastie, J. Taylor, R. Tibshirani, and G. Walther. Forward stagewise regression and the monotone lasso. *Electronic Journal of Statistics*, 2007:1–29, 2007.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference and Prediction.* Springer, New York, NY, 2nd edition, 2009.

D. Hedeker and R. Gibbons. *Longitudinal Data Analysis.* John Wiley & Sons, Inc, Hoboken, NJ, 2006.

D. Hedeker and R.D. Gibbons. A random-effects ordinal regression model for multilevel analysis. *Biometrics*, 50:933–944, 1994.

C. R. Henderson. *Applications of Linear Models in Animal Breeding.* Guelph, Canada, 1984.

C.R. Henderson, O. Kempthorne, S.R. Searle, and C.M. von Krosigk. The estimation of environmental and genetic trends from records subject to culling. *Biometrics*, 15:192–218, 1959.

L.H. Herbach. Properties of model ii-type analysis of variance tests, a: Optimum nature of the f-test for model ii in the balanced case. *Annals of Mathematical Statistics*, 30:939–959, 1959.

A. Hoerl and R. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12:55–67, 1970.

A.E. Ibrahim, K.A. Sarhane, S.P. Fagan, and J. Goverman. Renal dysfunction in burns: A review. Technical report, Massachusetts General Hospital, 2013.

J. Jansen. On the statistical analysis of ordinal data when extravariation is present. *Applied Statistics*, 39:75–84, 1990.

R.I. Jennrich and M.D. Schluchter. Unbalanced repeated measures models with structured covariance matrices. Technical report, University of California, Los Angeles, Department of Biomathematics, 1985.

R.A. Johnson and D.W. Wichern. *Applied Multivariate Statistical Analysis*. Pearson Eduction, Inc, Upper Saddle River, NJ, 6th edition, 2007.

A.M. Karssen, J.Z. Li, S. Her, P.D. Patel, F. Meng, S.J. Evans, M.P. Vawter, et al. Application of microarray technology in primate behavioral neuroscience research. *Methods*, 38 (3):227–234, 2006.

P. Khatri, M. Sirota, and A.J. Butte. Ten years of pathway analysis: current approaches and outstanding challenges. *PLoS computational biology*, 8(2):e1002375, 2012.

N.M. Laird and J.H. Ware. Random-effects models for longitudinal data. *Biometrics*, 38: 963–974, 1982.

N. Larid, N. Lange, and D. Stram. Maximum likelihood computations with repeated measures: Application of the em algorithm. *Journal of the American Statistical Association*, 83:97–105, 1987.

C. Leng, Y. Lin, and G. Wahba. A note on the lasso and related procedures in model selection. *Statistica Sinica*, 16:1273–1284, 2006.

C. Li and W. H. Wong. Model-based analysis of oligonucleotide arrays: Model validation, design issues and standard error application. *Genome Biology*, 2:1–11, 2001.

K.Y. Liang and SL. Zeger. Longitudinal data analysis using generalized linear models. *Biometrika*, 73:13–22, 1986.

M.J. Lindstrom and D.M. Bates. Nonlinear mixed effects models for repeated measures data. *Biometrics*, 46:673–687, 1990.

Q. Liu and D.A. Pierce. A note on gauss-hermite quadrature. *Biometrika*, 81:624–629, 1994.

M. Lorr and CJ. Klett. *Inpatient Multidimensional Psychiatric Scale*. Consulting Psychologists Press, Palo Alto, CA, 1966.

L.Tierney and J.B. Kadane. Accurate approximations for posterior moments and marginal densities. *Journal of the American Statistical Association*, 81:82–86, 1986.

D. Madigan and G. Ridgeway. Discussion of least angle regression. *Annals of Statistics*, 2004.

J.C. Marshall. Multiple organ dysfunction score: A reliable descriptor of a complex clinical outcome. *Critical Care Medicine*, 23:1638–1652, 1995.

A.S. Mayer and L.S. Newman. Genetic and environmental modulation of chronic obstructive pulmonary disease. *Respiration Physiology*, 128:3–11, 2001.

T. McCormick, C. Rudin, and D. Madigan. *A hierarchical model for association rule mining of sequential events: An approach to automated medical symptom prediction*, 2011.

P. McCullagh. Regression models for ordinal data(with discussion). *Journal of the Royal Statistical Society; Series B*, 42:109–142, 1980.

P. McCullagh and J.A. Nelder. *Generalized Linear Models*. Chapman and Hall, London, UK, 1989.

J. McKibben, M. Bresnick, S. Askay, and J. Fauerbach. Acute stress disorder and post-traumatic stress disorder: a prospective study of prevalence, course, and predictors in a sample with major burn injuries. *Journal of Burn Care & Research*, 29:22–35, 2008.

RH. Myers. *Classical and Modern Regression with Applications*. Duxbury Press, North Scituate, MA, 1990.

J Nash and R. Varadhan. optimx: A replacement and extension of the optim() function. Available at **http://cran.r-project.org/web/packages/optimx/index.html**, 2012.

JA. Nelder and R. Mead. A simplex method for function minimization. *The Computer Journal*, 7:308–313, 1965.

J.A. Nelder and R.W.M. Wedderburn. Generalized linear models. *Journal of the Royal Statistical Society. Series A*, 135:370–384, 1972.

R. Opgen-Rhein and K. Strimmer. longitudinal: Analysis of multiple time course data. Available at **http://cran.r-project.org/web/packages/longitudinal/longitudinal.pdf**, 2013.

W. Pan. Akaike's information criterion in generalized estimating equations. *Biometrics*, 57: 120–125, 2001.

D.J. Pasta. Learning when to be discrete: Continuous vs. categorical predictors. In *SAS Global Forum, Washington, DC*, 2009.

H.D. Patterson and R. Thompson. Recovery of inter-block information when block sizes are unequal. *Biometrika*, 58:545–554, 1971.

J. Peng, X. Xu, B.E. Mace, L.A. Vanderveer, L.R. Workman, M.J. Slifker, P.M. Sullivan, T.D. Veenstra, and M.L. Clapper. Estrogen metabolism within the lung and its modulation by tobacco smoke. *Carcinogenesis*, 34:909–915, 2013.

J. Pinheiro and D. Bates. Approximations to the log-likelihood function in the nonlinear mixed-effects model. *Journal of Computational and Graphical Statistics*, 4:12–35, 1995.

M.J.D. Powell. A survey of numerical methods for unconstrained optimization. *SIAM Review*, 12:79–97, 1970.

S. Pullamsetti, L. Kiss, H.A. Ghofrani, R. Voswinckel, P. Haredza, et al. Increased levels and reduced catabolism of asymmetric and symmetric dimethylarginines in pulmonary hypertension. *The FASEB journal*, 19(9):1175–1177, 2005.

C. Rangel, J. Angus, Z. Ghahramani, M. Lioumi, E. Sotheran, A. Gaiba, D.L. Wild, and F. Falciani. Modeling t-cell activation using gene expression profiling and state-space models. *Bioinformatics*, 20:1361–1372, 2004.

B. Renneberg, S. Ripper, J. Schulze, A. Seehausen, M. Weiler, G. Wind, B. Hartmann, G. Germann, and A. Liedl. Quality of life and predictors of long-term outcome after severe burn injury. *Journal of Behavioral Medicine*, pages 1–10, 2013.

J. Nocedal RH. Byrd, P. Lu and C. Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16:1190–1208, 1995.

G. Rivero, A. Gabilondo, J. García-Sevilla, R. La Harpe, B. Morentí, and J. Meana. Characterization of regulators of g-protein signaling rgs4 and rgs10 proteins in the postmortem human brain. *Neurochemistry International*, 57(7):722–729, 2010.

G. Rivero, A. Gabilondo, J. García-Sevilla, L. Callado, R. La Harpe, B. Morentin, and J. Meana. Brain rgs4 and rgs10 protein expression in schizophrenia and depression. effect of drug treatment. *Psychopharmacology*, 226:1–12, 2013.

G.K. Robinson. That blup is a good thing: The estimation of random effects. *Statistical Science*, 6:15–51, 1991.

RE. Schapire, Y. Freund, P. Bartlett, and W. Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. *Annals of Statistics*, 26:1651–1686, 1998.

G. E. Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6:461–464, 1978.

P. Sequeira, M. Martin, and M. Vawter. The first decade and beyond of transcriptional profiling in schizophrenia. *Neurobiology of Disease*, 45(1):23–36, 2012.

W.W. Soon, M. Hariharan, and M.P. Snyder. High-throughput sequencing for biology and medicine. *Molecular systems biology*, 9(1), 2013.

R. Stiratelli, N. Laird, and J.H. Ware. Random-effects models for serial observations with binary response. *Biometrics*, pages 961–971, 1984.

Y.C. Tai and T. Speed. A multivariate empirical bayes statistic for replicated microarray time course data. *Annals of Statistics*, 34:2387–2412, 2006.

W.A. Thompson. The problem of negative estimates of variance components. *Annals of Mathematical Statistics*, 33:273–289, 1962.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B*, 58:267–288, 1996.

T.W.Yee. Vgam: Vector generalized linear and additive models. Available at **http://cran.r-project.org/web/packages/VGAM/index.html**, 2013.

F. Vaida and S. Blanchard. Conditional akaike information for mixed-effects models. *Biometrika*, 92:351–370, 2005.

R. Varadhan. alabama: Constrained nonlinear optimization. Available at **http://cran.r-project.org/web/packages/alabama/index.html**, 2011.

SH. Walker and DB. Duncan. Estimation of the probability of an event as a function of several independent variables. *Biometrika*, 54:167–179, 1967.

R.W.M. Wedderburn. Quasi-likelihood functions, generalized linear models, and the gauss-newton method. *Biometrika*, 61:439–447, 1974.

B.J. Winer. *Statistical Principles in Experimental Design.* McGraw-Hill, New York, NY, 2nd edition, 1971.

M. Woenckhaus, L. Klein-Hitpass, U. Grepmeier, J. Merk, M. Pfeifer, P. Wild, M. Bettstetter, P. Wuensch, H. Blaszyk, A. Hartmann, F. Hofstaedter, and W. Dietmaier. Smoking and cancer-related gene expression in bronchial epithelium and non-small-cell lung cancers. *Journal of Pathology*, 210:192–204, 2006.

R. Wolfinger. Laplace's approximation for nonlinear mixed models. *Biometrika*, 80:791–795, 1993.

R. Wolfinger and M. O'connell. Generalized linear mixed models a pseudo-likelihood approach. *Journal of statistical Computation and Simulation*, 48(3-4):233–243, 1993.

TT. Wu, YF. Chen, T. Hastie, E. Sobel, and K. Lange. Genome-wide association analysis by lasso penalized logistic regression. *Bioinformatics*, 25:714–721, 2009.

M. Yang. Multinomial regression. In AH. Leyland and H. Goldstein, editors, *Multilevel Modeling of Health Sciences.* John Wiley & Sons, New York, NY, 2001.

M. Yuan and C. Kendziorski. Hidden markov models for microarray time course data under multiple biological conditions (with discussion). *Journal of the American Statistical Association*, 101:1323–1340, 2006.

M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67, 2006.

Y. Zhang, R. Tibshirani, and R. Davis. Classification of patients from time-course gene expression. *Biostatistics*, 14:87–98, 2013.

B. Zhou, W. Xu, D. Herndon, R. Tompkins, R. Davis, W. Xiao, and W. H. Wong. Analysis of factorial time-course microarrays with application to a clinical study of burn injury. *Proceedings of the National Academy of Sciences*, 107:9923–9928, 2010.

H. Zou. The adaptive lasso and its oracle properties. *Journal of the American Statistical Association*, 101:1418–1429, 2006.

H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B*, 67:301–320, 2005.

Appendices

Appendix A

NIMH Schizophrenia Data Code

A.1 R code for NIMH Schizophrenia Data

```
# Import the NIMH Schizophrenia Data and Preprocessing #
library(alabama)
NIMH<-read.csv('NIMH Schizophrenia.csv',header=TRUE)
attach(NIMH)
yi1<-ifelse(NIMH$imps79o==1,1,0)
yi2<-ifelse(NIMH$imps79o==2,1,0)
yi3<-ifelse(NIMH$imps79o==3,1,0)
yi4<-ifelse(NIMH$imps79o==4,1,0)
xi<-NIMH[,c('tx','sweek','txswk')]
G=function(z){
 G=exp(z)/(1+exp(z))
  return(G)
  }
g=function(z) {
 g=exp(z)/(1+exp(z))^2
  return(g)
  }
### Cumulative logit Ordinal Model ###
alpha<-vector(length=4,mode='numeric')
alpha[1]<-0
beta<-vector(length=3,mode='numeric')
par<-vector(length=6,mode='numeric')
logL.cum <-function(par){
           comp1<-comp2<-comp3<-comp4<-vector(length=dim(NIMH)[1],mode='numeric')
           z0<-z1<-z2<-z3<-z4<-vector(length=dim(NIMH)[1],mode='numeric')
           alpha<-par[1:3]; beta<-par[4:6]
for (ii in 1: dim(NIMH)[1]){
    z0[ii]<--Inf+sum(beta*xi[ii,])
    z1[ii]<-alpha[1]+sum(beta*xi[ii,])
    z2[ii]<-alpha[2]+sum(beta*xi[ii,])
    z3[ii]<-alpha[3]+sum(beta*xi[ii,])
    z4[ii]<-100+sum(beta*xi[ii,])
    }
    comp1<-yi1*log(G(z1)-G(z0))
    comp2<-yi2*log(G(z2)-G(z1))
    comp3<-yi3*log(G(z3)-G(z2))
    comp4<-yi4*log(G(z4)-G(z3))
    -sum(comp1+comp2+comp3+comp4)
 }
hin<-function(par){
alpha<-par[1:3]
h<-rep(NA,1)
h[1]<-alpha[2]-alpha[1]
h[2]<-alpha[3]-alpha[2]
h
}
hin.jac<-function(par){
alpha<-par[1:3]
j<-matrix(NA,2, length(par))
j[1,]<-c(1,0,1,0,0,0)
j[2,]<-c(0,1,1,0,0,0)
j
}
x0<-c(1,2,3)
fit.cum <- auglag(par=c(x0,c(0,0,0)),logL.cum, hin=hin,hin.jac=hin.jac)
### Adjacent-Category Ordinal Model ###
par<-vector(length=6,mode='numeric')
logL.acat<-function(par){
           comp1<-comp2<-comp3<-comp4<-vector(length=dim(NIMH)[1],mode='numeric')
           z1<-z2<-z3<-z4<-vector(length=dim(NIMH)[1],mode='numeric')
           alpha<-par[1:3];a1<-alpha[1];a2<-alpha[2];a3<-alpha[3]
           beta<-par[4:6]
for (ii in 1: dim(NIMH)[1]){
z1[ii]<-exp(-a1-a2-a3-3*sum(beta*xi[ii,]))/(1+exp(-a1-a2-a3-3*sum(beta*xi[ii,]))+
        exp(-a2-a3-2*sum(beta*xi[ii,]))+exp(-a3-sum(beta*xi[ii,])))
z2[ii]<-exp(-a2-a3-2*sum(beta*xi[ii,]))/(1+exp(-a1-a2-a3-3*sum(beta*xi[ii,]))+
        exp(-a2-a3-2*sum(beta*xi[ii,]))+exp(-a3-sum(beta*xi[ii,])))
z3[ii]<-exp(-a3-sum(beta*xi[ii,]))/(1+exp(-a1-a2-a3-3*sum(beta*xi[ii,]))+
        exp(-a2-a3-2*sum(beta*xi[ii,]))+exp(-a3-sum(beta*xi[ii,])))
z4[ii]<-1/(1+exp(-a1-a2-a3-3*sum(beta*xi[ii,]))+exp(-a2-a3-2*sum(beta*xi[ii,]))+
        exp(-a3-sum(beta*xi[ii,])))
}
    comp1<-yi1*log(z1)
    comp2<-yi2*log(z2)
    comp3<-yi3*log(z3)
    comp4<-yi4*log(z4)
    -sum(comp1+comp2+comp3+comp4)
}
    fit.acat<-nlm(logL.acat,c(0,0,0,0,0,0),hessian=T)
    se.acat <- sqrt(diag(solve(fit.acat$hessian)))
### Backward Continuation Ratio Model ###
par<-vector(length=6,mode='numeric')
```

```
logL.bwdcr<-function(par){
            comp1<-comp2<-comp3<-comp4<-vector(length=dim(NIMH)[1],mode='numeric')
            z1<-z2<-z3<-z4<-vector(length=dim(NIMH)[1],mode='numeric')
            alpha<-par[1:3];a2<-alpha[1];a3<-alpha[2];a4<-alpha[3]
            beta<-par[4:6]
 for (ii in 1: dim(NIMH)[1]){
    z4[ii]<-exp(a4+sum(beta*xi[ii,]))/(1+exp(a4+sum(beta*xi[ii,])))
    z3[ii]<-exp(a3+sum(beta*xi[ii,]))*(1-z4[ii])/(1+exp(a3+sum(beta*xi[ii,])))
    z2[ii]<-exp(a2+sum(beta*xi[ii,]))*(1-z4[ii]-z3[ii])/(1+exp(a2+sum(beta*xi[ii,])))
    z1[ii]<-1-z2[ii]-z3[ii]-z4[ii]
    }
    comp1<-yi1*log(z1)
    comp2<-yi2*log(z2)
    comp3<-yi3*log(z3)
    comp4<-yi4*log(z4)
    -sum(comp1+comp2+comp3+comp4)
 }
 fit.bwdcr<-nlm(logL.bwdcr,c(0,0,0,0,0,0),hessian=T)
 se.bwdcr <- sqrt(diag(solve(fit.bwdcr$hessian)))
### Forward Continuation Ratio Model ###
par<-vector(length=6,mode='numeric')
logL.fwdcr<-function(par){
            comp1<-comp2<-comp3<-comp4<-vector(length=dim(NIMH)[1],mode='numeric')
            z1<-z2<-z3<-z4<-vector(length=dim(NIMH)[1],mode='numeric')
            alpha<-par[1:3];a2<-alpha[1];a3<-alpha[2];a4<-alpha[3]
            beta<-par[4:6]
 for (ii in 1: dim(NIMH)[1]){
    z1[ii]<-exp(a2+sum(beta*xi[ii,]))/(1+exp(a2+sum(beta*xi[ii,])))
    z2[ii]<-exp(a3+sum(beta*xi[ii,]))*(1-z1[ii])/(1+exp(a3+sum(beta*xi[ii,])))
    z3[ii]<-exp(a4+sum(beta*xi[ii,]))*(1-z1[ii]-z2[ii])/(1+exp(a4+sum(beta*xi[ii,])))
    z4[ii]<-1-z1[ii]-z2[ii]-z3[ii]
    }
    comp1<-yi1*log(z1)
    comp2<-yi2*log(z2)
    comp3<-yi3*log(z3)
    comp4<-yi4*log(z4)
    -sum(comp1+comp2+comp3+comp4)
 }
   fit.fwdcr<-nlm(logL.fwdcr,c(0,0,0,0,0,0),hessian=T)
   se.fwrdcr<-sqrt(diag(solve(fit.fwdcr$hessian)))
```

A.2 R code for NIMH Schizophrenia Data using `VGAM` package

```
NIMH<-read.csv('NIMH Schizophrenia.csv',header=TRUE)
attach(NIMH)
library(VGAM)
optm.output <- function(fit.vglm) {
             output <- summary(fit.vglm)@coef3
             df <- dim(NIMH)[1]-6
             t.value <- output[,1]/output[,2]
             p.value <- rep(0,6)
             for (ii in 1:length(p.value)){
             p.value[ii] <- 2*pt(abs(t.value[ii]),  df, lower.tail=F)
             }
output1 <- data.frame(round(output[,1],2), round(output[,2],3),
                      round(rep(df,6),0), round(t.value,3), round(p.value,3))
return(output1)
}
### Cumulative logit Ordinal Model ###
fit.vglm<-vglm(imps79o~tx+sweek+tx*sweek, family=cumulative(parallel=T,reverse=F))
optm.output(fit.vglm)
### Adjacent-Categories ordinal model ###
fit.vglm<-vglm(imps79o~tx+sweek+tx*sweek, family=acat(parallel=T,reverse=F))
optm.output(fit.vglm)
### Backward Continuation Ratio Model ###
fit.vglm<-vglm(imps79o~tx+sweek+tx*sweek, family=sratio(parallel=T,reverse=T))
optm.output(fit.vglm)
### Forward Continuation Ratio model ###
fit.vglm<-vglm(imps79o~tx+sweek+tx*sweek, family=sratio(parallel=T,reverse=F ))
optm.output(fit.vglm)
```

A.3 SAS code for NIMH Schizophrenia Data

```
/*Ordinal Model with fixed effects*/;
DATA  one; INFILE 'V:\schizx1.dat';
INPUT id imps79 imps79b imps79o int tx week sweek txswk ;
run;
DATA one;
set one;
IF imps79 > -9;
run;
data three;
set one;
if imps79o=1 then do;
y1=1; y2=0; y3=0; y4=0;
end;
else if imps79o=2 then do;
y1=0; y2=1; y3=0; y4=0;
end;
else if imps79o=3 then do;
y1=0; y2=0; y3=1; y4=0;
end;
if imps79o=4 then do;
y1=0; y2=0; y3=0; y4=1;
end;
output;
run;
/*Cumulative logit Ordinal Model*/;
PROC GLIMMIX DATA=three METHOD=QUAD(QPOINTS=21) NOCLPRINT;
CLASS id;
MODEL imps79o = tx sweek txswk / SOLUTION DIST=MULTINOMIAL LINK=CUMLOGIT;
RUN;
/*Adjacent Category Ordinal Model*/;
proc nlmixed data=three method=Gauss tech=newrap qtol=1e-3;
parms alpha1=0, alpha2=0, alpha3=0,beta1=0,beta2=0,beta3=0;
/*linear predictor*/;
eta1=alpha1 + beta1*tx + beta2*sweek + beta3*txswk ;
eta2=alpha2 + beta1*tx + beta2*sweek + beta3*txswk ;
eta3=alpha3 + beta1*tx + beta2*sweek + beta3*txswk ;
pi1= 1/(exp(eta1+eta2+eta3)+exp(eta1+eta2) + exp(eta1) + 1);
pi2= exp(eta1)/(exp(eta1+eta2+eta3)+exp(eta1+eta2) + exp(eta1) + 1);
pi3= exp(eta1+eta2)/(exp(eta1+eta2+eta3)+exp(eta1+eta2) + exp(eta1) + 1);
pi4= exp(eta1+eta2+eta3) / (exp(eta1+eta2+eta3)+exp(eta1+eta2) + exp(eta1) + 1);
   z = (pi1**y1)*(pi2**y2)*(pi3**y3)*(pi4**y4);
   if (z > 1e-15) then ll = log(z);
   else ll=-1e100;
   model z~ general(ll);
run;
/*Backward Continuation Ratio Ordinal Model*/;
proc nlmixed data=three method=Gauss noad qtol=1e-4;
parms alpha2=0, alpha3=0, alpha4=0,beta1=0,beta2=0,beta3=0;
eta2 = alpha2 + beta1*tx + beta2*sweek + beta3*txswk ;
eta3 = alpha3 + beta1*tx + beta2*sweek + beta3*txswk ;
eta4 = alpha4 + beta1*tx + beta2*sweek + beta3*txswk ;
p4 = exp(eta4)/(1+exp(eta4));
p3 = (1-p4)*exp(eta3)/(1+exp(eta3));
```

```
p2 = (1-p3-p4)*exp(eta2)/(1+exp(eta2));
p1 = 1-p2-p3-p4;
/*Define likelihood*/
z = (p1**y1)*(p2**y2)*(p3**y3)*(p4**y4);
if (z > 1e-8) then ll = log(z);
else ll=-1e100;
model z~ general(ll);
run;
/*Forward Continuation Ratio Ordinal Model*/;
proc nlmixed data=three method=Gauss noad qtol=1e-4;
parms alpha1=0, alpha2=0, alpha3=0,beta1=0,beta2=0,beta3=0;
/*linear predictor*/;
eta1 = alpha1 + beta1*tx + beta2*sweek + beta3*txswk ;
eta2 = alpha2 + beta1*tx + beta2*sweek + beta3*txswk ;
eta3 = alpha3 + beta1*tx + beta2*sweek + beta3*txswk ;
p1 = exp(eta1)/(1+exp(eta1));
p2 = (1-p1)*exp(eta2)/(1+exp(eta2));
p3 = (1-p1-p2)*exp(eta3)/(1+exp(eta3));
p4 = 1-p1-p2-p3;
z = (p1**y1)*(p2**y2)*(p3**y3)*(p4**y4);
if (z > 1e-8) then ll = log(z);
else ll=-1e100;
model z~ General(ll);
run;
```

Appendix B

Orange Tree Example Code

B.1 R code for Orange Tree Example

```
# Implement EM algorithm to estimate the variance components #
library(MASS)
tree<-read.csv(file="tree.csv",header=TRUE)
attach(tree)
y<-tree$y
x<-tree$day
index<-tree$tree
n<-dim(tree)[1]
new.y<-y-parm[1]/(1+exp(-(x-parm[2])/parm[3]))
z<-matrix(rep(as.vector(diag(1,5)),rep(7,25)),35,5)
new.z<-z/(1+exp(-(x-parm[2])/parm[3]))
q1 <- nrow(new.z)
em.mixed <- function(y, x, z, var0, var1,maxiter=2000,tolerance = 1e-0010)
{
time <-proc.time()
n <- length(y)
conv <- 1
L0 <- loglike(y, x, new.z, var0, var1)
i<-0
cat(" Iter. sigma0 sigma1 Likelihood",fill=T)
repeat {
if(i>maxiter) {conv<-0
break}
V <- c(var1) * new.z %*% t(new.z) + c(var0) * diag(n)
Vinv <- solve(V)
resid <- new.y
temp1 <- Vinv %*% resid
s0 <- c(var0)^2 * t(temp1)%*%temp1 + c(var0) * n - c(var0)^2 * sum(diag(Vinv))
s1 <- c(var1)^2 * t(temp1)%*%z%*%t(z)%*%temp1+ c(var1)*q1 - c(var1)^2
*sum(diag(t(z)%*%Vinv%*%z))
var0 <- s0/n
var1 <- s1/q1
L1 <- loglike(y, x, new.z, var0, var1)
if(L1 < L0) { print("log-likelihood must increase, llikel <llike0, break.")
conv <- 0
break
}
i <- i + 1
 cat(" ", i," ",var0," ",var1," ",L1,fill=T)
if(abs(L1 - L0) < tolerance) {break} #check for convergence
L0 <- L1
}
list( var0=var0,var1=var1,Loglikelihood=L0)
}
loglike<- function(y, x, z, var0, var1)
{
n<- length(y)
V <- c(var1) * z %*% t(z) + c(var0) * diag(n)
Vinv <- ginv(V)
resid <- new.y
temp1 <- Vinv %*% resid
(-.5)*( log(det(V)) + t(resid) %*% temp1 )
# x<-b1/(1+exp(-(x-b2)/b3))
# (-.5)*( log(det(V))+ log (det(t(x)%*%Vinv%*%x))+ t(resid) %*% temp1 )
}
tolerance <- 1e-0010
maxiter <- 2000
seed<-100
# AGH numeric integration #
n1<-1
library(orthopolynom)
gh.poly<-ghermite.h.polynomials( n1, 0, normalized=FALSE)
node<-solve(gh.poly[[n1+1]])
weight <- ghermite.h.weight( node, 0 )
L1.star<-mu.star<-vector(length=7,mode='numeric')
temp<-b.out<-t<-vector(length=5,mode='numeric')
b.f<-function(parm){
for (kk in 1:5){
g1<-function(t) {
for (ii in 1:7){
mu.star[ii]<-(parm[1]+t )/(1+exp(-(x[ii+7*(kk-1)]-parm[2])/parm[3]))
L1.star[ii]<-(y[ii+7*(kk-1)]-mu.star[ii])^2
}
sum(L1.star)+(1/D.hat)*t^2
}
b.out[kk]<-nlm(g1,0)$estimate
}
return(b.out)
}
M<-5;N<-35;
temp1<-G<-vector(length=5,mode='numeric'); eval1<-matrix(nc=1,nr=N)
node1<-matrix(nc=n1,nr=N); L1<-mu<-matrix(nc=n1,nr=N)
L2<-L4<-matrix(nc=n1,nr=M)
c<-replicate(n1*5,1)
J.matrix<-matrix(c,nr=5,nc=n1)
f<-function(parm){
```

```
b<-b.f(parm)
for (kk in 1:5) {
for (ii in 1:7) {
b10<-parm[1] ; b20<-parm[2] ;b30<-parm[3]
fx<-deriv(y~(b1+t)/(1+exp(-(x-b2)/b3)),c("t"),
function(b1,b2,b3, x=day[ii+7*(kk-1)],t=b[kk]){} )
eval1[ii+7*(kk-1)]<-as.numeric(attributes(fx(b10,b20,b30))$gradient)
G[kk]<-sum(t(eval1[(1+7*(kk-1)):(7+7*(kk-1)),])%*%eval1[(1+7*(kk-1)):(7+7*(kk-1)),])+1/D.hat
    }
 }
for (kk in 1:5) {
for (ii in 1:7) {
for ( jj in 1:n1) {
 index<-ii+7*(kk-1)
 node1[(1+7*(kk-1)):(7+7*(kk-1)),jj]<-b[kk]+ sqrt(c(sigma2)*(1/G[kk]))*node[jj]
 mu[index,jj]<-(parm[1]+node1[index,jj] )/(1+exp(-(x[index]-parm[2])/parm[3]))
 L1[ii+7*(kk-1),jj]<- (y[ii+7*(kk-1)]-mu[ii+7*(kk-1),jj])^2
 tmp<-(b^2/D.hat**J.matrix)
 L2[kk,jj]<- sum(L1[(1+7*(kk-1)):(7+7*(kk-1)),jj])
 L3<-(L2+tmp)/(2*c(sigma2))
 L4[kk,jj]<-exp(-L3[kk,jj]+node[jj]^2/2)
              }
            }
          }
 tmp1<-L4*weight
 temp1<-apply(tmp1,1,sum)
 log(prod(temp1))-(35*log(2*pi*sigma2)+5*log(D.hat)+5*log(G[1]))/2
 }
# Incoporate AGH numeric intergration with EM algorithm #
j<-0
parm<-c(200,700,350)
repeat{
new.y<-y-parm[1]/(1+exp(-(x-parm[2])/parm[3]))
z<-matrix(rep(as.vector(diag(1,5)),rep(7,25)),35,5)
new.z<-z/(1+exp(-(x-parm[2])/parm[3]))
q1 <- nrow(new.z)
em.output<-em.mixed(y,x,new.z,1,1)
sigma2<-c(em.output$var0)
D.hat<-c(em.output$var1/em.output$var0)
AGH.output<-optim(parm,f,control=list(fnscale=-10, trace=5),hessian=TRUE)
if(abs(parm[1]-AGH.output$par[1])<0.0001) {break}
parm<-AGH.output$par
j<-j+1
cat(j,parm,sigma2,D.hat)
output<-list(sigma.e=sigma2, sigma.u=sigma2*D.hat, parm=parm)
}
output
```

B.2 R code for Orange Tree Example using `lme4` package

```
library(lme4)
tree<-read.csv(file="tree.csv", header=TRUE)
attach(tree)
circumference<-tree$y
age<-tree$day
tree<-tree$tree
# Laplacian Approximation #
(nm1 <- nlmer(circumference ~ SSlogis(age, Asym, xmid, scal) ~ Asym|Tree,
Orange, start = c(Asym = 200, xmid = 700, scal = 350), REML=F))
#Adaptive Gaussian-Hermite Quadrature Approximation with 5 points #
(nm1 <- nlmer(circumference ~ SSlogis(age, Asym, xmid, scal) ~ Asym|Tree,
Orange, start = c(Asym = 200, xmid = 700, scal = 350), nAGQ=5), REML=F))
```

B.3 SAS code Orange Tree Example

```
data tree;
input tree day y;
datalines;
1 118 30
1 484 58
1 664 87
1 1004 115
1 1231 120
1 1372 142
1 1582 145
2 118 33
2 484 69
2 664 111
2 1004 156
2 1231 172
2 1372 203
2 1582 203
3 118 30
3 484 51
3 664 75
3 1004 108
3 1231 115
3 1372 139
3 1582 140
4 118 32
4 484 62
4 664 112
4 1004 167
4 1231 179
4 1372 209
4 1582 214
5 118 30
5 484 49
5 664 81
5 1004 125
5 1231 142
5 1372 174
5 1582 177
run;
/*Adaptive Gaussian Quadrature N_{AGQ}=5*/;
/*Laplacian Approximation, equivalent to N_{AGQ)=1*/;
proc nlmixed data=tree tech=quanew update=bfgs qpoint=1;
parms b1=200 b2=700 b3=350 s2u=1000 s2e=60;
num = b1+u1;
ex = exp(-(day-b2)/b3);
den = 1 + ex;
model y ~ normal(num/den,s2e);
random u1 ~ normal(0,s2u) subject=tree;
run;
```

B.4 WinBUGS code for Orange Tree Example

Step 1. To check whether the model notation is correct for use, highlight the word model first. Then go to the toolbar click on model and specification. In the pop up window, select check model. If the model notation is correct, on the bottom left corner of the window a statement 'model is syntactically correct 'will show and in the Specification Tool window, the words load data and compile will change from gray to black.

```
# The Model #
model {
for (i in 1:K) {
u[i,1]~dnorm(0, tau)
for (j in 1:n) {
Y[i, j] ~ dnorm(eta[i, j], tauC)
eta[i, j] <- (phi1+u[i,1]) / (1 +  exp(-(x[j]-phi2)/phi3))
  }
}
varC ~ dgamma(3, 0.01); var~dgamma(3,0.01)
tau<-1/var; tauC<-1/varC
phi1~dunif(-1000,1000); phi2~dunif(-1000,1000); phi3~dunif(-1000,1000)
}
```

Step 2. To load data, highlight the word list and check load data. If data is successfully loaded, on the bottom left corner a statement will show 'data loaded'. Then click compile in the Specification Tool window to combine the data and model together.

```
# The Data #
list(n = 7, K = 5, x = c(118.00, 484.00, 664.00, 1004.00, 1231.00, 1372.00, 1582.00),
Y = structure(
.Data = c(30.00, 58.00, 87.00, 115.00, 120.00, 142.00, 145.00,
33.00, 69.00, 111.00, 156.00, 172.00, 203.00, 203.00,
30.00, 51.00, 75.00, 108.00, 115.00, 139.00, 140.00,
32.00, 62.00, 112.00, 167.00, 179.00, 209.00, 214.00,
30.00, 49.00, 81.00, 125.00, 142.00, 174.00, 177.00),
Dim = c(5, 7)))
```

Step 3: On specification tool window, click gen inits to generate initials and a statement 'initial values generated, model initialized'will show on the bottom left corner.

Step 4: To start the Gibbs sampling, go to the tool bar on top and choose model update. In the pop up Update tool, specify the number of updates and thinning, and click update. Then go to the tool bar check Inference and samples. In the pop up Sample Monitor Tool bar type the name of the parameter of interest, *e.g.* b1, b2,b3 in the node and click each one so that it can be added for monitoring. Next type * in the node box and specify the beginning iteration with consideration of the number of burn-in. After setting up all this, go to the tool bar select model and update, click update in the update tool window. At the meantime, in the sample monitor tool click trace, density, auto correlations so the pop up graphics window will help diagnose the convergence of samples. Click on stats on the sample monitoring tool to see the descriptive statistics of the parameters that typed in node. If the chain converges, the descriptive statistics including posterior mean, standard deviation, MC error, posterior median, 95% prediction interval will show in the stat. Otherwise, if the chain does not converge, an error message will report instead.

Appendix C

NIMH Schizophrenia Longitudinal Data Code

C.1 R code for NIMH Schizophrenia Longitudinal Data

Please see Section I.2

C.2 SAS code for NIMH Schizophrenia Longitudinal Data

```
/***Import the Data***/;
DATA  one; INFILE 'schizx1.dat';
INPUT id imps79 imps79b imps79o int tx week sweek txswk ;
run;
DATA one;
set one;
IF imps79 > -9;
run;
data three;
set one;
if imps79o=1 then do;
y1=1; y2=0; y3=0; y4=0;
end;
else if imps79o=2 then do;
y1=0; y2=1; y3=0; y4=0;
end;
else if imps79o=3 then do;
y1=0; y2=0; y3=1; y4=0;
end;
if imps79o=4 then do;
y1=0; y2=0; y3=0; y4=1;
end;
output;
run;
```

```
PROC FORMAT;
     VALUE tx      0 = 'placebo' 1 = 'drug';
run;
/*** Random Coefficient Model with Cumulative Logit ***/;
PROC GLIMMIX DATA=three METHOD=QUAD(QPOINTS=21) NOCLPRINT;
CLASS id;
MODEL imps79o = tx sweek txswk / SOLUTION DIST=MULTINOMIAL LINK=CUMLOGIT;
/* Random Intercept*/;
RANDOM INTERCEPT / SUBJECT=id;
/*Random Coefficient, COV_{int,slope}=0*/;
RANDOM INTERCEPT sweek / SUBJECT=id;
/*Random Coefficient*/;
RANDOM INTERCEPT sweek / SUBJECT=id type=un;
RUN;
/*** Random Coefficient Model with Adjacent-Category***/;
/*Random Intercept*/;
proc nlmixed data=three method=Gauss tech=newrap qtol=1e-3;
parms alpha1=0, alpha2=0, alpha3=0,beta1=0,beta2=0,beta3=0 s1=1;
/*linear predictor*/;
eta1=alpha1 + beta1*tx + beta2*sweek + beta3*txswk + t;
eta2=alpha2 + beta1*tx + beta2*sweek + beta3*txswk + t;
eta3=alpha3 + beta1*tx + beta2*sweek + beta3*txswk + t;
pi1= 1/(exp(eta1+eta2+eta3)+exp(eta1+eta2) + exp(eta1) + 1);
pi2= exp(eta1)/(exp(eta1+eta2+eta3)+exp(eta1+eta2) + exp(eta1) + 1);
pi3= exp(eta1+eta2)/(exp(eta1+eta2+eta3)+exp(eta1+eta2) + exp(eta1) + 1);
pi4= exp(eta1+eta2+eta3) / (exp(eta1+eta2+eta3)+exp(eta1+eta2) + exp(eta1) + 1);
   z = (pi1**y1)*(pi2**y2)*(pi3**y3)*(pi4**y4);
   if (z > 1e-15) then ll = log(z);
   else ll=-1e100;
   model z~ general(ll);
   random  t ~ normal(0, s1) subject = id;
run;
/*Random Coefficient*/;
proc nlmixed data=three method=Gauss noad qtol=1e-4;
*proc nlmixed data=three method=Gauss qtol=1e-4;
parms alpha1=0, alpha2=0, alpha3=0,beta1=0,beta2=0,beta3=0,s1=1,s2=1;
eta1=alpha1 + beta1*tx + beta2*sweek + beta3*txswk + u1 + u2*sweek;
eta2=alpha2 + beta1*tx + beta2*sweek + beta3*txswk + u1 + u2*sweek;
eta3=alpha3 + beta1*tx + beta2*sweek + beta3*txswk + u1 + u2*sweek;
pi1= 1/(exp(eta1+eta2+eta3)+exp(eta1+eta2) + exp(eta1) + 1);
pi2= exp(eta1)/(exp(eta1+eta2+eta3)+exp(eta1+eta2) + exp(eta1) + 1);
pi3= exp(eta1+eta2)/(exp(eta1+eta2+eta3)+exp(eta1+eta2) + exp(eta1) + 1);
pi4= exp(eta1+eta2+eta3) / (exp(eta1+eta2+eta3)+exp(eta1+eta2) + exp(eta1) + 1);
   z = (pi1**y1)*(pi2**y2)*(pi3**y3)*(pi4**y4);
   if (z > 1e-8) then ll = log(z);
   else ll=-1e100;
   model z ~ general(ll);
   /*Random Coefficient, COV_{int,slope}=0*/;
   random u1 u2 ~ normal([0,0], [s1,0,s2]) subject = id;
     /*Random Coefficient*/;
   random u1 u2 ~ normal([0,0], [s1,cov12,s2]) subject = id;
run;
```

```
/*** Random Coefficient Model with Backward Continuation Ratio ***/;
/*Random Intercept*/;
proc nlmixed data=three method=Gauss qtol=1e-4;
*proc nlmixed data=three method=Gauss noad qtol=1e-4;
parms alpha2=0, alpha3=0, alpha4=0,beta1=0,beta2=0,beta3=0, s1=1;
/*linear predictor*/;
eta2 = alpha2 + beta1*tx + beta2*sweek + beta3*txswk + u1;
eta3 = alpha3 + beta1*tx + beta2*sweek + beta3*txswk + u1;
eta4 = alpha4 + beta1*tx + beta2*sweek + beta3*txswk + u1;
p4 = exp(eta4)/(1+exp(eta4));
p3 = (1-p4)*exp(eta3)/(1+exp(eta3));
p2 = (1-p3-p4)*exp(eta2)/(1+exp(eta2));
p1 = 1-p2-p3-p4;
/*Define likelihood*/
z = (p1**y1)*(p2**y2)*(p3**y3)*(p4**y4);
if (z > 1e-8) then ll = log(z);
else ll=-1e100;
model z~ General(ll);
random u1 ~ normal(0, s1) subject = id;
run;
/*Random Coefficient*/;
proc nlmixed data=three method=Gauss qtol=1e-4;
*proc nlmixed data=three method=Gauss noad qtol=1e-4;
parms alpha2=0, alpha3=0, alpha4=0,beta1=0,beta2=0,beta3=0, s1=1, s2=1;
/*linear predictor*/;
eta2 = alpha2 + beta1*tx + beta2*sweek + beta3*txswk + u1 + u2*sweek;
eta3 = alpha3 + beta1*tx + beta2*sweek + beta3*txswk + u1 + u2*sweek;
eta4 = alpha4 + beta1*tx + beta2*sweek + beta3*txswk + u1 + u2*sweek;
p4 = exp(eta4)/(1+exp(eta4));
p3 = (1-p4)*exp(eta3)/(1+exp(eta3));
p2 = (1-p3-p4)*exp(eta2)/(1+exp(eta2));
p1 = 1-p2-p3-p4;
z = (p1**y1)*(p2**y2)*(p3**y3)*(p4**y4);
if (z > 1e-8) then ll = log(z);
else ll=-1e100;
model z~ General(ll);
   /*Random Coefficient, COV_{int,slope}=0*/;
   random u1 u2 ~ normal([0,0], [s1,0,s2]) subject = id;
   /*Random Coefficient*/;
   random u1 u2 ~ normal([0,0], [s1,cov12,s2]) subject = id;
run;
/*** Random Coefficient Model with Forward Continuation Ratio ***/;
/*Random Intercept*/;
*proc nlmixed data=three method=Gauss qtol=1e-4;
proc nlmixed data=three method=Gauss noad qtol=1e-4;
parms alpha1=0, alpha2=0, alpha3=0,beta1=0,beta2=0,beta3=0, s1=1;
/*linear predictor*/;
eta1 = alpha1 + beta1*tx + beta2*sweek + beta3*txswk + u1;
eta2 = alpha2 + beta1*tx + beta2*sweek + beta3*txswk + u1;
eta3 = alpha3 + beta1*tx + beta2*sweek + beta3*txswk + u1;
p1 = exp(eta1)/(1+exp(eta1));
p2 = (1-p1)*exp(eta2)/(1+exp(eta2));
```

```
p3 = (1-p1-p2)*exp(eta3)/(1+exp(eta3));
p4 = 1-p1-p2-p3;
/*Define likelihood*/
z = (p1**y1)*(p2**y2)*(p3**y3)*(p4**y4);
if (z > 1e-8) then ll = log(z);
else ll=-1e100;
model z~ General(ll);
random u1 ~ normal(0, s1) subject = id;
run;
/*Random Coefficient*/;
proc nlmixed data=three method=Gauss qtol=1e-4;
*proc nlmixed data=three method=Gauss noad qtol=1e-4;
parms alpha1=0, alpha2=0, alpha3=0,beta1=0,beta2=0,beta3=0, s1=1,s2=1;
/*linear predictor*/;
eta1 = alpha1 + beta1*tx + beta2*sweek + beta3*txswk + u1 + u2*sweek;
eta2 = alpha2 + beta1*tx + beta2*sweek + beta3*txswk + u1 + u2*sweek;
eta3 = alpha3 + beta1*tx + beta2*sweek + beta3*txswk + u1 + u2*sweek;
p1 = exp(eta1)/(1+exp(eta1));
p2 = (1-p1)*exp(eta2)/(1+exp(eta2));
p3 = (1-p1-p2)*exp(eta3)/(1+exp(eta3));
p4 = 1-p1-p2-p3;
/*Define likelihood*/
z = (p1**y1)*(p2**y2)*(p3**y3)*(p4**y4);
if (z > 1e-8) then ll = log(z);
else ll=-1e100;
model z~ General(ll);
   /*Random Coefficient, COV_{int,slope}=0*/;
   random u1 u2 ~ normal([0,0], [s1,0,s2]) subject = id;
   /*Random Coefficient*/;
   random u1 u2 ~ normal([0,0], [s1,cov12,s2]) subject = id;
run;
```

C.3 R code for NIMH Schizophrenia Longitudinal Data using `ordinal` pacakge

```
NIMH<-read.csv("NIMH Schizophrenia.csv")
attach(NIMH)
library(ordinal)
library(MASS)
# Random Intercept model #
# Nonadpative #
output1<-clmm(as.factor(imps79o)~tx+sweek+txswk+(1|id),link="logit",
threshold='flexible',nAGQ= -7)
# Adaptive #
output1<-clmm(as.factor(imps79o)~tx+sweek+txswk+(1|id),link="logit",
threshold='flexible',nAGQ= 3)
```

Appendix D

NIMH Schizophrenia Longitudinal Data Additional Results

D.1 Random Coefficient Model with Adjacent Categories Logit

Table D.1: R output: Random Intercept Model with Adjacent Categories Logit

Nonadaptive Gauss-Hermite Quadrature, $N_{GQ} = 21$

Parameter	Estimate	SE	DF	t-value	P-value
α_1	4.40	0.296	436	14.88	<0.0001*
α_2	1.96	0.244	436	8.05	<0.0001*
α_3	0.88	0.214	436	4.09	<0.0001*
β_{drug}	-0.05	0.240	436	-0.21	0.84
β_{time}	-0.61	0.101	436	-6.03	<0.0001*
$\beta_{drug \times time}$	-0.89	0.119	436	-7.45	<0.0001*
σ^2_{int}	2.15	0.301	436	7.15	<0.0001*
$-logL$	1698.0				

Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 3$

Parameter	Estimate	SE	DF	t-value	P-value
α_1	4.39	0.307	436	14.28	<0.0001*
α_2	1.96	0.259	436	7.57	<0.0001*
α_3	0.87	0.231	436	3.78	<0.0001*
β_{drug}	-0.05	0.265	436	-0.18	0.85
β_{time}	-0.61	0.104	436	-5.87	<0.0001*
$\beta_{drug \times time}$	-0.89	0.124	436	-7.17	<0.0001*
σ^2_{int}	2.12	0.296	436	7.16	<0.0001*
$-logL$	1698.9				

Table D.2: SAS output: Random Intercept Model with Adjacent Categories Logit

Nonadaptive Gauss-Hermite Quadrature, $N_{GQ} = 21$

Parameter	Estimate	SE	DF	t-value	P-value
α_1	4.40	0.296	436	14.88	<0.0001*
α_2	1.96	0.244	436	8.05	<0.0001*
α_3	0.88	0.214	436	4.09	<0.0001*
β_{drug}	-0.05	0.240	436	-0.21	0.83
β_{time}	-0.61	0.101	436	-6.03	<0.0001*
$\beta_{drug \times time}$	-0.89	0.120	436	-7.44	<0.0001*
σ^2_{int}	2.15	0.301	436	7.15	<0.0001*
$-logL$	1698.0				

Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 3$

Parameter	Estimate	SE	DF	t-value	P-value
α_1	4.38	0.295	436	14.89	<0.0001*
α_2	1.96	0.243	436	8.06	<0.0001*
α_3	0.87	0.214	436	4.09	<0.0001*
β_{drug}	-0.05	0.239	436	-0.20	0.84
β_{time}	-0.61	0.101	436	-6.03	<0.0001*
$\beta_{drug \times time}$	-0.89	0.119	436	-7.42	<0.0001*
σ^2_{int}	2.12	0.30	436	7.16	<0.0001*
$-logL$	1699.0				

Table D.3: R output: Random Coefficient Model with Adjacent Categories Logit assuming COV($\sigma_{int}, \sigma_{slope}$) = 0

Nonadaptive Gauss-Hermite Quadrature, $N_{GQ} = 21$					
Parameter	Estimate	SE	DF	t-value	P-value
α_1	5.48	0.385	435	14.24	<0.0001*
α_2	2.23	0.278	435	8.00	<0.0001*
α_3	0.73	0.236	435	3.09	0.002*
β_{drug}	0.08	0.264	435	0.31	0.76
β_{time}	-0.57	0.151	435	-3.81	0.0002*
$\beta_{drug \times time}$	-1.33	0.189	435	-7.05	<0.0001*
σ^2_{int}	2.65	0.470	435	5.65	<0.0001*
σ^2_{slope}	0.81	0.189	435	4.28	<0.0001*
$-logL$	1668.9				

Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 9$					
Parameter	Estimate	SE	DF	t-value	P-value
α_1	5.48	0.382	435	14.34	<0.0001*
α_2	2.23	0.275	435	8.09	<0.0001*
α_3	0.73	0.233	435	3.14	0.002*
β_{drug}	0.08	0.260	435	0.31	0.76
β_{time}	-0.57	0.150	435	-3.82	0.0002*
$\beta_{drug \times time}$	-1.33	0.189	435	-7.07	<0.0001*
σ^2_{int}	2.65	0.467	435	5.67	<0.0001*
σ^2_{slope}	0.80	0.188	435	4.29	< 0.0001*
$-logL$	1668.9				

Table D.4: SAS output: Random Coefficient Model with Adjacent Categories Logit assuming COV$(\sigma_{int}, \sigma_{slope}) = 0$

Nonadaptive Gauss-Hermite Quadrature ,$N_{GQ} = 21$					
Parameter	Estimate	SE	DF	t-value	P-value
α_1	5.49	0.387	435	14.24	<0.0001*
α_2	2.24	0.278	435	8.00	<0.0001*
α_3	0.75	0.234	435	3.10	0.002*
β_{drug}	0.06	0.262	435	0.31	0.76
β_{time}	-0.57	0.146	435	-3.81	0.0002*
$\beta_{drug\times time}$	-1.34	0.187	435	-7.05	<0.0001*
σ^2_{int}	2.66	0.467	435	5.69	<0.0001*
σ^2_{slope}	0.80	0.182	435	4.43	<0.0001*
$-logL$	1668.9				
Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 9$					
Parameter	Estimate	SE	DF	t-value	P-value
α_1	5.48	0.385	435	14.21	<0.0001*
α_2	2.23	0.278	435	8.05	<0.0001*
α_3	0.73	0.236	435	3.19	0.0015*
β_{drug}	0.08	0.265	435	0.23	0.82
β_{time}	-0.58	0.151	435	-3.92	0.0001*
$\beta_{drug\times time}$	-1.33	0.189	435	-7.17	<0.0001*
σ^2_{int}	2.66	0.470	435	5.69	<0.0001*
σ^2_{slope}	0.80	0.189	435	4.43	<0.0001*
$-logL$	1668.9				

Table D.5: R output: Random Coefficient Model with Adjacent Categories

Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 9$					
Parameter	Estimate	SE	DF	t-value	P-value
α_1	5.94	0.456	435	13.02	<0.0001*
α_2	2.58	0.349	435	7.38	<0.0001*
α_3	0.86	0.291	435	2.95	0.003*
β_{drug}	0.07	0.323	435	0.21	0.83
β_{time}	-0.71	0.180	435	-3.97	0.0001*
$\beta_{drug \times time}$	-1.38	0.212	435	-6.51	< 0.0001*
σ^2_{int}	4.34	0.892	435	4.87	<0.0001*
$\sigma_{int,slope}$	-0.85	0.331	435	-2.56	0.01*
σ^2_{slope}	1.26	0.325	435	3.88	0.0001*
$-logL$	1663.6				

Table D.6: SAS output: Random Coefficient Model with Adjacent Categories

Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 9$					
Parameter	Estimate	SE	DF	t-value	P-value
α_1	6.02	0.458	435	13.14	<0.0001*
α_2	2.62	0.348	435	7.55	<0.0001*
α_3	0.88	0.290	435	3.02	0.003*
β_1	0.07	0.321	435	0.21	0.83
β_2	-0.73	0.182	435	-4.02	<0.0001*
β_3	-1.39	0.216	435	-6.47	<0.0001*
$\sigma_{u_1^2}$	4.62	0.980	435	4.72	<0.0001*
σ_{u_1,u_2}	-0.97	0.374	435	-2.59	0.01*
$\sigma_{u_2^2}$	1.35	0.315	435	4.28	<0.0001*
$-logL$	1663.1				

D.2 Random Coefficient Model with Backward Continuation Ratio

Table D.7: R output: Random Intercept Model with Backward Continuation Ratio

Nonadaptive Gauss-Hermite Quadrature, $N_{GQ} = 3$					
Parameter	Estimate	SE	DF	t-value	P-value
α_2	5.22	0.302	436	17.28	<0.0001*
α_3	2.39	0.261	436	9.15	<0.0001*
α_4	0.90	0.232	436	3.88	<0.0001*
β_1	-0.45	0.262	436	-1.71	0.09
β_2	-0.75	0.118	436	-6.31	<0.0001*
β_3	-0.96	0.136	436	-7.06	<0.0001*
σ_u^2	2.23	0.231	436	9.66	<0.0001*
$-logL$	1710.4				
Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 3$					
Parameter	Estimate	SE	DF	t-value	P-value
α_2	5.18	0.323	436	16.05	<0.0001*
α_3	2.26	0.276	436	8.17	<0.0001*
α_4	0.68	0.252	436	2.71	0.007*
β_1	-0.14	0.286	436	-0.48	0.63
β_2	-0.76	0.120	436	-6.35	<0.0001*
β_3	-1.02	0.140	436	-7.28	<0.0001*
σ_u^2	3.10	0.395	436	7.84	<0.0001*
$-logL$	1702.5				

Table D.8: SAS output: Random Intercept Model with Backward Continuation Ratio

Nonadaptive Gauss-Hermite Quadrature, $N_{GQ} = 3$

Parameter	Estimate	SE	DF	t-value	P-value
α_2	5.22	0.302	436	17.28	<0.0001*
α_3	2.39	0.261	436	9.16	<0.0001*
α_4	0.90	0.232	436	3.88	0.0001*
β_1	-0.45	0.262	436	-1.71	0.09
β_2	-0.75	0.118	436	-6.32	<0.0001*
β_3	-0.96	0.136	436	-7.06	<0.0001*
σ_u^2	2.23	0.231	436	9.66	<0.0001*
$-logL$	1710.4				

Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 3$

Parameter	Estimate	SE	DF	t-value	P-value
α_2	5.19	0.324	436	16.04	<0.0001*
α_3	2.27	0.277	436	8.17	<0.0001*
α_4	0.69	0.253	436	2.71	0.007*
β_1	-0.14	0.287	436	-0.48	0.63
β_2	-0.76	0.120	436	-6.35	<0.0001*
β_3	-1.02	0.140	436	-7.30	<0.0001*
σ_u^2	3.14	0.401	436	7.82	<0.0001*
$-logL$	1701.9				

Table D.9: R output: Random Coefficient Model with Backward Continuation Ratio assuming COV$(\sigma_{int}, \sigma_{slope}) = 0$

Nonadaptive Gauss-Hermite Quadrature, $N_{GQ} = 9$					
Parameter	Estimate	SE	DF	t-value	P-value
α_2	6.26	0.406	435	15.43	<0.0001*
α_3	2.56	0.311	435	8.22	<0.0001*
α_4	0.59	0.273	435	2.17	0.03*
β_1	-0.005	0.307	435	-0.02	0.99
β_2	-0.67	0.169	435	-3.96	<0.0001*
β_3	-1.51	0.211	435	-7.20	<0.0001*
$\sigma^2_{u_1}$	3.67	0.587	435	6.26	<0.0001*
$\sigma^2_{u_2}$	1.11	0.230	435	4.83	<0.0001*
$-logL$	1557.2				
Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 7$					
Parameter	Estimate	SE	DF	t-value	P-value
α_2	6.23	0.399	435	15.64	<0.0001*
α_3	2.54	0.306	435	8.30	<0.0001*
α_4	0.57	0.271	435	2.11	0.04*
β_1	0.04	0.306	435	0.12	0.90
β_2	-0.69	0.172	435	-3.99	0.0001*
β_3	-1.50	0.211	435	-7.09	<0.0001*
$\sigma^2_{u_1}$	3.62	0.577	435	6.27	< 0.0001*
$\sigma^2_{u_2}$	1.06	0.231	435	4.57	< 0.0001*
$-logL$	1672.0				

Table D.10: SAS output: Random Coefficient Model with Backward Continuation Ratio assuming COV($\sigma_{int}, \sigma_{slope}$) $= 0$

Nonadaptive Gauss-Hermite Quadrature ,$N_{GQ} = 9$					
Parameter	Estimate	SE	DF	t-value	P-value
α_2	6.27	0.406	435	15.42	<0.0001*
α_3	2.56	0.311	435	8.25	<0.0001*
α_4	0.59	0.273	435	2.16	0.03*
β_1	0.004	0.308	435	0.01	0.99
β_2	-0.68	0.169	435	-4.03	<0.0001*
β_3	-1.51	0.211	435	-7.17	<0.0001*
$\sigma^2_{u_1}$	3.67	0.591	435	6.20	<0.0001*
$\sigma^2_{u_2}$	1.07	0.229	435	4.69	<0.0001*
$-logL$	1671.8				
Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 7$					
Parameter	Estimate	SE	DF	t-value	P-value
α_2	6.25	0.401	435	15.58	<0.0001*
α_3	2.55	0.307	435	8.30	<0.0001*
α_4	0.57	0.271	435	2.11	0.04*
β_1	0.04	0.307	435	0.12	0.90
β_2	-0.69	0.172	435	-3.98	<0.0001*
β_3	-1.50	0.212	435	-7.10	<0.0001*
$\sigma^2_{u_1}$	3.66	0.589	435	6.22	<0.0001*
$\sigma^2_{u_2}$	1.07	0.236	435	4.55	<0.0001*
$-logL$	1671.8				

Table D.11: R output: Random Coefficient Model with Backward Continuation Ratio

Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 9$					
Parameter	Estimate	SE	DF	t-value	P-value
α_2	6.71	0.460	435	14.57	<0.0001*
α_3	2.92	0.370	435	7.91	<0.0001*
α_4	0.75	0.325	435	2.30	0.02*
β_1	0.03	0.363	435	0.08	0.94
β_2	-0.84	0.200	435	-4.21	<0.0001*
β_3	-1.54	0.236	435	-6.53	<0.0001*
$\sigma_{u_1^2}$	5.87	1.073	435	5.48	<0.0001*
σ_{u_1,u_2}	-1.18	0.412	435	-2.87	0.004*
$\sigma_{u_2^2}$	1.65	0.432	435	3.83	0.0001*
$-logL$	1665.3				

Table D.12: SAS output: Random Coefficient Model with Backward Continuation Ratio

Nonadaptive Gauss-Hermite Quadrature ,$N_{GQ} = 21$					
Parameter	Estimate	SE	DF	t-value	P-value
α_2	6.79	0.475	435	14.28	<0.0001*
α_3	2.98	0.380	435	7.83	<0.0001*
α_4	0.77	0.332	435	2.32	0.02*
β_1	0.03	0.369	435	0.08	0.94
β_2	-0.86	0.205	435	-4.21	<0.0001*
β_3	-1.55	0.240	435	-6.48	<0.0001*
$\sigma_{u_1^2}$	6.24	1.189	435	5.25	<0.0001*
σ_{u_1,u_2}	-1.35	0.463	435	-2.91	0.004*
$\sigma_{u_2^2}$	1.76	0.382	435	4.61	<0.0001*
$-logL$	1664.6				
Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 9$					
Parameter	Estimate	SE	DF	t-value	P-value
α_2	6.78	0.473	435	14.35	<0.0001*
α_3	2.98	0.379	435	7.85	<0.0001*
α_4	0.77	0.332	435	2.33	0.02*
β_1	0.03	0.369	435	0.08	0.94
β_2	-0.86	0.205	435	-4.21	<0.0001*
β_3	-1.55	0.240	435	-6.47	<0.0001*
$\sigma_{u_1^2}$	6.24	1.19	435	5.26	<0.0001*
σ_{u_1,u_2}	-1.35	0.467	435	-2.88	0.004*
$\sigma_{u_2^2}$	1.76	0.377	435	4.67	<0.0001*
$-logL$	1664.7				

D.3 Random Coefficient Model with Forward Continuation Ratio

Table D.13: R output: Random Intercept Model with Forward Continuation Ratio

Nonadaptive Gauss-Hermite Quadrature, $N_{GQ} = 3$					
Parameter	Estimate	SE	DF	t-value	P-value
α_1	-5.38	0.296	436	-18.19	<0.0001*
α_2	-2.80	0.258	436	-10.86	<0.0001*
α_3	-1.17	0.238	436	-4.91	<0.0001*
β_1	0.27	0.267	436	1.03	0.30
β_2	0.66	0.118	436	5.62	<0.0001*
β_3	1.06	0.137	436	7.77	<0.0001*
σ_u^2	2.27	0.230	436	9.89	<0.0001*
$-logL$	1709.8				
Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 5$					
Parameter	Estimate	SE	DF	t-value	P-value
α_1	-5.39	0.312	436	-17.25	<0.0001*
α_2	-2.69	0.269	436	-10.03	<0.0001*
α_3	-0.97	0.253	436	-3.84	0.0001*
β_1	-0.01	0.286	436	-0.05	0.96
β_2	0.67	0.119	436	5.61	<0.0001*
β_3	1.14	0.140	436	8.15	<0.0001*
σ_u^2	3.13	0.392	436	7.98	<0.0001*
$-logL$	1701.9				

Table D.14: SAS output: Random Intercept Model with Forward Continuation Ratio

Nonadaptive Gauss-Hermite Quadrature, $N_{GQ} = 3$					
Parameter	Estimate	SE	DF	t-value	P-value
α_1	-5.38	0.296	436	-18.19	<0.0001*
α_2	-2.80	0.258	436	-10.86	<0.0001*
α_3	-1.17	0.238	436	-4.91	<0.0001*
β_1	0.28	0.267	436	1.03	0.30
β_2	0.66	0.118	436	5.62	<0.0001*
β_3	1.06	0.137	436	7.77	<0.0001*
σ_u^2	2.27	0.230	436	9.89	<0.0001*
$-logL$	1709.8				
Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 5$					
Parameter	Estimate	SE	DF	t-value	P-value
α_1	-5.39	0.313	436	-17.25	<0.0001*
α_2	-2.69	0.269	436	-10.03	<0.0001*
α_3	-0.97	0.253	436	-3.84	0.0001*
β_1	-0.01	0.286	436	-0.05	0.96
β_2	0.67	0.119	436	5.61	<0.0001*
β_3	1.14	0.140	436	8.16	<0.0001*
σ_u^2	3.13	0.392	436	7.98	<0.0001*
$-logL$	1701.9				

Table D.15: R output: Random Coefficient Model with Forward Continuation Ratio assuming COV($\sigma_{int}, \sigma_{slope}$) $= 0$

Nonadaptive Gauss-Hermite Quadrature, $N_{GQ} = 21$					
Parameter	Estimate	SE	DF	t-value	P-value
α_1	-6.35	0.382	435	-16.62	< 0.0001*
α_2	-2.85	0.291	435	-9.77	< 0.0001*
α_3	-0.81	0.267	435	-3.03	0.003*
β_1	-0.13	0.301	435	-0.43	0.67
β_2	0.62	0.172	435	3.63	0.0003*
β_3	1.58	0.210	435	7.51	< 0.0001*
$\sigma^2_{u_1}$	3.46	0.566	435	6.12	< 0.0001*
$\sigma^2_{u_2}$	1.09	0.231	435	4.73	< 0.0001*
$-logL$	1668.3				

Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 7$					
Parameter	Estimate	SE	DF	t-value	P-value
α_1	-6.33	0.378	435	-16.72	<0.0001*
α_2	-2.84	0.290	435	-9.79	<0.0001*
α_3	-0.81	0.266	435	-3.03	0.0026*
β_1	-0.13	0.300	435	-0.43	0.67
β_2	0.62	0.171	435	3.64	0.0003*
β_3	1.57	0.209	435	7.52	<0.0001*
$\sigma^2_{u_1}$	3.42	0.555	435	6.17	< 0.0001*
$\sigma^2_{u_2}$	1.07	0.226	435	4.76	< 0.0001*
$-logL$	1668.5				

Table D.16: SAS output: Random Coefficient Model with Forward Continuation Ratio assuming COV($\sigma_{int}, \sigma_{slope}$) $= 0$

Nonadaptive Gauss-Hermite Quadrature ,$N_{GQ} = 21$					
Parameter	Estimate	SE	DF	t-value	P-value
α_1	-6.35	0.382	435	-16.62	<0.0001*
α_2	-2.85	0.291	435	-9.77	<0.0001*
α_3	-0.81	0.267	435	-3.02	0.003*
β_1	-0.13	0.301	435	-0.43	0.67
β_2	0.63	0.172	435	3.63	0.0003*
β_3	1.58	0.210	435	7.51	<0.0001*
$\sigma^2_{u_1}$	3.46	0.566	435	6.12	<0.0001*
$\sigma^2_{u_2}$	1.09	0.231	435	4.73	<0.0001*
$-logL$	1668.3				
Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 7$					
Parameter	Estimate	SE	DF	t-value	P-value
α_1	-6.35	0.382	435	-16.62	<0.0001*
α_2	-2.85	0.291	435	-9.77	<0.0001*
α_3	-0.81	0.267	435	-3.02	0.003*
β_1	-0.13	0.301	435	-0.43	0.67
β_2	0.63	0.172	435	3.63	0.0003*
β_3	1.58	0.210	435	7.51	<0.0001*
$\sigma^2_{u_1}$	3.46	0.566	435	6.12	<0.0001*
$\sigma^2_{u_2}$	1.09	0.231	435	4.73	<0.0001*
$-logL$	1668.4				

Table D.17: R output: Random Coefficient Model with Forward Continuation Ratio

Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 7$

Parameter	Estimate	SE	DF	t-value	P-value
α_1	-6.68	0.429	435	-15.55	<0.0001*
α_2	-3.10	0.340	435	-9.12	<0.0001*
α_3	-0.92	0.307	435	-3.01	0.003*
β_1	-0.10	0.342	435	-0.30	0.77
β_2	0.75	0.193	435	3.87	0.0001*
β_3	1.58	0.227	435	6.99	<0.0001*
$\sigma_{u_1^2}$	5.09	0.957	435	5.32	<0.0001*
σ_{u_1,u_2}	-0.88	0.361	435	-2.44	0.02*
$\sigma_{u_2^2}$	1.50	0.395	435	3.80	0.0002*
$-logL$	1414.2				

Table D.18: SAS output: Random Coefficient Model with Forward Continuation Ratio

Nonadaptive Gauss-Hermite Quadrature ,$N_{GQ} = 31$					
Parameter	Estimate	SE	DF	t-value	P-value
α_1	-6.86	0.457	435	-14.99	<0.0001*
α_2	-3.22	0.363	435	-8.86	<0.0001*
α_3	-0.98	0.326	435	-3.00	<0.002*
β_1	-0.10	0.362	435	-0.27	0.79
β_2	0.80	0.204	435	3.91	0.0001*
β_3	1.61	0.237	435	6.79	<0.0001*
$\sigma_{u_1^2}$	5.95	1.176	435	5.06	<0.0001*
σ_{u_1,u_2}	-1.27	0.470	435	-2.69	0.007*
$\sigma_{u_2^2}$	1.75	0.375	435	4.67	<0.0001*
$-logL$	1662.4				
Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 7$					
Parameter	Estimate	SE	DF	t-value	P-value
α_1	-6.68	0.429	435	-15.54	<0.0001*
α_2	-3.10	0.340	435	-9.11	<0.0001*
α_3	-0.92	0.307	435	-3.01	0.003*
β_1	-0.10	0.343	435	-0.30	0.77
β_2	0.75	0.193	435	3.87	0.0001*
β_3	1.58	0.227	435	6.99	<0.0001*
$\sigma_{u_1^2}$	5.09	0.957	435	5.32	<0.0001*
σ_{u_1,u_2}	-0.88	0.361	435	-2.44	0.02*
$\sigma_{u_2^2}$	1.50	0.395	435	3.80	0.0002*
$-logL$	1664.3				

Appendix E

Health Service Research Example Code

E.1 R code for Health Service Research Example

Please see Section I.3

E.2 SAS code for Health Service Research Example

```
OPTIONS NOCENTER ;
TITLE 'Mixed-effects Analyses of San Diego Homeless Data';
DATA one;
INFILE "\sdhouse.DAT";
INPUT Id Housing Int Section8 Time1 Time2 Time3 Sect8T1 Sect8T2 Sect8T3
      NonSect8 LinTime S8LinT ;
      IF Housing = 999 then Housing = .;run;
DATA one;
set one;
if housing = 0 then do ;
y1=1; y2=0; y3=0;
end;
else if housing=1 then do;
y1=0; y2=1; y3=0;
end;
else if housing=2 then do;
y1=0; y2=0; y3=1;
end;
output;
run;
/* fixed-effects proportional odds model */
PROC LOGISTIC DESCENDING DATA=one;
MODEL Housing = Time1 Time2 Time3 Section8 Sect8T1 Sect8T2 Sect8T3;
RUN;
/* Cumulative logit: Random-intercept proportional odds model*/
```

```
PROC GLIMMIX DATA=one METHOD=QUAD(QPOINTS=21) NOCLPRINT;
CLASS id;
MODEL Housing =Time1 Time2 Time3  Section8 Sect8T1 Sect8T2 Sect8T3 /
      SOLUTION DIST=MULTINOMIAL LINK=CUMLOGIT;
RANDOM INTERCEPT / SUBJECT=id;
RUN;
/*Adjacent Category: Random-intercept proportional odds model*/
PROC NLMIXED DATA=one method=Gauss tech=newrap qtol=1e-3;
parms alpha1=0, alpha2=0, beta1=0, beta2=0, beta3=0, beta4=0,
      beta5=0, beta6=0, beta7=0 s1=1;
eta1 = alpha1 + beta1*Time1 + beta2*Time2 + beta3*Time3 +
       beta4*Section8 + beta5*Sect8T1 + beta6*Sect8T2 + beta7*Sect8T3 + u;
eta2 = alpha2 + beta1*Time1 + beta2*Time2 + beta3*Time3 +
       beta4*Section8 + beta5*Sect8T1 + beta6*Sect8T2 + beta7*Sect8T3 + u;
pi1 = 1/(exp(eta1+eta2) + exp(eta1) + 1);
pi2 = exp(eta1)/(exp(eta1+eta2) + exp(eta1) + 1);
pi3 = exp(eta1+eta2)/(exp(eta1+eta2) + exp(eta1) + 1);
z=(pi1**y1)*(pi2**y2)*(pi3**y3);
ll=log(z);
model z~general(ll);
random u~normal(0, s1) subject=id;
run;
/*Backward Continuation Ratio:Random-intercept proportional odds model*/
PROC NLMIXED DATA=one method=Gauss tech=newrap qtol=1e-3;
parms alpha2=0, alpha3=0, beta1=0, beta2=0, beta3=0, beta4=0,
      beta5=0, beta6=0, beta7=0 s1=1;
eta2 = alpha2 + beta1*Time1 + beta2*Time2 + beta3*Time3 +
       beta4*Section8 + beta5*Sect8T1 + beta6*Sect8T2 + beta7*Sect8T3 + u;
eta3 = alpha3 + beta1*Time1 + beta2*Time2 + beta3*Time3 +
       beta4*Section8 + beta5*Sect8T1 + beta6*Sect8T2 + beta7*Sect8T3 + u;
pi3 = exp(eta3)/(1+exp(eta3));
pi2 = (1-pi3)*exp(eta2)/(1+exp(eta2));
pi1 = 1-pi2-pi3;
z=(pi1**y1)*(pi2**y2)*(pi3**y3);
ll=log(z);
model z~general(ll);
random u~normal(0, s1) subject=id;
run;
/*Forward Continuation Ratio:Random-intercept proportional odds model*/
PROC NLMIXED DATA=one method=Gauss tech=newrap qtol=1e-3;
parms alpha1=0, alpha2=0, beta1=0, beta2=0, beta3=0, beta4=0,
      beta5=0, beta6=0, beta7=0, s1=1;
eta1 = alpha1 + beta1*Time1 + beta2*Time2 + beta3*Time3 +
       beta4*Section8 + beta5*Sect8T1 + beta6*Sect8T2 + beta7*Sect8T3 + u;
eta2 = alpha2 + beta1*Time1 + beta2*Time2 + beta3*Time3 +
       beta4*Section8 + beta5*Sect8T1 + beta6*Sect8T2 + beta7*Sect8T3 + u;
pi1 = exp(eta1)/(1+exp(eta1));
pi2 = (1-pi1)*exp(eta2)/(1+exp(eta2));
pi3 = 1-pi1-pi2;
z=(pi1**y1)*(pi2**y2)*(pi3**y3);
ll=log(z);
model z~general(ll);
random u~normal(0, s1) subject=id;
run;
```

Appendix F

Health Service Research Example Additional Results

F.1 Health Service Research Example output: Random Intercept Model with Adjacent-Category Logit

Table F.1: R output: Random Intercept Model with Adjacent-Category Logit

Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 10$

Parameter	Estimate	SE	DF	t-stat	P-value
$\alpha 1$	-0.33	0.169	-1.99	360	0.05*
$\alpha 2$	-2.28	0.218	-10.48	360	<0.0001*
6 Month	1.43	0.200	7.14	360	<0.0001*
12 Month	1.94	0.215	9.00	360	<0.0001*
24 Month	2.07	0.219	9.47	360	<0.0001*
Section 8	0.42	0.229	1.81	360	0.07
Section 8 at 6 Month	1.16	0.288	4.02	360	<0.0001*
Section 8 at 12 Month	0.89	0.298	2.97	360	<0.0001*
Section 8 at 24 Month	0.47	0.292	1.61	360	0.11
σ^2_{int}	1.49				
-logL	1142.27				

Table F.2: SAS output: Random Intercept Model with Adjacent-Category Logit

Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 3$					
Parameter	Estimate	SE	DF	t-stat	P-value
$\alpha 1$	-0.33	0.168	-1.99	360	0.05*
$\alpha 2$	-2.28	0.217	-10.49	360	<0.0001*
6 Month	1.42	0.200	7.13	360	<0.0001*
12 Month	1.93	0.215	9.00	360	<0.0001*
24 Month	2.07	0.219	9.47	360	<0.0001*
Section 8	0.42	0.229	1.81	360	0.07
Section 8 at 6 Month	1.16	0.288	4.02	360	<0.0001*
Section 8 at 12 Month	0.88	0.298	2.97	360	<0.0001*
Section 8 at 24 Month	0.47	0.291	1.60	360	0.11
σ^2_{int}	1.47				
-logL	1142.65				

F.2 Random Intercept Model with Backward Continuation Ratio

Table F.3: R output: Random Intercept Model with Backward Continuation Ratio

Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 10$

Parameter	Estimate	SE	DF	t-stat	P-value
$\alpha 1$	-0.45	0.189	-2.40	360	0.02*
$\alpha 2$	-2.82	0.223	-12.65	360	<0.0001*
6 Month	1.67	0.220	7.57	360	<0.0001*
12 Month	2.25	0.235	9.58	360	<0.0001*
24 Month	2.37	0.240	9.86	360	<0.0001*
Section 8	0.45	0.259	1.75	360	0.08
Section 8 at 6 Month	1.31	0.321	4.09	360	0.0001*
Section 8 at 12 Month	0.98	0.331	2.96	360	0.003*
Section 8 at 24 Month	0.55	0.326	1.69	360	0.09
σ^2_{int}	1.93				
-logL	1141.40				

Table F.4: SAS output: Random Intercept Model with Backward Continuation Ratio

Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 3$

Parameter	Estimate	SE	DF	t-stat	P-value
$\alpha 1$	-0.45	0.188	-2.41	360	0.02*
$\alpha 2$	-2.81	0.222	-12.67	360	<0.0001*
6 Month	1.67	0.220	7.56	360	<0.0001*
12 Month	2.24	0.234	9.57	360	<0.0001*
24 Month	2.36	0.240	9.86	360	<0.0001*
Section 8	0.45	0.259	1.75	360	0.08
Section 8 at 6 Month	1.31	0.321	4.08	360	0.0001*
Section 8 at 12 Month	0.98	0.331	2.96	360	0.003*
Section 8 at 24 Month	0.55	0.326	1.68	360	0.09
σ^2_{int}	1.91				
-logL	1141.75				

F.3 Random Intercept Model with Forward Continuation Ratio Logit

Table F.5: R output: Random Intercept Model with Forward Continuation Ratio

Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 10$

Parameter	Estimate	SE	DF	t-stat	P-value
α_1	0.12	0.185	0.66	360	0.51
α_2	2.52	0.227	11.07	360	<0.0001*
6 Month	-1.55	0.220	-7.04	360	<0.0001*
12 Month	-2.08	0.232	-8.96	360	<0.0001*
24 Month	-2.29	0.237	-9.64	360	<0.0001*
Section 8	-0.46	0.257	-1.81	360	0.07
Section 8 at 6 Month	-1.33	0.319	-4.15	360	<0.0001*
Section 8 at 12 Month	-1.14	0.332	-3.45	360	<0.0001*
Section 8 at 24 Month	-0.60	0.326	-1.84	360	0.07
σ^2_{int}	1.86				
-logL	1138.63				

Table F.6: SAS output: Random Intercept Model with Forward Continuation Ratio

Adaptive Gauss-Hermite Quadrature, $N_{AGQ} = 3$					
Parameter	Estimate	SE	DF	t-stat	P-value
α_1	0.12	0.184	0.66	360	0.51
α_2	2.51	0.227	11.08	360	<0.0001*
6 Month	-1.55	0.220	-7.04	360	<0.0001*
12 Month	-2.08	0.232	-8.95	360	<0.0001*
24 Month	-2.29	0.237	-9.64	360	<0.0001*
Section 8	-0.46	0.256	-1.81	360	0.07
Section 8 at 6 Month	-1.32	0.319	-4.15	360	<0.0001*
Section 8 at 12 Month	-1.14	0.332	-3.44	360	0.0006*
Section 8 at 24 Month	-0.60	0.326	-1.83	360	0.07
σ^2_{int}	1.82				
-logL	1138.15				

Appendix G

GSE10006 Smoking Study Additional Results

Table G.1: A full list of Affymetrix probes detected by Forward Stagwise method using N-fold cross validation in GSE10006 data from the smoking study. All probes are matched to Gene Symbol using R package `hgu133a2.db`.

No.	AFFY Probe	Gene Symbol	CV%
1	205281_s_at	PIGA	100%
2	229623_at		98.3%
3	240727_s_at		96.6%
4	202435_s_at	CYP1B1	91.4%
5	202436_s_at	CYP1B1	89.7%
6	1557136_at		87.9%
7	209331_s_at	MAX	87.9%
8	219563_at	LINC00341	87.9%
9	201387_s_at	UCHL1	86.2%
10	205064_at	SPRR1B	86.2%
11	211220_s_at	HSF2	86.2%
12	205513_at	TCN1	84.5%
13	224901_at		72.4%
14	202254_at	SIPA1L1	70.7%
15	242755_at		65.5%
16	229354_at		62.1%
17	218980_at	FHOD3	56.9%
18	222288_at		53.4%
19	234541_s_at		50%

Continued on next page

Table G.1 – Continued from previous page

No.	AFFY probe	Gene Symbol	CV%
20	227787_s_at		22.4%
21	232098_at		19%
22	1559403_at		17.2%
23	229302_at		12.1%
24	236261_at		12.1%
25	1565921_a_at		10.3%
26	222662_at		10.3%
27	217997_at	PHLDA1	6.9%
28	231061_at		6.9%
29	823_at	CX3CL1	6.9%
31	1560105_at		5.2%
32	211998_at	H3F3A	5.2%
33	211998_at	H3F3B	5.2%
34	226545_at		5.2%
35	243635_at		5.2%
36	202341_s_at	TRIM2	3.4%
37	220494_s_at		3.4%
38	237972_at		3.4%
39	241772_at		3.4%
40	1555682_at		1.7%
41	1558794_at		1.7%
42	1566032_at		1.7%
43	204427_s_at	TMED2	1.7%
44	208456_s_at	RRAS2	1.7%
45	213069_at	HEG1	1.7%
46	213989_x_at	SETD4	1.7%
47	215253_s_at	RCAN1	1.7%
48	221886_at	DENND2A	1.7%
49	224570_s_at		1.7%
50	224894_at		1.7%
51	229378_at		1.7%
52	232898_at		1.7%
53	237351_at		1.7%
54	240418_at		1.7%

Appendix H

Glue Grant Burn Injury Study Example Additional Results

Table H.1: A full list of Affymetrix probes detected by Forward Stagwise method using cross validation from Glue Grant burn injury data with modified Marshall score assessed on renal system as the ordinal outcome. All probes are matched to Gene Symbol using R package `hgu133a2.db`. The probe set having a non-zero coefficient is ordered in a descending order of CV% where CV% represents the percentage of times a probe set was identified by GMIFS in the cross-validation models. The test dataset in each cross-validation model includes five subjects.

No.	AFFY Probe	Gene Symbol	CV%
1	1556471_at		100%
2	203932_at	HLA-DMB	100%
3	209446_s_at		100%
4	209770_at	BTN3A1	100%
5	212033_at	RBM25	100%
6	214088_s_at	FUT3	100%
7	214909_s_at	DDAH2	100%
8	222982_x_at		100%
9	224414_s_at		100%
10	224935_at		100%
11	226932_at		100%
12	227616_at		100%
13	38918_at	SOX13	100%
14	209514_s_at	RAB27A	97%

Continued on next page

Table H.1 – Continued from previous page

No.	AFFY probe	Gene Symbol	CV%
15	214025_at	DDX28	97%
16	224478_s_at		97%
17	241710_at		97%
18	39817_s_at	C6orf108	97%
19	218191_s_at	LMBRD1	93.9%
20	1553286_at		84.8%
21	227091_at		84.8%
22	229382_at		84.8%
23	216336_x_at	MT1E	81.8%
24	231779_at		75.8%
25	227290_at		72.7%
26	232128_s_at		72.7%
27	207539_s_at	IL4	69.7%
28	229007_at		69.7%
29	204970_s_at	MAFG	66.7%
30	244348_at		63.6%
31	201968_s_at	PGM1	51.5%
32	220924_s_at	SLC38A2	51.5%
33	224797_at		51.5%
34	204944_at	PTPRG	39.4%
35	205900_at	KRT1	39.4%
36	208964_s_at	FADS1	36.4%
37	208964_s_at	MIR1908	36.4%
38	209345_s_at	PI4K2A	36.4%
39	209762_x_at	SP110	36.4%
40	233602_at		36.4%
41	215087_at	C15orf39	33.3%
42	219975_x_at	OLAH	33.3%
43	235568_at		30.3%
44	205001_s_at	DDX3Y	27.3%
45	208392_x_at	SP110	27.3%
46	208962_s_at	FADS1	21.2%
47	208962_s_at	MIR1908	18.2%
48	221763_at	JMJD1C	18.2%
49	203234_at	UPP1	15.2%
50	203547_at	CD4	15.2%
51	231380_at		15.2%
52	209958_s_at	BBS9	12.1%

Continued on next page

Table H.1 – Continued from previous page

No.	AFFY probe	Gene Symbol	CV%
53	244856_at		12.1%
54	1554280_a_at		9.1%
55	202218_s_at	FADS2	9.1%
56	213700_s_at		9.1%
57	216910_at	XPNPEP2	9.1%
58	222760_at		9.1%
59	224936_at		9.1%
60	226189_at		9.1%
61	233950_at		9.1%
62	1561437_at		6.1%
63	202313_at	PPP2R2A	6.1%
64	203633_at	CPT1A	6.1%
65	204838_s_at	MLH3	6.1%
66	221959_at	FAM110B	6.1%
67	227125_at		6.1%
68	231070_at		6.1%
69	233265_at		6.1%
70	237009_at		6.1%
71	240970_x_at		6.1%
72	1562794_at		3%
73	1567697_at		3%
74	1569955_at		3%
75	1570173_at		3%
76	1729_at	TRADD	3%
77	201155_s_at	MFN2	3%
78	204205_at	APOBEC3G	3%
79	204963_at	SSPN	3%
80	210914_at		3%
81	215837_x_at		3%
82	219587_at	TTC12	3%
83	220987_s_at	NUAK2	3%
84	220987_s_at	AKIP1	3%
85	223124_s_at		3%
86	227177_at		3%
87	227992_s_at		3%
88	233886_at		3%
89	237275_at		3%
90	240480_at		3%

Continued on next page

Table H.1 – Continued from previous page

No.	**AFFY probe**	**Gene Symbol**	**CV%**
91	241808_at		3%

Table H.2: A full list of Affymetrix probes detected by Forward Stagwise method using full set of Glue Grant burn injury data with modified Marshall score assessed on central nervous system as the ordinal outcome. All probes are matched to Gene Symbol using R package `hgu133a2.db`.

No.	AFFY Probe	Gene Symbol	Penalized Est
1	1552773_at		-0.24
2	226747_at		0.17
3	207564_x_at	OGT	0.16
4	214909_s_at	DDAH2	-0.14
5	218078_s_at	ZDHHC3	-0.12
6	226932_at		0.11
7	241133_at		0.09
8	203633_at	CPT1A	0.08
9	204316_at	RGS10	0.08
10	227345_at		-0.07
11	204970_s_at	MAFG	-0.06
12	205686_s_at	CD86	0.06
13	205517_at	GATA4	-0.06
14	205425_at	HIP1	-0.06
15	1552772_at		-0.05
16	208433_s_at	LRP8	-0.05
17	206110_at	HIST1H3H	0.05
18	219452_at	DPEP2	0.05
19	236278_at		-0.05
20	206114_at	EPHA4	-0.05
21	225755_at		0.05
22	233602_at		-0.04
23	209553_at	VPS8	0.04
24	209553_at	LOC100505729	0.04
25	1553575_at		0.04
26	211998_at	H3F3A	0.04
27	211998_at	H3F3B	0.04
28	229821_at		0.04
29	201978_s_at	KIAA0141	0.04
30	205781_at	C16orf7	-0.04
31	229593_at		0.03
32	1565809_x_at		-0.03
33	211743_s_at	PRG2	-0.03
34	236487_at		0.03

Continued on next page

Table H.2 – Continued from previous page

No.	AFFY probe	Gene Symbol	Penalized Est
35	203932_at	HLA-DMB	0.03
36	242565_x_at		0.03
37	244553_at		0.03
38	228338_at		0.03
39	220614_s_at	C6orf103	-0.02
40	60474_at	FERMT1	0.02
41	226630_at		0.02
42	242311_x_at		-0.02
43	240416_at		-0.02
44	213624_at	SMPDL3A	-0.02
45	206705_at	TULP1	-0.02
46	202381_at	ADAM9	-0.02
47	222287_at	TRDN	0.02
48	241837_at		-0.02
49	201810_s_at	LOC100505696	-0.02
50	201810_s_at	SH3BP5	-0.02
51	202673_at	DPM1	0.01
52	238495_at		-0.01
53	200899_s_at	MGEA5	0.01
54	228656_at		0.01
55	225051_at		0.01
56	226531_at		-0.01
57	1561590_a_at		-0.01
58	202925_s_at	PLAGL2	-0.01
59	203389_at	KIF3C	-0.01
60	204611_s_at	PPP2R5B	-0.01
61	205098_at	CCR1	-0.01
62	231779_at		-0.01
63	224102_at		0.01
64	214469_at	HIST1H2AE	0.00
65	214469_at	HIST1H2AB	0.00
66	243211_at		-0.00
67	229962_at		0.00
68	213298_at	NFIC	-0.00

Table H.3: A full list of Affymetrix probes detected by Forward Stagwise method using full set of Glue Grant burn injury data with modified aggregated Marshall score as the ordinal outcome. All probes are matched to Gene Symbol using R package `hgu133a2.db`.

No.	**AFFY Probe**	**Gene Symbol**	**Penalized Est**
1	214909_s_at	DDAH2	-0.23
2	203932_at	HLA-DMB	0.18
3	241133_at		0.14
4	235568_at		-0.13
5	202381_at	ADAM9	-0.08
6	225755_at		0.07
7	231779_at		-0.07
8	213624_at	SMPDL3A	-0.07
9	224414_s_at		-0.07
10	212033_at	RBM25	0.05
11	209375_at	XPC	0.05
12	204970_s_at	MAFG	-0.05
13	202973_x_at	FAM13A	0.05
14	209514_s_at	RAB27A	-0.04
15	220924_s_at	SLC38A2	-0.04
16	223073_at		-0.04
17	227265_at		0.04
18	236487_at		0.04
19	238994_at		-0.03
20	1557821_at		0.03
21	240384_at		-0.03
22	214025_at	DDX28	-0.02
23	227345_at		-0.02
24	226932_at		0.02
25	218380_at	LOC728392	0.02
26	213199_at	C2CD3	0.01
27	206110_at	HIST1H3H	0.01
28	219681_s_at	RAB11FIP1	0.01
29	229306_at		-0.01
30	217443_at		-0.01
31	223203_at		0.01
32	209770_at	BTN3A1	0.01
33	202116_at	DPF2	0.01
34	220112_at	ANKRD55	-0.00
35	226747_at		0.00

Appendix I

R code for R package ordinalmixed with Applications

I.1 Source Code

```
##############################################################################
### START "Compile_step1.R ###
pairwise.multiply1 <- function(A,B){
                   output<-matrix(nc=dim(A)[2],nr=dim(A)[1])
                   for (ii in 1: dim(A)[1]){
                   for (jj in 1: dim(A)[2]){
                        output[ii,jj]<-(A[ii,jj]*B[ii,jj])
                        }
                   }
                   output
                   }
##############################################################################
forward.stagewise.cum <- function(x, y, epsilon, tol, cut.prop) {
                       X <- as.matrix(cbind(x,-x))
                       y <- as.numeric(y)
                       levels <- sort(unique(y))
                       cat <- length(levels)
                       Ymat<-matrix(0,nrow=length(y),ncol=cat)
                        for (ii in 1:cat){
                              index <- which(y==ii, arr.ind=T)
                              Ymat[index,ii] <- 1
                          }
                       n <- dim(X)[1]
                       p <- dim(X)[2]
                      alpha <- vector(length=cat-1, mode="numeric")
                      for (ii in 1:length(alpha)){
                          kk <- 1:ii
                          alpha[ii] <- log(sum(Ymat[,kk])/(n - sum(Ymat[,kk])))
                          }
                       beta <- rep(0, ncol(X))
```

```
                        names(beta)<- 1:dim(X)[2]
                        output <- beta.list <- Num.nonzero <- list()
kk <- 1
        repeat{
        p <- matrix(0, nrow=dim(X)[1], ncol=cat-1)
        pi <- matrix(0, nrow=dim(X)[1], ncol=cat)
        for (ii in 1:(cat-1)){
            p[,ii] <- exp(rep(alpha[ii],dim(X)[1]) + (X%*%beta)) /
                      (1 + exp(rep(alpha[ii],dim(X)[1]) + (X%*%beta)))
        }
        for (ii in 1:cat){
             if (ii==1){pi[,ii] <- p[,ii]
              } else if (ii==cat) {pi[,cat] <- 1-p[,(cat-1)]
             } else {pi[,ii] <- p[,ii]-p[,(ii-1)]}
        }
    # Check if pi has zero elements #
    for (ii in 1:cat){
      for (jj in 1:dim(pi)[1]){
        if (pi[jj,ii]==0) {pi[jj,ii] <- 1e-10}
        }
    }
        LL.tmp <- pairwise.multiply1(Ymat, log(pi))
        LL0 <- -sum(LL.tmp)
    # First derivative #
        u <- matrix(0, ncol=cat, nrow=n)
        for (ii in 1:cat){
         if (ii==1){u[,1] <- pairwise.multiply1(as.matrix(-Ymat[,1]),
                                                as.matrix(1-p[,1]))
         }
         else if (ii==cat){u[,cat] <- pairwise.multiply1(as.matrix(Ymat[,cat]),
                                                         as.matrix(p[,(cat-1)]))
         } else {u[,ii] <- pairwise.multiply1(as.matrix(-Ymat[,ii]),
                                            as.matrix(1-p[,(ii-1)]-p[,ii]))
         }
        }
        u <- apply(u, 1, sum)
    # Forward Stagewise variable selection #
      jacobian <- t(X) %*% u
      if (min(jacobian)<=0) {update.j <- which.min(jacobian)} #
      beta[update.j] <- beta[update.j]+ epsilon
      # Caculate the updated likelihood #
         for (ii in 1:(cat-1)){
            p[,ii] <- exp(rep(alpha[ii],dim(X)[1]) + (X%*%beta)) /
                      (1 + exp(rep(alpha[ii],dim(X)[1]) + (X%*%beta)))
         }
         for (ii in 1:cat){
             if (ii==1){pi[,ii] <- p[,ii]
              } else if (ii==cat) {pi[,cat] <- 1-p[,(cat-1)]
             } else {pi[,ii] <- p[,ii]-p[,(ii-1)]}
         }
# pi can be 0, fit too good! #
# Check if pi has zero elements #
```

```
   for (ii in 1:cat){
     for (jj in 1:dim(pi)[1]){
       if (pi[jj,ii]==0) {pi[jj,ii] <- 1e-10}
       }
   }
        LL.tmp <- pairwise.multiply1(Ymat, log(pi))
        LL1 <- -sum(LL.tmp)
   beta2 <- beta[1:dim(x)[2]] - beta[(dim(x)[2]+1):dim(X)[2]]
   names(beta2) <- 1:dim(x)[2]
   beta.list[[kk]] <- beta2[beta2!=0]
   Num.nonzero[[kk]] <- length(beta2[beta2!=0])
   if ( (abs(LL1-LL0) < tol) || (Num.nonzero[[kk]] > dim(x)[2]*cut.prop)) {
              break
   }
          LL0 <- LL1
          kk <- kk + 1
   }
   list(beta.list=beta.list, num.nonzero=Num.nonzero)
   }
### END Compile_step1.R ###
###################################################################
###################################################################
### START CumForwardStagewise_v1.R ###
library(alabama)
###Pairwise.multiply function ###
pairwise.multiply1 <- function(A,B){
                   output<-matrix(nc=dim(A)[2],nr=dim(A)[1])
                   for (ii in 1: dim(A)[1]){
                   for (jj in 1: dim(A)[2]){
                        output[ii,jj]<-(A[ii,jj]*B[ii,jj])
                        }
                   }
                   output
                   }
###################################################################
### A loglikelihood with respect to the intercept only ###
logLL <- function(x, y, beta1) {
         y <- as.numeric(y)
         levels <- sort(unique(y))
         cat <- length(levels)
         alpha <- rep(0, (cat-1))
         p <- matrix(0, nrow=dim(x)[1], ncol=cat-1)
         pi <- matrix(0, nrow=dim(x)[1], ncol=cat)
         Ymat<-matrix(0,nrow=length(y),ncol=cat)
                        for (ii in 1:cat){
                              index <- which(y==ii, arr.ind=T)
                              Ymat[index,ii] <- 1
                          }
         #if(scale==T) { x <- scale(x)}
         newx <- x[,as.numeric(names(beta1))]
         logL <- function(alpha) {
                 for (ii in 1:(cat-1)){
                 p[,ii] <- exp(rep(alpha[ii], dim(x)[1]) +
                              (as.matrix(newx)%*%as.matrix(beta1))) /
```

```
                  (1 + exp(rep(alpha[ii],dim(x)[1]) +
                             (as.matrix(newx)%*%as.matrix(beta1))))
                  }
                  for (ii in 1:cat){
                       if (ii==1){pi[,ii] <- p[,ii]
                       } else if (ii==cat) {pi[,cat] <- 1-p[,(cat-1)]
                       } else {
                      if (alpha[ii] > alpha[ii-1]) {pi[,ii] <- p[,ii]-p[,(ii-1)]
                      } else {pi[,ii] <- 1}
                  }
          }
         LL.tmp <- pairwise.multiply1(Ymat, log(pi))
         -sum(LL.tmp)
      }
if (cat > 2) {
hin<-function(alpha){
     h<-rep(NA,(cat-2))
     for (ii in 1:(cat-2)){
          h[ii]<- alpha[ii+1]-alpha[ii]
          }
     h
     }
hin.jac<-function(alpha){
        j<-matrix(0, (cat-2), (cat-1))
        for (ii in 1:(cat-2)){
             j[ii, (ii:(ii+1))] <- c(-1,1)
             }
        j
        }
alpha0 <- 1:(cat-1)
fit <-constrOptim.nl(par=alpha0, logL, hin=hin,hin.jac=hin.jac,
                       control.outer=list(trace=F))
}  else if (cat==2) {
   alpha0 <- 0
   fit <-  optim(par=alpha0, logL, method="BFGS", hessian=T)
}
return(fit)
}
###############################################################################
Predict <- function(x, y, alpha, beta1) {
           num.cat <- nlevels(y)
           newx <- as.matrix(x[,as.numeric(names(beta1))])
           xbeta <- matrix(rep(newx %*% as.matrix(beta1), num.cat), nc=num.cat)
           alpha1 <- c(alpha,500)
           eta <- sweep(xbeta, 2, alpha1, "+")
           cumulative.p <- t(G(eta))
           mat <- diag(1,num.cat)
           mat1<- cbind(diag(-1,num.cat-1), (rep(0,num.cat-1)))
           mat2 <- rbind(rep(0,num.cat), mat1)
           trans.mat <- mat + mat2
           p <- t(trans.mat %*% cumulative.p)
           pred.y <- apply(p, 1, which.max)
           return(pred.y)
   }
### END CumForwardStagewise_v1.R ###
############################################################################
############################################################################
### START Compile_step2.R ###
output.CumFS <- function(forwardstagewise, x, y) {
                Num.nonzero <- unlist(forwardstagewise$num.nonzero)
                beta.list <- forwardstagewise$beta.list
                unique.num <- unique(Num.nonzero)
                nonzero <- data.frame(Num.nonzero, 1:length(Num.nonzero))
                aic <- bic <- LL <- step <- rep(0, length(unique.num))
                temp <- NULL
     for (ii in 1:length(unique.num)){
          temp <- nonzero[nonzero[,1]==unique.num[ii],]
          KK <- temp[dim(temp)[1],2]
          new.beta <- beta.list[[KK]]
            fit <- logLL(x, y, beta1=new.beta)
             new.alpha <- fit$par
              new.LL <- fit$value
                aic[ii] <- 2*unique.num[ii] + 2*new.LL
              bic[ii] <- unique.num[ii]*log(dim(x)[1]) + 2*new.LL
             LL[ii] <- new.LL
          step[ii] <- KK
     }
     output <- data.frame(step, unique.num, aic, bic, LL)
     return(output)
     }
###########################################################################
steps <- function(output, range, criteria){
               if (criteria=="AIC") {
                   optim.step <- output[which.min(output[,"aic"]),"unique.num"]
                   num.index <- (optim.step-range):(optim.step+range)
                   step.index <- rep(0,length(num.index))
                   for (ii in 1:length(step.index)){
                   step.index[ii] <- output[output[,"unique.num"]== num.index[ii], "step"]
                   }
               } else if (criteria=="BIC") {
                   optim.step <- output[which.min(output[,"bic"]),"unique.num"]
                   num.index <- (optim.step-range):(optim.step+range)
                   step.index <- rep(0,length(num.index))
                   for (ii in 1:length(step.index)){
                    step.index[ii] <- output[output[,"unique.num"]== num.index[ii], "step"]
                   }
               }
               return(step.index)
              }
### END Compile_step2.R ###
###########################################################################
###########################################################################
### START ordinalmixedv2.R ###
# Load packages #
library(numDeriv)
library(ucminf)
library(glmmML)
library(MASS)
library(optimx)
library(alabama)
# 1.G function used in cumulative and continuation ratio logit #
 G <- function(z){
      G=exp(z)/(1+exp(z))
      return(G)
      }
##########################################################################
# 2. pairwise.multiply function #
pairwise.multiply <- function(A,B){
                   output<-matrix(nc=dim(A)[2],nr=dim(A)[1])
                   for (ii in 1: dim(A)[1]){
                   for (jj in 1: dim(A)[2]){
                        output[ii,jj]<-(A[ii,jj]*B[ii,jj])
                        }
                   }
                   prod(output[output!=0])
                   }
###########################################################################
# 3. This function will be used to indicate which category the obs falls.
# 1 indicates the obs falls into the corresponding category while others are
# set to be 0.
index.all1 <- function (subject, id.name, predictor.name, response.name,
                        ordinal.level, data) {
              sub.matrix <- as.matrix(data[data[,id.name]==subject,
                            c(predictor.name, response.name)])
              if (dim(sub.matrix)[2]==1){
                  sub.matrix <- t(sub.matrix)
              }
              tmp <- subset(sub.matrix, select=response.name)
              index.matrix <- matrix(nr=dim(sub.matrix)[1],
                                     nc=length(ordinal.level))
              for (ii in 1:length(ordinal.level)) {
                  index.matrix[,ii] <-ifelse(tmp==ordinal.level[ii],1,0)
              }
              return (index.matrix)
              }
##########################################################################
# 4. This function calculates the probabilities of each category ....
# should talk about predicted or expected!#
linear.predictor <- function(response.name, predictor.name, id.name, data,
                             time.name, nGauss=NULL, gene.name=NULL, beta.gene=NULL,
 link=c("Cumulative", "Adjacent", "Forward CR", "Backward CR"),
model=c("Random Intercept", "Random Coefficient", "Indep Random Coefficient"),
Adaptive=c("TRUE", "FALSE"), Cholesky=c("TRUE", "FALSE"),
                             num.cat, ordinal.level,
                             alpha, beta, sigma, weight, node, new.weight,
                             random1, random2, subject){
 link <- match.arg(link)
 model <- match.arg(model)
 Adaptive <- match.arg(Adaptive)
 Cholesky <- match.arg(Cholesky)
 sub.matrix <- data.matrix(data[data[,id.name]==subject,
                                           c(predictor.name, response.name)])
if (dim(sub.matrix)[2]==1){
                       sub.matrix <- t(sub.matrix)
                       }
if (is.null(beta.gene)==F){
                     xgene <- data.matrix(data[data[,id.name]==subject, gene.name])
                    }
if(is.null(time.name)==TRUE){
                       time.name <- predictor.name[1]
                    }
if(is.null(nGauss)==TRUE){nGauss <- 10}
if (c("Cumulative","Adjacent","Forward CR","Backward CR")[charmatch(link,
    c("Cumulative","Adjacent","Forward CR","Backward CR"))] == "Cumulative")  {
    alpha1 <- c(alpha, 500) # set the floor and ceiling value #
 if (is.null(beta.gene)==T) {
        xbeta <- matrix(rep((sub.matrix[, predictor.name] %*% as.matrix(beta) +
                 random1 + random2*sub.matrix[,time.name]), num.cat), nc=num.cat)
    } else if (is.null(beta.gene)==F) {
    xbeta <- matrix(rep(sub.matrix[, predictor.name] %*% as.matrix(beta) +
                 random1 + random2*sub.matrix[,time.name]+xgene %*% beta.gene,
 num.cat), nc=num.cat)
}
     eta <- sweep(xbeta, 2, alpha1, "+")
         cumulative.p <- t(G(eta))
         mat <- diag(1,num.cat)
         mat1<- cbind(diag(-1,num.cat-1), (rep(0,num.cat-1)))
         mat2 <- rbind(rep(0,num.cat), mat1)
         trans.mat <- mat + mat2
         p <- t(trans.mat %*% cumulative.p)
}
if (c("Cumulative","Adjacent","Forward CR","Backward CR")[charmatch(link,
    c("Cumulative","Adjacent","Forward CR","Backward CR"))] == "Adjacent") {
          visit <- 1:dim(sub.matrix)[1]
predict.adj <- function(visit){
               new.eta <- eta <- vector(length=num.cat)
# Random effect model with penalized estimate(ordinalmixedalpha.R)#
 if (is.null(beta.gene)==F){
              for (ii in 2:num.cat) {
                   eta[1] <- 1
                   eta[ii] <- alpha[ii-1] + sum(beta*sub.matrix[visit,predictor.name])+
   sum(beta.gene * xgene[visit,])+ random1 + random2*sub.matrix[visit,time.name]
              }
           } else if (is.null(beta.gene)==T){
               for (ii in 2:num.cat) {
                   eta[1] <- 1
                   eta[ii] <- alpha[ii-1] + sum(beta*sub.matrix[visit,predictor.name])+
                              random1 + random2*sub.matrix[visit,time.name]
               }
           }
                for(kk in 2:num.cat) {
                   new.eta[1] <- eta[1]
                   new.eta[kk] <- exp(sum(eta[2:kk]))
                }
                p <- new.eta/sum(new.eta)
                return(p)
             }
                p <- matrix( unlist(lapply(visit, predict.adj)),
                             nr=dim(sub.matrix)[1], nc=num.cat, byrow=T)
             }
if (c("Cumulative","Adjacent","Forward CR","Backward CR")[charmatch(link,
    c("Cumulative","Adjacent","Forward CR","Backward CR"))] == "Backward CR") {
               visit <- 1:dim(sub.matrix)[1]
predict.adj <- function(visit){
               p <- eta <- vector(length=num.cat)
      if (is.null(beta.gene)==F){
              for (ii in 2:num.cat) {
                   eta[1] <- 1
                   eta[ii] <- alpha[ii-1] + sum(beta*sub.matrix[visit,predictor.name])+
      sum(beta.gene * xgene[visit,])+ random1 + random2*sub.matrix[visit,time.name]
              }
           } else if (is.null(beta.gene)==T){
               for (ii in 2:num.cat) {
                   eta[1] <- 1
                   eta[ii] <- alpha[ii-1] + sum(beta*sub.matrix[visit,predictor.name])+
                              random1 + random2*sub.matrix[visit,time.name]
               }
           }
               p[num.cat] <- exp(eta[num.cat])/(1+exp(eta[num.cat]))
               for (kk in (num.cat-1):1) {
                    tmp <- sum(p[(kk+1):num.cat])
               if (kk >=2) {
                    p[kk] <- (1-tmp)* exp(eta[kk])/ (1+ exp(eta[kk]))
               } else {
                    p[kk] <-  1-tmp
                  }
                 }
                return(p)
               }
               p <- matrix( unlist(lapply(visit, predict.adj)),
                            nr=dim(sub.matrix)[1], nc=num.cat, byrow=T)
            }
if (c("Cumulative","Adjacent","Forward CR","Backward CR")[charmatch(link,
    c("Cumulative","Adjacent","Forward CR","Backward CR"))] == "Forward CR"){
               visit <- 1:dim(sub.matrix)[1]
predict.adj <- function(visit){
               p <- eta <- vector(length=num.cat)
         if (is.null(beta.gene)==F){
              for (ii in 1:(num.cat-1)) {
                   eta[num.cat] <- 1
                   eta[ii] <- alpha[ii] + sum(beta*sub.matrix[visit,predictor.name])+
         sum(beta.gene * xgene[visit,])+ random1 + random2*sub.matrix[visit,time.name]
              }
           } else if (is.null(beta.gene)==T){
               for (ii in 1:(num.cat-1)) {
                   eta[num.cat] <- 1
                   eta[ii] <- alpha[ii] + sum(beta*sub.matrix[visit,predictor.name])+
                              random1 + random2*sub.matrix[visit,time.name]
               }
           }
               p[1] <- exp(eta[1])/(1 + exp(eta[1]))
               for (kk in 2:num.cat){
                    tmp <- sum(p[1:(kk-1)])
               if (kk < num.cat){
                    p[kk] <- (1-tmp) * exp(eta[kk])/(1 + exp(eta[kk]))
               } else {
                    p[kk] <- 1-tmp
                 }
               }
               return(p)
              }
               p <- matrix( unlist(lapply(visit, predict.adj)),
                            nr=dim(sub.matrix)[1], nc=num.cat, byrow=T)
              }
              return(p)
          }
################################################################################
# 5. This function creates outputs table #
chol2var <- function (data, response.name, predictor.name, parm, hessian, df,
                      model=c("Random Intercept", "Random Coefficient",
            "Indep Random Coefficient")){
            model <- match.arg(model)
            hessian <- as.matrix(-hessian)
            var.mat <- ginv(hessian)
if (c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
    c("Indep Random Coefficient","Random Coefficient",
      "Random Intercept"))] == "Random Intercept"){
            index.u1 <- dim(hessian)[1]
            se.sigmau1 <- sqrt(4*exp(4*parm[index.u1])* var.mat[index.u1,index.u1])
            sigmau1 <- exp(parm[index.u1])^2
            new.parm <- c(parm[1:(index.u1-1)], sigmau1)
            new.se <- c(sqrt(diag(var.mat))[1:(index.u1-1)],se.sigmau1)
    } else
if (c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
    c("Indep Random Coefficient","Random Coefficient",
      "Random Intercept"))] == "Indep Random Coefficient"){
                index.u1 <- dim(hessian)[1]-1
                index.u2 <- dim(hessian)[1]
            # From Delta's method #
            se.sigmau1 <- sqrt(4*exp(4*parm[index.u1])* var.mat[index.u1,index.u1])
            se.sigmau2 <- sqrt(4*exp(4*parm[index.u2])* var.mat[index.u2,index.u2])
            se.sigma <- c(se.sigmau1, se.sigmau2)
            sigmau1 <- exp(parm[index.u1])^2
            sigmau2 <- exp(parm[index.u2])^2
            sigma <- c(sigmau1, sigmau2)
            new.parm <- c(parm[1:(index.u1-1)], sigma)
            new.se <- c(sqrt(diag(var.mat))[1:(index.u1-1)],se.sigma)
            } else
if (c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
    c("Indep Random Coefficient","Random Coefficient",
    "Random Intercept"))] == "Random Coefficient") {
               index.l1 <- dim(hessian)[1]-2
               index.l2 <- dim(hessian)[1]-1
               index.l3 <- dim(hessian)[1]
            # SE for sigmau1 #
            se.sigmau1 <- sqrt(4*exp(4*parm[index.l1])* var.mat[index.l1,index.l1])
            # SE for sigmu2 #
            dev <- c(2*exp(2*parm[index.l3]), 2*parm[index.l2])
            mat <- var.mat[index.l2:index.l3,index.l2:index.l3]
            se.sigmau2 <- sqrt(t(dev) %*% mat %*% dev)
            # SE for covu1u2 #
            covu1u2 <- exp(parm[index.l1])*parm[index.l2]
            dev <- c(1, 1/parm[index.l2])
            mat <- var.mat[index.l1:index.l2, index.l1:index.l2]
            var.logcov <- t(dev) %*% mat %*% dev
            se.cov <- sqrt((covu1u2)^2 * var.logcov)
            new.parm <- c(parm[1:(index.l1-1)], exp(parm[index.l1])^2, covu1u2,
                          exp(parm[index.l3])^2+parm[index.l2]^2)
            new.se <- c(sqrt(diag(var.mat))[1:(index.l1-1)], se.sigmau1,
                        se.cov, se.sigmau2)
            }
            t.stat <- new.parm/new.se
            p.value <- 2*pt(abs(t.stat), df=df, lower.tail=F)
            output <- data.frame(round(new.parm,2), round(new.se,3),
                                 round(t.stat,2), round(p.value,4))
            colnames(output) <- c("parm", "SE", "t.stat", "p.value")
            response <- data[,response.name]
            num.cat <-  nlevels(as.factor(response))
            name.alpha <- paste("alpha",1:(num.cat-1),sep="")
            name.beta <- paste("beta", predictor.name[1:length(predictor.name)], sep="_")
if (c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
    c("Indep Random Coefficient","Random Coefficient",
      "Random Intercept"))] == "Random Intercept") {
            name.sigma <- c("sigma_int^2")
            } else
if (c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
    c("Indep Random Coefficient","Random Coefficient",
      "Random Intercept"))] == "Indep Random Coefficient") {
            name.sigma <- c("sigma_int^2", "sigma_time^2")
            } else
if (c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
    c("Indep Random Coefficient","Random Coefficient",
       "Random Intercept"))] == "Random Coefficient") {
            name.sigma <- c("sigma_int^2", "cov_int_time", "sigma_time^2")
            }
            rownames(output) <- c(name.alpha, name.beta, name.sigma )
            return(output)
            }
# se2var creates output when Cholesky=='False', transform SE to Var #
# Only valid for "Random Coefficient indep" model #
se2var <- function (data, response.name, predictor.name, parm, hessian, df,
                    model=c("Random Intercept", "Random Coefficient",
"Indep Random Coefficient")){
          model <- match.arg(model)
           hessian <- as.matrix(-hessian)
           var.mat <- ginv(hessian)
           se <- sqrt(diag(var.mat))
if (c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
    c("Indep Random Coefficient","Random Coefficient",
       "Random Intercept"))] == "Random Intercept"){
               index.u1 <- dim(hessian)[1]
               sigmau1 <- parm[index.u1]^2
               se.sigmau1 <- 2*abs(parm[index.u1])*se[index.u1]
               new.parm <- c(parm[1:(index.u1-1)], sigmau1)
               new.se <- c(se[1:(index.u1-1)], se.sigmau1)
    } else
if (c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
    c("Indep Random Coefficient","Random Coefficient",
       "Random Intercept"))] == "Indep Random Coefficient"){
               index.u1 <- dim(hessian)[1]-1
               index.u2 <- dim(hessian)[1]
               sigmau1 <- parm[index.u1]^2
               sigmau2 <- parm[index.u2]^2
               sigma <- c(sigmau1, sigmau2)
         # Delta's method#
               se.sigmau1 <- 2*abs(parm[index.u1])*se[index.u1]
               se.sigmau2 <- 2*abs(parm[index.u2])*se[index.u2]
               se.sigma <- c(se.sigmau1, se.sigmau2)
               new.parm <- c(parm[1:(index.u1-1)], sigma)
               new.se <- c(se[1:(index.u1-1)], se.sigma)
            }
               t.stat <- new.parm/new.se
               p.value <- 2*pt(abs(t.stat), df=df, lower.tail=F)
               output <- data.frame(round(new.parm,2), round(new.se,3),
                                    round(t.stat,2), round(p.value,4))
               colnames(output) <- c("parm", "SE", "t.stat", "p.value")
               response <- data[,response.name]
               num.cat <-  nlevels(as.factor(response))
               name.alpha <- paste("alpha",1:(num.cat-1),sep="")
               name.beta <- paste("beta", predictor.name[1:length(predictor.name)], sep="_")
if (c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
    c("Indep Random Coefficient","Random Coefficient",
      "Random Intercept"))] == "Random Intercept") {
            name.sigma <- c("sigma_int^2")
            } else
if (c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
    c("Indep Random Coefficient","Random Coefficient",
       "Random Intercept"))] == "Indep Random Coefficient") {
            name.sigma <- c("sigma_int^2", "sigma_time^2")
            } else
if (c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
    c("Indep Random Coefficient","Random Coefficient",
       "Random Intercept"))] == "Random Coefficient") {
            name.sigma <- c("sigma_int^2", "cov_int_time", "sigma_time^2")
            }
            rownames(output) <- c(name.alpha, name.beta, name.sigma )
               return(output)
           }
################################################################################
# 6. This function aggregates all other functions defined previously #
ordinal.model <- function (response.name, predictor.name, id.name, time.name,
                           data, nGauss=NULL, gene.name=NULL, beta.gene=NULL,
  link=c("Cumulative", "Adjacent", "Forward CR", "Backward CR"),
                          model=c("Random Intercept", "Random Coefficient",
                          "Indep Random Coefficient"),
                          Adaptive=c("TRUE", "FALSE"), Cholesky=c("TRUE", "FALSE"),
                          parm, ...){
                   response <- data[,response.name]
                   predictor <- data[,predictor.name]
                   id <- as.numeric(unique(data[,id.name]))
                   num.cat <-  nlevels(as.factor(response)) #maybe problem#
                   ordinal.level <- as.numeric(levels(as.factor(response)))
           link <- match.arg(link)
                   model <- match.arg(model)
                   Adaptive <- match.arg(Adaptive)
                   Cholesky <- match.arg(Cholesky)
   if(is.null(nGauss)==TRUE) { nGauss <- 10}
                   # Define random effects #
                   random1 <- 0; random2 <- 0
                   # Define ordinal model parameters #
                   alpha <- vector(length=num.cat-1, mode='numeric')
                   alpha <- parm[1:length(alpha)]
                   beta  <- vector(length=length(predictor.name), mode='numeric')
                   beta  <- parm[(length(alpha)+1):(length(alpha)+length(beta))]
if (c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
    c("Indep Random Coefficient","Random Coefficient",
       "Random Intercept"))] == "Random Intercept"){
if (c("TRUE","FALSE")[charmatch(Cholesky, c("TRUE","FALSE"))]=="FALSE"){
                   sigma <- parm[(length(alpha)+length(beta)+1):length(parm)]
                   } else
if (c("TRUE","FALSE")[charmatch(Cholesky, c("TRUE","FALSE"))]=="TRUE"){
                   L <- parm[(length(alpha)+length(beta)+1):length(parm)]
                   sigma <- exp(L[1])
                   }
        }  else
if (c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
    c("Indep Random Coefficient","Random Coefficient",
      "Random Intercept"))] == "Indep Random Coefficient"){
if (c("TRUE","FALSE")[charmatch(Cholesky, c("TRUE","FALSE"))]=="FALSE"){
```

```
                  sigma <- parm[(length(alpha)+length(beta)+1):length(parm)]
                  } else
if (c("TRUE","FALSE")[charmatch(Cholesky, c("TRUE","FALSE"))]=="TRUE"){
                  L <- parm[(length(alpha)+length(beta)+1):length(parm)]
                  sigma <- c(exp(L[1]), exp(L[2]))
                  }
        } else
if (c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
    c("Indep Random Coefficient","Random Coefficient",
       "Random Intercept"))] == "Random Coefficient"){
if (c("TRUE","FALSE")[charmatch(Cholesky, c("TRUE","FALSE"))]=="FALSE"){
                  tmp.sigma <- parm[(length(alpha)+length(beta)+1):length(parm)]
                  sigma1 <- tmp.sigma[1]; cov12 <- tmp.sigma[2]; sigma2 <- tmp.sigma[3]
                  rho <- cov12/(sigma1*sigma2)
                  sigma <- c(sigma1, sigma2)
                  } else
if (c("TRUE","FALSE")[charmatch(Cholesky, c("TRUE","FALSE"))]=="TRUE"){
                  L <- parm[(length(alpha)+length(beta)+1):length(parm)]
                  ll1 <- L[1]; l2 <- L[2]; ll3 <- L[3]
                  l1<- exp(ll1); l3<- exp(ll3)
                  sigma <- c(l1, sqrt(l2^2+l3^2))
                  covu1u2 <- l1*l2
                  rho <- covu1u2/(sigma[1]*sigma[2])
                  }
        }
        index.all <- function (subject) {
                   index.all1(subject, id.name, predictor.name, response.name,
                             ordinal.level, data)
                   }
        index.list <- list()
        index.list <- lapply(id,index.all)
        index.list <- rep(index.list, times=nGauss)
        # The same strategy used in index.all() function will be applied to
        # linear.predictor(). This function is defined here because when AdpGHQ='True',
        # The em.bayes() defined later will be needed to call GHQ.intu1().
        # Pay attention to the order of the unknown parameters #
        GHQ.intu1 <- function (subject, random1, random2) {
        linear.predictor (   response.name, predictor.name, id.name, data,
                             time.name, nGauss=NULL, gene.name=NULL, beta.gene=NULL,
               link, model, Adaptive, Cholesky,
                             num.cat, ordinal.level,
                             alpha, beta, sigma, weight, node, new.weight,
                             random1, random2, subject)
        }
        weight <- ghq(n.points=nGauss,modified=FALSE)$weights
        node <- ghq(n.points=nGauss,modified=FALSE)$ zeros
        new.weight <- weight*exp(node^2)
if (c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
    c("Indep Random Coefficient","Random Coefficient",
       "Random Intercept"))] == "Random Intercept"){
    if (c("TRUE","FALSE")[charmatch(Adaptive,c("TRUE","FALSE"))]=="FALSE"){
        abs1 <- sqrt(2)*sigma*node
        u1 <- nodes1 <- matrix(rep(abs1,length(id)),nr=length(id),byrow=T)
        u2 <- nodes2 <- matrix(0,nr=length(id),nc=nGauss)
    } else
    if (c("TRUE","FALSE")[charmatch(Adaptive,c("TRUE","FALSE"))]=="TRUE"){
    em.bayes <- function (subject) {
             em.bayes1 <- function(random,subject) {
                          intu1 <- GHQ.intu1 (subject,random1=random[1],
                                              random2=0)
                          Cals <- pairwise.multiply(index.list[[rank[rank[,
                                                          'id']==subject,1]]], intu1)
             fn <- -log(Cals)+ random[1]^2/(2*sigma^2)
             return(fn)
             }
             em.bayes2 <- function (random) {
                          em.bayes1(random,subject)
             }
             bayes <- list()
             optim.output <- ucminf(0, em.bayes2, hessian=T)
             u.hat <- optim.output$par
             f.sec <- diag(optim.output$hessian)
             u1 <- u.hat[1] + sqrt(2)*(abs(f.sec)^(-1/2))*node
             bayes[[1]] <- u1
             bayes[[2]] <- f.sec
             return(bayes)
             }
             random <- vector(length=1, mode='numeric')
             rank <- data.frame(order(id),id)
             abs.temp <- lapply(id, em.bayes)
             u1 <- u2 <- nodes1 <- nodes2 <- matrix(nc=nGauss, nr=length(id))
             f.sec <- vector(length=length(id), mode='numeric')
             for ( ii in 1:length(id)){
                  u1 [ii,] <- nodes1[ii,] <- abs.temp[[ii]][[1]]
                  f.sec[ii] <- abs.temp[[ii]][[2]]
             }
             u2 <- nodes2 <- matrix(0, nr=length(id),nc=nGauss)
        }
    } else
if (c("TRUE","FALSE")[charmatch(Adaptive,c("TRUE","FALSE"))]=="FALSE"){
            abs1 <- sqrt(2)*sigma[1]*node
            abs2 <- sqrt(2)*sigma[2]*node
            u1 <- nodes1 <- matrix(rep(abs1,length(id)),nr=length(id),byrow=T)
            u2 <- nodes2 <- matrix(rep(abs2,length(id)),nr=length(id),byrow=T)
        } else
if (c("TRUE","FALSE")[charmatch(Adaptive,c("TRUE","FALSE"))]=="TRUE"){
            em.bayes <- function (subject) {
             em.bayes1 <- function(random,subject) {
                          intu1 <- GHQ.intu1 (subject,random1=random[1],
                                              random2=random[2])
                          Cals <- pairwise.multiply(index.list[[rank[rank[,
                                                    'id']==subject,1]]], intu1)
if (c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
    c("Indep Random Coefficient","Random Coefficient",
       "Random Intercept"))] == "Indep Random Coefficient"){
             fn <- -log(Cals)+ random[1]^2/(2*sigma[1]^2) + random[2]^2/(2*sigma[2]^2)
             } else
if (c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
    c("Indep Random Coefficient","Random Coefficient",
       "Random Intercept"))] == "Random Coefficient"){
             fn <- -log(Cals) + 1/(2*(1-rho^2))*(random[1]^2/(sigma[1]^2) +
random[2]^2/(sigma[2]^2)- 2*rho*random[1]*random[2]/sqrt(sigma[1]^2*sigma[2]^2))
             }
             return(fn)
             }
             em.bayes2 <- function (random) {
                          em.bayes1(random,subject)
             }
             bayes <- list()
             optim.output <- ucminf(c(0,0), em.bayes2, hessian=T)
             u.hat <- optim.output$par
             f.sec <- diag(optim.output$hessian)
             u1 <- u.hat[1] + sqrt(2)*(abs(f.sec[1])^(-1/2))*node
             u2 <- u.hat[2] + sqrt(2)*(abs(f.sec[2])^(-1/2))*node
             bayes[[1]] <- data.frame(u1,u2)
             bayes[[2]] <- f.sec
             return(bayes)
             }
             random <- vector(length=2, mode='numeric')
             rank <- data.frame(order(id),id)
             abs.temp <- lapply(id, em.bayes)
             u1 <- u2 <- nodes1 <- nodes2 <- matrix(nc=nGauss, nr=length(id))
             f.sec <- matrix(nr=length(id),nc=2)
             for ( ii in 1:length(id)){
                  u1 [ii,] <- nodes1[ii,] <- abs.temp[[ii]][[1]]$u1
                  u2 [ii,] <- nodes2[ii,] <- abs.temp[[ii]][[1]]$u2
                  f.sec[ii,] <- abs.temp[[ii]][[2]]
            }
         }
# Gauss-Hermite Integration procedure #
               logL.intu1.list  <- list()
               prod2 <- temp <- matrix(nr=length(id), nc=nGauss)
               Gauss.index <- 1:nGauss
               # fn1 is part of GHQ integral on random effect u1 #
if ((c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
     c("Indep Random Coefficient","Random Coefficient",
       "Random Intercept"))] == "Random Coefficient Indep")||
    (c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
     c("Indep Random Coefficient","Random Coefficient",
        "Random Intercept"))] == "Random Intercept")){
                   fn1 <- function (x,u1) {
                          exp(log(x) - (u1^2)/(2*sigma[1]^2))
                   }
               } else
if (c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
    c("Indep Random Coefficient","Random Coefficient",
      "Random Intercept"))] == "Random Coefficient"){
                   fn1 <- function(x,u1,u2) {
                          exp(log(x) - 1/(2*(1-rho^2))*(u1^2/sigma[1]^2+
                   u2^2/sigma[2]^2 - 2*rho*u1*u2/sqrt(sigma[1]^2*sigma[2]^2)))
                   }
                }
               # GHQ integral on both random effects u1,u2 #
               fn2 <- function(Gauss.index) {
                      intu1 <- mapply(GHQ.intu1, id,
                                      random1=u1, u2[,Gauss.index])
                      # Notice:#
                      # intu1 should be a list but when all subject have equal number of #
                      # repeated measurement, it is not a list anymore #
                      if (is.list(intu1)==F) {
                      tmp <- list()
                          for (ii in 1:dim(intu1)[2]){
                          tmp[[ii]] <- matrix(intu1[,ii], nc=num.cat,
                                             nr=length(intu1[,ii])[1]/num.cat)
                          }
                      intu1 <- tmp
                      }
                      Cals <- mapply(pairwise.multiply, index.list, intu1)
                      temp <- matrix(Cals, nr=length(id), nc=nGauss)
if (c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
    c("Indep Random Coefficient","Random Coefficient",
      "Random Intercept"))] == "Random Coefficient"){
                      temp1 <- mapply(fn1, temp, u1, u2[,Gauss.index])
                      } else
                      { temp1 <- mapply(fn1, temp, u1)}
                      temp2 <- matrix(temp1, nc=nGauss, nr=length(id), byrow=F)
                      apply(temp2 %*% new.weight,1,sum)
                      }
if (c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
    c("Indep Random Coefficient","Random Coefficient",
       "Random Intercept"))] == "Random Intercept"){
                      prod2 <- fn2(1)
if (c("TRUE","FALSE")[charmatch(Adaptive,c("TRUE","FALSE"))]=="FALSE"){
                      sum(log(prod2))- length(id)/2*log(pi)
                      } else {
                      sum(log(prod2))- (length(id)*log(sigma^2*pi) + sum(log(abs(f.sec))))/2
                      }
                    } else {
                      prod2 <- mapply(fn2,Gauss.index)
 if(c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
    c("Indep Random Coefficient","Random Coefficient",
      "Random Intercept"))] == "Indep Random Coefficient") {
                   L2  <- exp(log(prod2) - (nodes2^2)/(2*sigma[2]^2))
                   L21 <- apply(L2 %*% new.weight, 1, sum)
if (c("TRUE","FALSE")[charmatch(Adaptive,c("TRUE","FALSE"))]=="FALSE"){
```

```
                  sum(log(L21))-length(id)*log(pi)
                  } else
if (c("TRUE","FALSE")[charmatch(Adaptive,c("TRUE","FALSE"))]=="TRUE"){
                  sum(log(L21))-length(id)*log(pi)-length(id)/2*log(sigma[1]^2*sigma[2]^2)-
                  sum(log(abs(f.sec[,1])))/2-sum(log(abs(f.sec[,2])))/2
                 }
              } else
if (c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
    c("Indep Random Coefficient","Random Coefficient",
       "Random Intercept"))] == "Random Coefficient"){
                  L2  <- exp(log(prod2))
                  L21 <- apply(L2 %*% new.weight, 1, sum)
if (c("TRUE","FALSE")[charmatch(Adaptive,c("TRUE","FALSE"))]=="FALSE"){
                  sum(log(L21))- length(id)*log(pi*sqrt(1-rho^2))
              } else
if (c("TRUE","FALSE")[charmatch(Adaptive,c("TRUE","FALSE"))]=="TRUE"){
sum(log(L21))-length(id)*log(pi)-length(id)/2*log(sigma[1]^2*
sigma[2]^2*(1-rho^2))- sum(log(abs(f.sec[,1])))/2-
sum(log(abs(f.sec[,2])))/2
              }
          }
       }
     }
#####################################################################################
# 7. This function onstructs the starting value parm0 #
# The starting value for cumulative logit would be slightly different from others
# with alpha1 < alpha2 <....alpha(c-1) #
initial.value <- function (data, response.name,predictor.name,
                          link=c("Cumulative", "Adjacent", "Forward CR", "Backward CR"),
                          model=c("Random Intercept", "Random Coefficient",
              "Indep Random Coefficient")){
              link <- match.arg(link)
                          model <- match.arg(model)
                          response <- data[,response.name]
                          num.cat <-  nlevels(as.factor(response))
if (c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
    c("Indep Random Coefficient","Random Coefficient",
      "Random Intercept"))] == "Random Intercept") {
    if (c("Cumulative","Adjacent","Forward CR","Backward CR")[charmatch(link,
    c("Cumulative","Adjacent","Forward CR","Backward CR"))] == "Cumulative"){
                a0 <- -1
                alpha0 <- vector(length=(num.cat-1), mode="numeric")
                for (ii in 1: (num.cat-1)){
                    alpha0[ii] <- a0+(ii-1)
                }
                beta0  <- rep(0, length(predictor.name))
                parm0 <- c(alpha0, beta0, 1)
             } else
                alpha0 <- rep(0, num.cat-1)
                beta0  <- rep(0, length(predictor.name))
                parm0 <- c(alpha0, beta0, 1)
             } else
```

```
if (c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
    c("Indep Random Coefficient","Random Coefficient",
      "Random Intercept"))] == "Indep Random Coefficient") {
if (c("Cumulative","Adjacent","Forward CR","Backward CR")[charmatch(link,
    c("Cumulative","Adjacent","Forward CR","Backward CR"))] == "Cumulative"){
                 a0 <- -1
                 alpha0 <- vector(length=(num.cat-1), mode="numeric")
                 for (ii in 1: (num.cat-1)){
                     alpha0[ii] <- a0+(ii-1)
                 }
                 beta0  <- rep(0, length(predictor.name))
                 parm0 <- c(alpha0, beta0, 1, 1)
              } else
                 alpha0 <- rep(0, num.cat-1)
                 beta0  <- rep(0, length(predictor.name))
                 parm0 <- c(alpha0, beta0, 1, 1)
              } else
if (c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
    c("Indep Random Coefficient","Random Coefficient",
       "Random Intercept"))] == "Random Coefficient") {
if (c("Cumulative","Adjacent","Forward CR","Backward CR")[charmatch(link,
    c("Cumulative","Adjacent","Forward CR","Backward CR"))] == "Cumulative"){
                 a0 <- -1
                 alpha0 <- vector(length=(num.cat-1), mode="numeric")
                 for (ii in 1: (num.cat-1)){
                     alpha0[ii] <- a0+(ii-1)
                 }
                 beta0  <- rep(0, length(predictor.name))
                 parm0 <- c(alpha0, beta0, 1, 0, 1)
              } else
                 alpha0 <- rep(0, num.cat-1)
                 beta0  <- rep(0, length(predictor.name))
                 parm0 <- c(alpha0, beta0, 1, 0, 1)
              }
           }
##############################################################################
# 8.To use the ordinal.model() function #
# ordinal.mixed.model <- function(response.name, predictor.name, id.name, time.name,
#                                 data, beta.gene, gene.name, nGauss,link,model,
                                   Adaptive, Cholesky){
         # check if predictors are numeric #
# ordinal.mixed.model(response.name, predictor.name, id.name, time.name, data)
ordinal.mixed.model <- function(response.name, predictor.name, id.name, time.name,
                                data, nGauss=NULL, gene.name=NULL, beta.gene=NULL,
 link=c("Cumulative", "Adjacent", "Forward CR", "Backward CR"),
                model=c("Random Intercept", "Random Coefficient",
   "Indep Random Coefficient"), Adaptive=c("TRUE", "FALSE"),
 Cholesky=c("TRUE", "FALSE"),...){
                                link <- match.arg(link)
                                model <- match.arg(model)
                                Adaptive <- match.arg(Adaptive)
                                Cholesky <- match.arg(Cholesky)
                                data <- data.matrix(data)
        if (is.null(time.name)==TRUE){
            time.name==predictor.name[1]
            }
    if(is.null(nGauss)==TRUE) {nGauss <-10}
         parm <- initial.value(data, response.name,predictor.name, link, model)
        # Possible error messages and solutions #
        # 1. Force Cholesky="TRUE" if random coefficient model #
if ((c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
     c("Indep Random Coefficient","Random Coefficient",
       "Random Intercept"))] == "Random Coefficient")&& (
     c("TRUE","FALSE")[charmatch(Cholesky, c("TRUE","FALSE"))]=="FALSE")) {
     cat("Note: Cholesky Decomposition is required for fitting Random Coefficient model \n" )
     cat("Cholesky option automatically switched to TRUE \n\n")
     Cholesky = "TRUE"
     }
       # 2. Recommend but not force Adaptive="TRUE" if random coefficient model #
 if ((c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
      c("Indep Random Coefficient","Random Coefficient",
        "Random Intercept"))] == "Random Coefficient")&& (
      c("TRUE","FALSE")[charmatch(Adaptive, c("TRUE","FALSE"))]=="FALSE")) {
      cat("Caution:Adaptive Gauss-Hermite Quadrature is recommended for
            Random Coefficient model\n\n")
     }
     ordinal.model1 <- function(parm){
                      ordinal.model(response.name, predictor.name, id.name,
                                   time.name, data, nGauss, gene.name,
                                   beta.gene, link, model, Adaptive, Cholesky,
                                   parm)
                      }
 # ordinal.model1(parm)
 #}
 # ordinal.mixed.model(response.name, predictor.name, id.name, time.name,
     #                  data, link="Adjacent", nGauss=20)
# Specify the degree of freedom #
if (c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
    c("Indep Random Coefficient","Random Coefficient",
       "Random Intercept"))] == "Random Intercept"){
           df <- length(unique(data[,id.name]))- 1
           } else {
           df <- length(unique(data[,id.name]))- 2
           }
# Optimization procedure: for all "Cumulative" link, we use optimization function 'optim' #
# while for all other links, we use "optimx" function. "optimx" function is less sensitive #
# to the initial values of parameters compared with "optim". However, "optimx" does not work #
# so well for "Cumlative" link potentially because of the constraint on the intercepts #
final.outcome <- list()
if (c("Cumulative","Adjacent","Forward CR","Backward CR")[charmatch(link,
    c("Cumulative","Adjacent","Forward CR","Backward CR"))] == "Cumulative"){
           output <- optim(parm, ordinal.model1, method="L-BFGS-B",
                     control=list(fnscale=-10,trace=5),hessian=TRUE)
            if (c("TRUE","FALSE")[charmatch(Cholesky, c("TRUE","FALSE"))]=="TRUE"){
                result <- chol2var(data, response.name, predictor.name,
                   output$par, output$hessian, df, model)
            } else
            if (c("TRUE","FALSE")[charmatch(Cholesky, c("TRUE","FALSE"))]=="FALSE"){
                result <- se2var(data, response.name, predictor.name,
                 output$par, output$hessian, df, model)
            }
            final.outcome[[1]] <- result
            final.outcome[[2]] <- output$value
    }  else {
            output <- optimx(parm, ordinal.model1, method="BFGS",
                       control=list(fnscale=-10,trace=5, kkt=FALSE),hessian=TRUE)
            output1 <- attributes(output)
            if (c("TRUE","FALSE")[charmatch(Cholesky, c("TRUE","FALSE"))]=="TRUE"){
                result <- chol2var(data, response.name, predictor.name,
                                   parm= output1$details[[1]]$par,
                                   hessian= output1$details[[1]]$nhatend,
                           df, model)
            } else
            if (c("TRUE","FALSE")[charmatch(Cholesky, c("TRUE","FALSE"))]=="FALSE"){
                result <- se2var(data, response.name, predictor.name,
                                 parm=output1$details[[1]]$par,
                                 hessian= output1$details[[1]]$nhatend,
                         df, model)
            }
            final.outcome[[1]] <- result
            final.outcome[[2]] <- output1$details[[1]]$value
         }
       return(final.outcome)
 }
### END ordinalmixedv2.R ###
###########################################################################
###########################################################################
### START Compile_step3.R ###
pi <- atan(1)*4
temp <- temp1 <- temp2<- tmp <- NULL
linear.predictor.Cum <- function(response.name, predictor.name, id.name, data,
                                 time.name,  num.cat, ordinal.level, alpha,
                                 beta, sigma, random1, random2, subject,
     model=c("Random Intercept", "Random Coefficient", "Indep Random Coefficient"),
                                 gene.name=NULL, beta.gene=NULL,...){
                         model <- match.arg(model)
                         sub.matrix <- data.matrix(data[data[,id.name]==subject,
                                            c(predictor.name, response.name)])
                        if (dim(sub.matrix)[2]==1){
                            sub.matrix <- t(sub.matrix)
                            }
if (is.null(beta.gene)==F){
                        xgene <- data.matrix(data[data[,id.name]==subject, gene.name])
                    }
          if(is.null(time.name)==TRUE){
                       time.name <- predictor.name[1]
                   }
          alpha1 <- c(alpha, 500)
      if (is.null(beta.gene)==T) {
          xbeta <- matrix(rep((sub.matrix[, predictor.name] %*% as.matrix(beta) +
                 random1 + random2*sub.matrix[,time.name]), num.cat), nc=num.cat)
      } else if (is.null(beta.gene)==F) {
        xbeta <- matrix(rep(sub.matrix[, predictor.name] %*% as.matrix(beta) +
                 random1 + random2*sub.matrix[,time.name]+xgene %*% beta.gene,
         num.cat), nc=num.cat)
        }
       eta <- sweep(xbeta, 2, alpha1, "+")
         cumulative.p <- t(G(eta))
         mat <- diag(1,num.cat)
         mat1<- cbind(diag(-1,num.cat-1), (rep(0,num.cat-1)))
         mat2 <- rbind(rep(0,num.cat), mat1)
         trans.mat <- mat + mat2
         p <- t(trans.mat %*% cumulative.p)
         return(p)
         }
################################################################################
ConditionalMode <- function(response.name, predictor.name, id.name, data,
                            time.name, parm, gene.name=NULL, beta.gene=NULL,
model=c("Random Intercept", "Random Coefficient", "Indep Random Coefficient")){
                   model <- match.arg(model)
                   response <- data[,response.name]
                   predictor <- data[,predictor.name]
                   id <- as.numeric(unique(data[,id.name]))
                   num.cat <-  nlevels(as.factor(response))
                   ordinal.level <- as.numeric(levels(as.factor(response)))
                   # Define random effects #
                   random1 <- 0; random2 <- 0
                   # Define ordinal model parameters #
                   alpha <- vector(length=num.cat-1, mode='numeric')
                   alpha <- parm[1:length(alpha)]
                   beta  <- vector(length=length(predictor.name), mode='numeric')
                   beta  <- parm[(length(alpha)+1):(length(alpha)+length(beta))]
if (c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
    c("Indep Random Coefficient","Random Coefficient",
       "Random Intercept"))] == "Random Intercept"){
                   sigma <- parm[(length(alpha)+length(beta)+1):length(parm)]
                   } else
if  (c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
     c("Indep Random Coefficient","Random Coefficient",
        "Random Intercept"))] == "Random Coefficient"){
                   sigma <- parm[(length(alpha)+length(beta)+1):length(parm)]
                   sigma1 <- sigma[1]
                   sigma2 <- sigma[3]
                   cov <- sigma[2]
                   rho <- cov/sqrt(sigma1*sigma2)
                   } else
if  (c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
     c("Indep Random Coefficient","Random Coefficient",
        "Random Intercept"))] == "Indep Random Coefficient"){
```

```
                  sigma <- parm[(length(alpha)+length(beta)+1):length(parm)]
                  sigma1 <- sigma[1]
                  sigma2 <- sigma[2]
                  }
        index.all <- function (subject) {
                    index.all1(subject, id.name, predictor.name, response.name,
                               ordinal.level, data)
                    }
        index.list <- list()
        index.list <- lapply(id,index.all)
        nGauss=1
        index.list <- rep(index.list, times=nGauss)
# Define function to calculate condition mode for different models #
# Function 1. Random Intercept Model #
   ConditionMode.Int <- function(subject){
        ConditionMode1 <- function(random1){
        p <- linear.predictor.Cum(response.name, predictor.name, id.name, data,
                                  time.name,  num.cat, ordinal.level, alpha,
                                  beta, sigma, random1, random2=0, subject)
                     logp <- log(p)
                     id <- as.numeric(unique(data[,id.name]))
                     rank <- data.frame(1:length(id),id)
                     part1 <- sum(pairwise.multiply1(index.list[[rank[rank[,
                                  'id']==subject,1]]], logp))
                     part2 <- 1/2*log(2*pi*sigma)+random1^2/(2*sigma)
                     nlogh <- part2 - part1
                     return(nlogh)
                     }
         optim(0, ConditionMode1, method="L-BFGS-B")$par
        }
# Function 2. Random Coefficient Model #
ConditionMode.Coef <- function(subject){
        ConditionMode2 <- function(random){
                     random1 <- random[1]
                     random2 <- random[2]
                p <- linear.predictor.Cum(response.name, predictor.name, id.name, data,
                                          time.name,  num.cat, ordinal.level, alpha,
                                          beta, sigma, random1, random2, subject)
                     logp <- log(p)
                     id <- as.numeric(unique(data[,id.name]))
                     rank <- data.frame(1:length(id),id)
                     part1 <- sum(pairwise.multiply1(index.list[[rank[rank[,'id']==
                                                                      subject,1]]], logp))
                     part2 <- log(2*pi*sqrt(sigma1*sigma2)) + 1/2*log(1-rho^2)+
                              1/(2*(1-rho^2))*(random1^2/sigma1 + random2^2/sigma2 -
                              2*rho*random1*random2/sqrt(sigma1*sigma2))
                     nlogh <- part2 - part1
                     return(nlogh)
                     }
         optim(c(0,0), ConditionMode2, method="L-BFGS-B")$par
        }
# Function 3.#
ConditionMode.Coef0 <- function(subject){
        ConditionMode2 <- function(random){
                     random1 <- random[1]
                     random2 <- random[2]
        p <- linear.predictor.Cum(response.name, predictor.name, id.name, data,
                                  time.name,  num.cat, ordinal.level, alpha,
                                  beta, sigma, random1, random2, subject)
                     logp <- log(p)
                     id <- as.numeric(unique(data[,id.name]))
                     rank <- data.frame(1:length(id),id)
                     part1 <- sum(pairwise.multiply1(index.list[[rank[rank[,'id']==
                                                                      subject,1]]],  logp))
                     rho <- 0
                     part2 <- log(2*pi*sqrt(sigma1*sigma2)) + 1/2*log(1-rho^2)+
                     1/(2*(1-rho^2))*(random1^2/sigma1 + random2^2/sigma2 -
                     2*rho*random1*random2/sqrt(sigma1*sigma2))
                     nlogh <- part2 - part1
                     return(nlogh)
                     }
         optim(c(0,0), ConditionMode2, method="L-BFGS-B")$par
        }
# Function 4. Prediction from Random Intercept Model #
predict1 <- function(subject, random1){
         p <- linear.predictor.Cum(response.name, predictor.name, id.name, data,
                                  time.name,  num.cat, ordinal.level, alpha,
                                  beta, sigma, random1, random2=0, subject)
     apply(p, 1, which.max)
     }
# Function 5. Prediction from Random Coefficient Model #
predict2 <- function(subject, random1, random2){
  p <- linear.predictor.Cum(response.name, predictor.name, id.name, data,
                            time.name,  num.cat, ordinal.level, alpha,
                            beta, sigma, random1, random2, subject)
     apply(p, 1, which.max)
     }
if (c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
    c("Indep Random Coefficient","Random Coefficient",
       "Random Intercept"))] == "Random Intercept"){
        u.mode <- mapply(ConditionMode.Int,id)
        pred.cat <- unlist(mapply(predict1, id, u.mode))
        } else
if (c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
    c("Indep Random Coefficient","Random Coefficient",
       "Random Intercept"))] == "Random Coefficient"){
        u.mode <- t(mapply(ConditionMode.Coef,id))
        pred.cat <- unlist(mapply(predict2, id, u.mode[,1],u.mode[,2]))
        } else
if (c("Indep Random Coefficient","Random Coefficient","Random Intercept")[charmatch(model,
    c("Indep Random Coefficient","Random Coefficient",
       "Random Intercept"))] == "Indep Random Coefficient"){
         u.mode <- t(mapply(ConditionMode.Coef0,id))
         pred.cat <- unlist(mapply(predict2, id, u.mode[,1],u.mode[,2]))
```

```
          }
if (is.matrix(pred.cat)==TRUE) {
    pred.cat <- as.vector(pred.cat)
    }
return(pred.cat)
}
### END Compile_step3.R ###
###########################################################################
###########################################################################
### "FSPenFixed" function ###
FSPenFixed <- function(x, y, data,
                             gene.name=NULL, beta.name=NULL,
                             epsilon=1e-4, tol=1e-6, cut.prop=0.002,
                             criteria="AIC", standardize=FALSE){
source("Compile_step1.R")
source("Compile_step2.R")
source("Compile_step3.R")
if (standardize==TRUE) {x <- scale(x)
                        data <- data.frame(y, x)
                       }
y <- as.ordered(y)
forwardstagewise <- forward.stagewise.cum (x, y, epsilon, tol, cut.prop)
output<- output.CumFS(forwardstagewise, x, y)
step.index <- steps(output, range=0, criteria)
beta.list <- forwardstagewise$beta.list
Num.nonzero <- unlist(forwardstagewise$num.nonzero)
new.beta <- beta.gene <- beta.list[[step.index]]
gene.name <- colnames(x)[as.numeric(names(new.beta))]
fit <- logLL(x, y, beta1=new.beta)
new.alpha <- fit$par
logLL <- fit$value
pred.y <- Predict(x, y, new.alpha, new.beta)
ylevels <- levels(as.ordered(y))
kk <- 1:length(ylevels)
         for (ii in 1:length(pred.y)){
              for (kk in 1:length(ylevels)){
                   if (pred.y[ii]==kk) {pred.y[ii] <- ylevels[kk]}
                   }
              }
pred.y <- factor(pred.y, levels=levels(y),ordered=TRUE)
pred.table <- table(pred.y, y)
list(alpha=new.alpha, beta.gene=beta.gene, gene.name=gene.name,
     logLL=logLL, pred.table=pred.table)
}
### END "FSPenFixed" function ###
#############################################################################
#############################################################################
### START "FSPenFixedCV" function ###
FSPenFixedCV <- function(x, y, data, kfold,
                         gene.name=NULL, beta.name=NULL,
                         epsilon=1e-4, tol=1e-6, cut.prop=0.002,
                         criteria="AIC", standardize=FALSE){
source("Compile_step1.R")
source("Compile_step2.R")
source("Compile_step3.R")
if (standardize==TRUE) {x <- scale(x)
                        data <- data.frame(y, x)
                       }
y <- as.ordered(y)
PenFixedCV <- list()
for (jj in 1:kfold){
     test.index <- (as.integer((jj-1)/kfold*dim(x)[1])+1):
                   (as.integer(jj/kfold*dim(x)[1]))
     train.index <- setdiff(1:dim(x)[1], test.index)
     index <- grep(colnames(x)[1], colnames(data))
     test.data <- data[test.index,]
     train.data <- data[train.index,]
     test.x <- data.matrix(test.data[,index:dim(data)[2]])
     train.x <- data.matrix(train.data[,index:dim(data)[2]])
     test.y <- y[test.index]
     train.y <- y[train.index]
     forwardstagewise <- forward.stagewise.cum (train.x, train.y,
                                              epsilon, tol, cut.prop)
     output<- output.CumFS(forwardstagewise, train.x, train.y)
     step.index <- steps(output, range=0, criteria)
     beta.list <- forwardstagewise$beta.list
     Num.nonzero <- unlist(forwardstagewise$num.nonzero)
     new.beta <- beta.gene <- beta.list[[step.index]]
     gene.name <- colnames(x)[as.numeric(names(new.beta))]
     fit <- logLL(train.x, train.y, beta1=new.beta)
     new.alpha <- fit$par
     pred.y <- Predict(x, y, new.alpha, new.beta)
     ylevels <- levels(as.ordered(y))
     kk <- 1:length(ylevels)
         for (ii in 1:length(pred.y)){
              for (kk in 1:length(ylevels)){
                   if (pred.y[ii]==kk) {pred.y[ii] <- ylevels[kk]}
                   }
              }
     pred.y <- factor(pred.y, levels=levels(y),ordered=TRUE)
     pred.table <- table(pred.y, y)
     PenFixedCV[[jj]] <- list(test.index=test.index, new.beta=new.beta,
                         gene.name=gene.name, pred.table=pred.table)
     }
return(PenFixedCV)
}
### END "FSPenFixedCV" function ###
#############################################################################
#############################################################################
### START "FSPenMixed function ###
FSPenMixed <- function(response.name, predictor.name,
                       id.name, time.name, data, nGauss=NULL,
                       gene.name=NULL, beta.name=NULL,
 link=c("Cumulative", "Adjacent", "Forward CR", "Backward CR"),
  model=c("Random Intercept", "Random Coefficient",
   "Indep Random Coefficient"), Adaptive=c("TRUE", "FALSE"),
 Cholesky=c("TRUE", "FALSE"), x,y,
 epsilon=1e-4, tol=1e-3, cut.prop=0.0015,
 range=0, criteria="AIC", standardize=FALSE){
### load functions ###
source("Compile_step1.R")
source("Compile_step2.R")
source("Compile_step3.R")
model <- match.arg(model)
link <- match.arg(link)
Adaptive <- match.arg(Adaptive)
Cholesky <- match.arg(Cholesky)
if (is.null(nGauss)==TRUE) {nGauss=10}
if (standardize==TRUE) {x <- scale(x)
                        data1 <- scale(data[, c(predictor.name, time.name)])
data <- data.frame(data[,c(response.name, id.name)], data1, x)
} #what if time.name=NULL???#
### List of Parameters ###
# x: matrix contained the probes or genes                             #
# y: ordinal response                                                  #
# epsilon: incremental amount in the forward stagewise.Default=1e-4 #
# tol: converge criteria. Default=1e-3                                 #
# cut.prop: proportion of features assumed to be important.           #
# range:  number of random coef ordinal models fitted. Default=1      #
# criteria: AIC or BIC. Default="AIC"                                  #
# data: the data used.                                                 #
# response.name: name of ordinal response                              #
# predictor.name: fixed effect, exclude the features selected by FS #
# id.name: subject name                                                #
# time.name: time                                                      #
# nGauss: number of Gauss-Hermite Quadrature points. Default=10       #
# link: Type of random coef ordinal models. Default="Cumulative"      #
# model: "Random Intercept", "Random Coefficient"                      #
# Adaptive: Adaptive GHQ procedure. Default=TRUE                       #
# Cholesky: Cholesky Decomposition for the variance. Default=TRUE     #
### Conduct the Forward Stagewise Procedure ###
y <- as.ordered(data[,response.name])
forwardstagewise <- forward.stagewise.cum (x, y, epsilon, tol, cut.prop)
beta.list <- forwardstagewise$beta.list
Num.nonzero <- unlist(forwardstagewise$num.nonzero)
output<- output.CumFS(forwardstagewise, x=x, y=y)
step.index <- steps(output, range, criteria)
PenMixed <- list()
for (jj in 1:length(step.index)) {
     new.beta <- beta.gene <- beta.list[[step.index[jj]]]
     gene.name <- colnames(x)[as.numeric(names(new.beta))]
  fit <- ordinal.mixed.model(response.name, predictor.name, id.name,
                             time.name, data, nGauss, gene.name,
                             beta.gene, link, model, Adaptive, Cholesky)
         parm <- fit[[1]]$parm
         logLL <- fit[[2]]
         aic <- 2*length(new.beta) - 2*logLL
         bic <- length(new.beta)*log(dim(x)[2]) - 2*logLL
         pred.y <-  ConditionalMode(response.name, predictor.name, id.name,
                                   data, time.name, parm, gene.name, beta.gene,
                                   model)
         ylevels <- levels(y)
         kk <- 1:length(ylevels)
         for (ii in 1:length(pred.y)){
              for (kk in 1:length(ylevels)){
                   if (pred.y[ii]==kk) {pred.y[ii] <- ylevels[kk]}
                   }
              }
         pred.table <- table(pred.y, y)
         PenMixed[[jj]] <- list(parm=parm, new.beta=new.beta, gene.name=gene.name,
                              AIC=aic, BIC=bic, logLL=logLL, pred.table=pred.table)
        }
return(PenMixed)
}
### END "FSPenMixed" function ###
##############################################################################
##############################################################################
### START "FSPenMixedCV" function ###
FSPenMixedCV <- function(response.name, predictor.name,
                       id.name, time.name, data, kfold=1, loop=20,
                       nGauss=NULL, gene.name=NULL, beta.name=NULL,
           link=c("Cumulative", "Adjacent", "Forward CR", "Backward CR"),
                       model=c("Random Intercept", "Random Coefficient",
      "Indep Random Coefficient"), Adaptive=c("TRUE", "FALSE"),
     Cholesky=c("TRUE", "FALSE"), x, y,
     epsilon=1e-4, tol=1e-3, cut.prop=0.0014,
    range=0, criteria="AIC", standardize=FALSE){
### load functions ###
source("Compile_step1.R")
source("Compile_step2.R")
source("Compile_step3.R")
model <- match.arg(model)
link <- match.arg(link)
Adaptive <- match.arg(Adaptive)
Cholesky <- match.arg(Cholesky)
if (is.null(nGauss)==TRUE) {nGauss=10}
if (standardize==TRUE) {x <- scale(x)
                        data1 <- scale(data[, c(predictor.name, time.name)])
                        data <- data.frame(data[,c(response.name, id.name)],
                                         data1, x)
}
y <- as.ordered(y)
N <- length(unique(data[,id.name]))
id <- unique(data[,id.name])
PenMixedCV <- list()
for (jj in 1:kfold){
               test.subject <-  as.integer((jj-1)/kfold*N+1):
                                (as.integer(jj/kfold*N))
               test.index <- as.matrix(id[test.subject])
               train.subject <- setdiff(1:N, test.subject)
               train.index <- as.matrix(id[train.subject])
```

```
                colnames(test.index)<-"test.index"
                test.data<-merge(test.index, data, by.x="test.index",
                                 by.y=id.name, all.x=T)
                colnames(test.data)[1] <- id.name
                train.index <- as.matrix(setdiff(id, test.index))
                colnames(train.index)<-"train.index"
                train.data<-merge(train.index, data, by.x="train.index",
                                  by.y=id.name, all.x=T)
                colnames(train.data)[1] <- id.name
                index <- grep(colnames(x)[1], colnames(data))
                test.x <- data.matrix(test.data[,index:dim(data)[2]])
                train.x <- data.matrix(train.data[,index:dim(data)[2]])
if ( nlevels(as.ordered(train.data[, response.name]))!=
     nlevels(as.ordered(data[,response.name]))) { next }
### Conduct the Forward Stagewise Procedure ###
train.y <- as.ordered(train.data[,response.name])
test.y <- test.data[, c(id.name, response.name)]
forwardstagewise <- forward.stagewise.cum (x=train.x, y=train.y,
                                           epsilon, tol, cut.prop)
beta.list <- forwardstagewise$beta.list
Num.nonzero <- unlist(forwardstagewise$num.nonzero)
output<- output.CumFS(forwardstagewise, x=train.x, y=train.y)
step.index <- steps(output, range, criteria)
     new.beta <- beta.gene <- beta.list[[step.index]]
     gene.name <- colnames(x)[as.numeric(names(new.beta))]
     fit <- ordinal.mixed.model(response.name, predictor.name, id.name,
                                time.name, data=train.data, nGauss, gene.name,
                                beta.gene, link, model, Adaptive, Cholesky)
         parm <- fit[[1]]$parm
         logLL <- fit[[2]]
         aic <- 2*length(new.beta) - 2*logLL
         bic <- length(new.beta)*log(dim(x)[2]) - 2*logLL
         pred.y <-  ConditionalMode(response.name, predictor.name, id.name,
                                    data, time.name, parm, gene.name, beta.gene,
                                    model)
         ylevels <- levels(as.ordered(y))
         kk <- 1:length(ylevels)
         for (ii in 1:length(pred.y)){
              for (kk in 1:length(ylevels)){
                   if (pred.y[ii]==kk) {pred.y[ii] <- ylevels[kk]}
                   }
              }
         pred.y <- factor(pred.y, levels=levels(y), ordered=T)
         pred.table <- table(pred.y, y)
         PenMixedCV[[jj]] <- list(test.index=test.index, new.beta=new.beta,
                                  gene.name=gene.name, pred.table=pred.table)
         }
return(PenMixedCV)
}
### END "FSPenMixedCV" function ###
###############################################################################
```

I.2 Application to NIMH Schizophrenia Longitudinal Data

```
# Import the function and data #
source("ordinalmixedv2.R")
data <- read.csv("NIMH Schizophrenia.csv")
response.name = "imps79o"
predictor.name= c('tx','sweek','txswk')
id.name = "id"
time.name = 'sweek'
# Ordinal random intercept and random coefficient models with cumulative logit #
fit1 <- ordinal.mixed.model (response.name, predictor.name, id.name, time.name,
                                              data, nGauss=15)
fit2 <- ordinal.mixed.model (response.name, predictor.name, id.name,  time.name, data,
                                        gene.name=NULL, beta.gene=NULL,
                                        model="Random Coefficient")
# Ordinal random intercept and random coefficient models with adjacent category #
fit1 <- ordinal.mixed.model (response.name, predictor.name, id.name,  time.name, data,
                                         gene.name=NULL, beta.gene=NULL,
                                        link="Adjacent", nGauss=5)
fit2 <- ordinal.mixed.model (response.name, predictor.name, id.name, time.name,
                                              data, gene.name=NULL, beta.gene=NULL,
                                         model="Random Coefficient", link="Adjacent")
# Ordinal random intercept and random coefficient models with forward continuation ratio #
fit1 <- ordinal.mixed.model (response.name, predictor.name, id.name, time.name,
                                              data, link="Forward")
fit2 <- ordinal.mixed.model (response.name, predictor.name, id.name, time.name, data,
                                         model="Random Coefficient", link="Forward")
# Ordinal random intercept and random coefficient models with backward continuation ratio #
fit1 <- ordinal.mixed.model (response.name, predictor.name, id.name, time.name,
                                              data, link="Backward")
fit2 <- ordinal.mixed.model (response.name, predictor.name, id.name, time.name, data,
                                         model="Random Coefficient", link="Backward")
```

I.3 Application to Health Service Research Example

```
# Import the function and data #
source("ordinalmixedv2.R")
data <- read.csv("San Diego Homeless.csv")
response.name = "Housing"
predictor.name= c("Time1","Time2","Time3", "Section8", "Sect8T1", "Sect8T2","Sect8T3")
id.name = "Id"
time.name = NULL
# Ordinal random intercept and random coefficient models with cumulative logit #
fit1 <- ordinal.mixed.model (response.name, predictor.name, id.name, time.name,
                                             data, nGauss=15)
fit2 <- ordinal.mixed.model (response.name, predictor.name, id.name,  time.name, data,
                                      gene.name=NULL, beta.gene=NULL,
                                      model="Random Coefficient")
# Ordinal random intercept and random coefficient models with adjacent category #
fit1 <- ordinal.mixed.model (response.name, predictor.name, id.name,  time.name, data,
                                       gene.name=NULL, beta.gene=NULL, link="Adjacent",
                                      nGauss=5)
fit2 <- ordinal.mixed.model (response.name, predictor.name, id.name, time.name, data,
                                      gene.name=NULL,  beta.gene=NULL,
                                     model="Random Coefficient", link="Adjacent")
# Ordinal random intercept and random coefficient models with forward continuation ratio #
fit1 <- ordinal.mixed.model (response.name, predictor.name, id.name, time.name, data,
                                             link="Forward")
fit2 <- ordinal.mixed.model (response.name, predictor.name, id.name, time.name, data,
                                       model="Random Coefficient", link="Forward")
# Ordinal random intercept and random coefficient models with backward continuation ratio #
fit1 <- ordinal.mixed.model (response.name, predictor.name, id.name, time.name, data,
                                             link="Backward")
fit2 <- ordinal.mixed.model (response.name, predictor.name, id.name, time.name, data,
                                       model="Random Coefficient", link="Backward")
```

I.4 Application to GSE10006 Smoking Study

```
### Model fitting ###
source("FSPenFixed.R")
data <- read.csv("GSE10006.csv")
x=data[,3:dim(data)[2]]
y=data[,2]
y <-factor(y, levels=c("non-smoker","smoker","early-COPD","COPD"), ordered=TRUE)
output <- FSPenFixed(x, y, data, tol=1e-5)
##############################################################################
### Visualization ###
source("Compile_step1.R")
source("Compile_step2.R")
data <- read.csv("GSE10006.csv")
x=data[,3:dim(data)[2]]
y=data[,2]
y <-factor(y, levels=c("non-smoker","smoker","early-COPD","COPD"),ordered=TRUE)
fit <- forward.stagewise.cum(x, y, 1e-4,  1e-4, 1)
summary <- output.CumFS(fit, x, y)
beta <- fit$beta.list
step <- length(beta)
# full list of genes consecutively enters the active set #
add.gene <- list()
for (ii in 2:dim(summary)[1]){
    old.gene <- names(beta[[summary[ii-1,"step"]]])
    new.gene <- names(beta[[summary[ii,"step"]]])
    add.gene[[ii]] <- setdiff(new.gene, old.gene)
    add.gene[[1]] <- names(beta[[summary[1,"step"]]])
    }
step <- summary[60,"step"]
plot(0,0,type="n", xlim=c(0,step),
     ylim=c(min(beta[[step]]),max(beta[[step]])),
     xlab="Step", ylab="Penalized Estimate",cex.lab=1.3,
     main="GSE10006 Forward Stagewise Path", cex.main=1.3)
lines(x=c(0,step),y=c(0,0), type="l", col=1, lwd=2)
for (ii in 1:60){
gene <- add.gene[[ii]]
for (kk in ii:dim(summary)[1]) {
if (kk==1){
    lines(x=c(0,summary[kk,"step"]), y=c(0, beta[[summary[kk,"step"]]]),col=2,lwd=1.5)
} else if (kk==ii){
    lines(x=c(summary[kk-1,"step"],summary[kk,"step"]),
          y=c(0,beta[[summary[kk,"step"]]][names(beta[[summary[kk,"step"]]])==gene]),
                 col=ii+1, lwd=1.5)
} else {
lines(x=c(summary[kk-1,"step"],summary[kk,"step"]),
      y=c(beta[[summary[kk-1,"step"]]][names(beta[[summary[kk-1,"step"]]])==gene],
          beta[[summary[kk,"step"]]][names(beta[[summary[kk,"step"]]])==gene]),
          col=ii+1,lwd=1.5)
      }
}
}
```

```
lines(x=c(15965, 15965), y=c(-1,1),col=1,lty=2)
plot(summary[, "step"],  summary[,"aic"], type="b", xlab="step", ylab="AIC", pch=16)
###########################################################################
### N-fold cross-validation ###
source("FSPenFixedCV.R")
data <- read.csv("GSE10006.csv")
x=data[,3:dim(data)[2]]
y=data[,2]
y <-factor(y, levels=c("non-smoker","smoker","early-COPD","COPD"),ordered=TRUE)
FSPen <- FSPenFixedCV(x, y, data, kfold=dim(x)[1])
### Gathering cross-validation together ###
# Define Goodman and Kruskal's gamma #
Gamma.stat4 <- function(tables) {
              c11 <- tables[1,1]*(sum(tables[2:4,2:4]))
              c12 <- tables[1,2]* sum(tables[2:4,3:4])
              c13 <- tables[1,3]* sum(tables[2:4,4])
              c21 <- tables[2,1]* sum(tables[3:4,2:4])
              c22 <- tables[2,2]* sum(tables[2:4,2:4])
              c23 <- tables[2,3]* sum(tables[3:4,4])
              c31 <- tables[3,1]* sum(tables[4,2:4])
              c32 <- tables[3,2]* sum(tables[4,3:4])
              c33 <- tables[3,3]* sum(tables[4,4])
              C <- c11 + c12 + c13 + c21 + c22 + c23 + c31 + c32 + c33
              d12 <- tables[1,2]*sum(tables[2:4,1])
              d13 <- tables[1,3]*sum(tables[2:4,1:2])
              d14 <- tables[1,4]*sum(tables[2:4,1:3])
              d22 <- tables[2,2]*sum(tables[3:4,1])
              d23 <- tables[2,3]*sum(tables[3:4,1:2])
              d24 <- tables[2,4]*sum(tables[3:4,1:3])
              d32 <- tables[3,2]*sum(tables[4,1])
              d33 <- tables[3,3]*sum(tables[4,1:2])
              d34 <- tables[3,4]*sum(tables[4,1:3])
              D <- d12 + d13 + d14 + d22 + d23 + d24 + d32 + d33 + d34
    gamma.stat <- (C-D)/(C+D)
    return(gamma.stat)
}
# kk varies according to the number of split in k-fold cv #
gene.freq <- predaccuracy <- gamma <- NULL
for(kk in 1:48){
     gene.freq <- c(gene.freq, FSPen[[kk]]$gene.name)
     predaccuracy <- c(predaccuracy,
     sum(diag(FSPen[[kk]]$pred.table))/sum(FSPen[[kk]]$pred.table))
     gamma <- c(gamma, Gamma.stat4(FSPen[[kk]]$pred.table))
}
gene.freq <- table(gene.freq)
gene.freq1 <- gene.freq[order(gene.freq, decreasing=T)]
gene.freq2 <- round(gene.freq1/48,digits=3)
library(hgu133a2.db)
keys = as.character(names(gene.freq2))
for (ii in 1:length(keys)){
     keys[ii] <- substr(keys[ii], 2, nchar(keys[ii]))
     }
match <- select(hgu133a2.db, keys=keys, cols="SYMBOL")
match <- match[-30,]
```

```
percent <- paste(round(gene.freq2*100, 3), "%", sep="")
match1 <- data.frame(match, percent)
```

I.5 Application to Glue Grant Burn Injury Study

```
### Data Preprocessing ###
# Data management for the Glue Grant Data #
data.clinical<-read.csv('TRDB_TIME_VARY_TRAUMA_RPT_20110919_111822.csv',header=T)
pat_id1<-as.factor(data.clinical[,1])
data.chip<-read.csv('TRDB_TRAUMA-PT_DEMO_MICRO_SVRTY_RPT_20110919_112945.csv',header=T)
pat_id2<-as.factor(data.chip[,1])
# This allows to match "patient_id/time" records in clinical data with microarray data
hours_data2<-data.chip[,"HOURS_SINCE_INJURY"]
data.chip[,31]<-as.integer(hours_data2/24)
colnames(data.chip)[31]<-"days_in_hospital"
data.clinical<-transform(data.clinical,pat_days=as.vector(paste(PATIENT_ID,
                      HOS_DAY_SNC_INJ, sep='/')))
data.chip<-transform(data.chip,pat_days=as.vector(paste(PATIENT_ID,
                      days_in_hospital, sep='/')))
new.data<-merge(data.clinical,data.chip, by.x='pat_days',by.y='pat_days',
                   all.x=FALSE,all.y=TRUE)
# Select columns of interest #
new.data1<-new.data[c('PATIENT_ID.x',"HOS_DAY_SNC_INJ", "HOURS_SINCE_INJURY",
                     "MARSHALL_NO_GCS",  "MARSHALL_MOF","MICROARRAY_ID","ARRAY_SAMPLE_TYPE",
                      "CHIP_NAME","ANALYSIS_TOOL", "NEURO_MOF", "RESP_MOF", "CARDIO_MOF",
                      "RENAL_MOF","HEP_MOF","HEM_MOF")]
table( new.data1[,'ARRAY_SAMPLE_TYPE'], new.data1[,'CHIP_NAME'])
#             HG-U133_Plus_2 hGlue_2_0_v1.1sq hGlue1_0.1sq
#  Buffy Coat            869                0            0
#  Monocyte                0              494          211
#  PMN                     0              501          218
#  PMN cDNA                0                2            0
#  T-cell                  0              492          216
data.chip1<-data.chip[data.chip[,"ARRAY_SAMPLE_TYPE"]=="Buffy Coat",]
data.clinical<-transform(data.clinical,pat_days=as.vector(paste(PATIENT_ID,
                       HOS_DAY_SNC_INJ, sep='/')))
data.chip1<-transform(data.chip1,pat_days=as.vector(paste(PATIENT_ID,
                     SAMPLE_STUDYSTART_DAYS, sep='/')))
new.data<-merge(data.clinical,data.chip1, by.x='pat_days',by.y='pat_days',
                   all.x=FALSE,all.y=TRUE)
attach(new.data)
new.data1<-new.data[order(PATIENT_ID.y,SAMPLE_STUDYSTART_DAYS),]
# extract columns of interest #
final.data<-new.data1[,c(  "PATIENT_ID.y","SAMPLE_STUDYSTART_DAYS","HOS_DAY_SNC_INJ",
                           "MARSHALL_NO_GCS", "MARSHALL_MOF", "MICROARRAY_ID", "NEURO_MOF",
 "RESP_MOF", "CARDIO_MOF","RENAL_MOF","HEP_MOF","HEM_MOF")]
meta.data <- final.data
meta.data <-read.csv ('final data.csv',row.name=1)
raw.data<-read.delim('dchip_tt_20111108_124331.txt',row.name=1) # 54675*1971 #
col.name<-colnames(raw.data)
exprs.index<-grep("expression",col.name)
exprs.data<-raw.data[,exprs.index] #54675*657#
call.index<-grep("call",col.name)
call.data<-raw.data[,call.index] #54675*657#
# To filter the All absent probes and create R object: Allabsent.RData#
# Allabsent <- rep(0, dim(call.data)[1])
# for (ii in 1: dim(call.data)[1]){
#     tmp <- ifelse(call.data[ii,]=="A",1,0)
#      Allabsent[ii] <- sum(tmp)
# }
load("Allabsent.RData")
temp <- data.frame(1:dim(call.data)[1], Allabsent)
remove.index <- temp[temp[,"Allabsent"]==dim(call.data)[2],1]
exprs.data <- exprs.data[-remove.index,] #48093*657 #
col.name<-colnames(exprs.data)
microarray.name<-vector(mode='numeric')
for ( ii in 1:length(col.name)){
  temp1<-strsplit(col.name[ii],"m")[[1]][2]
    temp2<-strsplit(temp1,"_")[[1]][1]
      microarray.name[ii]<-temp2
  }
microarray.name<-as.numeric(microarray.name)  # 657 records #
microarray.diff<-setdiff(meta.data[,6],microarray.name) # 212 records missing #
# Extract the missing information #
# Either the Marshall score missed or the number of days in hospital missed #
miss.pat<-matrix(nc=12,nr=length(microarray.diff))
for (ii in 1:length(microarray.diff)){
  miss.pat[ii,]<-as.matrix(meta.data[meta.data[,6]==microarray.diff[ii],])
  }
colnames(miss.pat)<-colnames(meta.data)
all.equal(miss.pat[,6], microarray.diff)
rownames(meta.data) <- 1:dim(meta.data)[1]
row.index<-vector(mode='numeric')
for ( ii in 1: length(microarray.diff)){
 row.index[ii]<-rownames(meta.data[meta.data[,"MICROARRAY_ID"]==microarray.diff[ii],])
}
row.index<-as.numeric(row.index)
meta.data<-meta.data[-row.index,]
all.equal(microarray.name, meta.data[,6])
Pheno.AFX<-new('AnnotatedDataFrame', data=exprs.data, varMetadata=meta.data)
# Merge raw.data and meta.data #
exprs.data <- t(exprs.data)
row.name <- rownames(exprs.data)
microarray.name<-vector(mode='numeric')
for ( ii in 1:length(row.name)){
  temp1<-strsplit(row.name[ii],"m")[[1]][2]
    temp2<-strsplit(temp1,"_")[[1]][1]
      microarray.name[ii]<-temp2
  }
microarray.name<-as.numeric(microarray.name)
rownames(exprs.data) <- microarray.name
exprs.data <- data.frame(microarray.name, exprs.data)
new.data <- merge(meta.data, exprs.data, by.x="MICROARRAY_ID", by.y="microarray.name")
attach(new.data) #657*48105#
new.data1 <- new.data[order(PATIENT_ID.y, SAMPLE_STUDYSTART_DAYS),]
new.data1 <- data.frame(new.data1[,1:12], log(new.data1[,13:dim(new.data1)[2]], base=2))
x <- new.data1[,13:dim(new.data1)[2]] #657*48093#
# Create modified ordinal outcome #
```

```
y <- new.data1[, "MARSHALL_MOF"]
ycat3 <- ycat4 <- NULL
for (ii in 1: length(y)){
     if ((y[ii]>=0) && (y[ii] <=4)) { ycat3[ii] <- 1
      } else if ((y[ii]>=5) && (y[ii] <=9)) { ycat3[ii] <- 2
       }  else if ((y[ii] >=10) && (y[ii] <=15)) {ycat3[ii] <- 3
     }
}
summary(as.ordered(ycat3))
# 1   2   3
# 287 298  72
for (ii in 1: length(y)){
     if (y[ii]==0) { ycat4[ii] <- 1
      } else if ((y[ii]>=1) && (y[ii] <=5)) { ycat4[ii] <- 2
       }  else if ((y[ii] >=6) && (y[ii] <=9)) {ycat4[ii] <- 3
      } else {ycat4[ii] <- 4}
}
summary(as.ordered(ycat4))
# 1   2   3   4
#172 178 235  72
y <- new.data1[,"RENAL_MOF"]
RENAL3 <- NULL
for (ii in 1:length(y)){
     if (y[ii]==0) { RENAL3[ii] <-0
     } else if (y[ii]==1) {RENAL3[ii] <- 1
      } else {RENAL3[ii] <-2}
}
summary(as.ordered(RENAL3))
#  0   1   2
# 236 395  26
y <- new.data1[,"NEURO_MOF"]
new.y <- vector(length=length(y), mode="numeric")
for (ii in 1:length(y)) {
     if((y[ii]==0) ||(y[ii]==1)) {new.y[ii] <- 1
     } else if ((y[ii]==2)||(y[ii]==3)) {new.y[ii] <- 2
       } else {new.y[ii] <- 3}
}
new.data1[, "NEURO_MOF"] <- new.y
new.data1 <- data.frame(ycat3, ycat4, RENAL3, new.data1)
write.csv(new.data1, "gluegrant.csv")
##############################################################################
### Model Fitting Procedure ###
source("FSPenMixed.R")]
data <- read.csv("gluegrant.csv")
### Marshall score assessed on renal system ###
# ordinal random intercept model #
output1 <-  FSPenMixed(response.name="RENAL3",
                       predictor.name="SAMPLE_STUDYSTART_DAYS",
                       id.name="PATIENT_ID",
                       time.name="SAMPLE_STUDYSTART_DAYS", data,
                       x=data.matrix(data[, 15:dim(data)[2]]),
                       y=data[,"RENAL3"],
                       model="Random Intercept", range=0, cut.prop=0.0015)
# ordinal random coefficient model #
```

```
output2 <-  FSPenMixed(response.name="RENAL3",
                       predictor.name="SAMPLE_STUDYSTART_DAYS",
                       id.name="PATIENT_ID",
                       time.name="SAMPLE_STUDYSTART_DAYS", data,
                       x=data.matrix(data[, 15:dim(data)[2]]),
                       y=data[,"RENAL3"],
                       model="Random Coefficient", range=0, cut.prop=0.0015)
### Marshall score assessed on central nervous system ###
# ordinal random intercept model #
 output1 <-  FSPenMixed(response.name="NEURO_MOF",
                       predictor.name="SAMPLE_STUDYSTART_DAYS",
                       id.name="PATIENT_ID",
                       time.name="SAMPLE_STUDYSTART_DAYS", data,
                       x=data.matrix(data[, 15:dim(data)[2]]),
                       y=data[,"NEURO_MOF"],
                       model="Random Intercept", range=0, cut.prop=0.0015)
# ordinal random coefficient model #
output2 <-  FSPenMixed(response.name="NEURO_MOF",
                       predictor.name="SAMPLE_STUDYSTART_DAYS",
                       id.name="PATIENT_ID",
                       time.name="SAMPLE_STUDYSTART_DAYS", data,
                       x=data.matrix(data[, 15:dim(data)[2]]),
                       y=data[,"NEURO_MOF"],
                       model="Random Coefficient", range=0, cut.prop=0.0015)
### aggregated Marshall score with three categories ###
# ordinal random intercept model #
output1 <-  FSPenMixed(response.name="ycat3",
                       predictor.name="SAMPLE_STUDYSTART_DAYS",
                       id.name="PATIENT_ID",
                       time.name="SAMPLE_STUDYSTART_DAYS", data,
                       x=data.matrix(data[, 15:dim(data)[2]]),
                       y=data[,"ycat3"],
                       model="Random Intercept", range=0, cut.prop=0.0015)
#ordinal random coefficient model #
output2 <-  FSPenMixed(response.name="ycat3",
                       predictor.name="SAMPLE_STUDYSTART_DAYS",
                       id.name="PATIENT_ID",
                       time.name="SAMPLE_STUDYSTART_DAYS", data,
                       x=data.matrix(data[, 15:dim(data)[2]]),
                       y=data[,"ycat3"],
                       model="Random Coefficient", range=0, cut.prop=0.0015)
# Affymatrix annotation match to Gene Symbol #
beta <- output1[[1]]$new.beta
name <- output1[[1]]$gene.name
tmp <- data.frame(name, beta, abs(beta))
tmp <- tmp[with(tmp, order(-tmp[,3])),]
# Match probes to Gene Symbol #
# source("http://bioconductor.org/biocLite.R")
# biocLite("hgu133a2.db")
library(hgu133a2.db)
keys = as.character(tmp[,"name"])
for (ii in 1:length(keys)){
     keys[ii] <- substr(keys[ii], 2, nchar(keys[ii]))
     }
match <- select(hgu133a2.db, keys=keys, cols=c("SYMBOL"))
tmp[,1] <- keys
tmp1 <-merge(match, tmp, by.x="PROBEID", by.y="name", all.x=T)
tmp1 <- tmp1[with(tmp1, order(-tmp1[,4])),]
tmp1 <- tmp1[,-4]
###############################################################################
###  Visualization using data with Marshall score assessed on the renal system ###
### Regularization Path Plot ###
source("Compile_step1.R")
source("Compile_step2.R")
data <- read.csv("gluegrant.csv")
x=data.matrix(data[, 15:dim(data)[2]])
y=data[, "RENAL3"]
y <-factor(y, levels=c(0, 1, 2), ordered=TRUE)
fit <- forward.stagewise.cum(x, y,  1e-4,  1e-4, 0.0015)
summary <- output.CumFS(fit, x, y)
beta <- fit$beta.list
step <- length(beta)
# full list of genes consecutively enters the active set #
add.gene <- list()
for (ii in 2:dim(summary)[1]){
    old.gene <- names(beta[[summary[ii-1,"step"]]])
    new.gene <- names(beta[[summary[ii,"step"]]])
    add.gene[[ii]] <- setdiff(new.gene, old.gene)
    add.gene[[1]] <- names(beta[[summary[1,"step"]]])
    }
step <- summary[60,"step"]
plot(0,0,type="n", xlim=c(0,step),
     ylim=c(min(beta[[step]]),max(beta[[step]])),
     xlab="Step", ylab="Penalized Estimate",cex.lab=1.3,
     main="GSE10006 Forward Stagewise Path", cex.main=1.3)
lines(x=c(0,step),y=c(0,0), type="l", col=1, lwd=2)
for (ii in 1:60){
gene <- add.gene[[ii]]
for (kk in ii:dim(summary)[1]) {
if (kk==1){
    lines(x=c(0,summary[kk,"step"]), y=c(0, beta[[summary[kk,"step"]]]),col=2,lwd=1.5)
} else if (kk==ii){
    lines(x=c(summary[kk-1,"step"],summary[kk,"step"]),
          y=c(0,beta[[summary[kk,"step"]]][names(beta[[summary[kk,"step"]]])==gene]),
          col=ii+1, lwd=1.5)
} else {
lines(x=c(summary[kk-1,"step"],summary[kk,"step"]),
      y=c(beta[[summary[kk-1,"step"]]][names(beta[[summary[kk-1,"step"]]])==gene],
          beta[[summary[kk,"step"]]][names(beta[[summary[kk,"step"]]])==gene]),
          col=ii+1,lwd=1.5)
      }
}
}
### Other plots ###
time <- as.factor(data[,"HOS_DAY_SNC_INJ.x"]) #the same as sample study day #
```

```
freq.time <- t(as.matrix(table(time)))
barplot1 <- barplot(freq.time, names.arg=as.numeric(colnames(freq.time)),ylim=c(0,170),
            main="Summary of Sample Study Days in Glue Grant Data", col=1,
            xlab="Days of hospitalization",
            ylab="Number of Subjects")
text(barplot1, freq.time, labels = freq.time, pos = 3)
time <- data[,"HOS_DAY_SNC_INJ.x"]
time1 <- vector(length=length(time), mode="character")
tmp1 <- data[data[,"HOS_DAY_SNC_INJ.x"]==0, 1:10]
tmp2 <- data[data[,"HOS_DAY_SNC_INJ.x"]==4, 1:10]
tmp3 <- data[data[,"HOS_DAY_SNC_INJ.x"]==7, 1:10]
tmp4 <- data[data[,"HOS_DAY_SNC_INJ.x"]==14, 1:10]
tmp <- rbind(tmp1, tmp2, tmp3, tmp4)
hist2 <- table(tmp[,"new.y"], tmp[,5])
colnames(hist2) <- c("Baseline","Day 4", "Day 7","Day 14")
pdf(file="/home/houj2/gluegrant/Stagewise/Gluegrant_renal_bar3.pdf")
barplot2 <- barplot(hist2, beside=T,
                    names.arg=levels(c("Baseline","Day 4", "Day 7","Day 14")),
                    col=terrain.colors(3,alpha=0.6), ylim=c(0,130),
                    legend.text=c("Normal", "Mild", "Moderate+"),
                    main="Distribution of Marshall Score on Renal by Time",
                    xlab="Days of hospitalization",
                    ylab="Number of Subjects")
text(barplot2, hist2, labels = hist2, pos = 3)
time <- data[," MARSHALL_MOF.x"]
freq.time <- t(as.matrix(table(time)))
barplot1 <- barplot(freq.time, names.arg=as.numeric(colnames(freq.time)),ylim=c(0,180),
            main="Distribution of Aggregated Marshall Score in Burn Injury Dataset",
            col=1, xlab="Marshall Score",
            ylab="Number of Observations")
text(barplot1, freq.time, labels = freq.time, pos = 3)
library(ggplot2)
colnames(data)[7] <- "TIME"
colnames(data)[4] <- "RENAL"
data[,"RENAL"] <- as.factor(data[,"RENAL"])
r <- qplot(TIME, X203932_at, data=data, colour=RENAL) +
xlab("Days of Hospitalization")+ylab("log2 Gene Expression")+opts(title = "HLA-DMB")
r+geom_smooth(size=2)
r <- qplot(TIME, X214909_s_at, data=data, colour=RENAL)+
xlab("Days of Hospitalization")+ylab("log2 Gene Expression")+opts(title="DDAH2")
r+geom_smooth(size=2)
r <- qplot(TIME, X209770_at, data=data, colour=RENAL)+
xlab("Days of Hospitalization")+ylab("log2 Gene Expression")+opts(title="BTN3A1")
r+geom_smooth(size=2)
r <- qplot(TIME, X214088_s_at, data=data, color=RENAL)+
xlab("Days of Hospitalization")+ylab("log2 Gene Expression")+opts(title="FUT3")
r+geom_smooth(size=2)
###############################################################################
###  Cross-validation using data with Marshall score assessed on the renal system ###
###  Each time, test data contains five subjects ###
load("FSPenMixedCV.R")
```

```
data <- read.csv("gluegrantl")
PenMixedCV <- FSPenMixedCV (response.name="RENAL3",
                            predictor.name="SAMPLE_STUDYSTART_DAYS",
                            id.name="PATIENT_ID",
                            time.name="SAMPLE_STUDYSTART_DAYS" ,
                            data, kfold=33,
                            x=data.matrix(data[, 14:dim(data)[2]]),
                            y=data[,"RENAL3"])
### Gathering all the cross validation results ###
# Define Goodman and Kruskal's gamma #
Gamma.stat <- function(tables) {
              c11 <- tables[1,1]*(sum(tables[2:3,2:3]))
              c12 <- tables[1,2]* sum(tables[2:3,3])
              c21 <- tables[2,1]* sum(tables[3,2:3])
              c22 <- tables[2,2]* tables[3,3]
              C <- c11 + c12 + c21 + c22
              d12 <- tables[1,2]*sum(tables[2:3,1])
              d13 <- tables[1,3]*sum(tables[2:3,1:2])
              d22 <- tables[2,2]*tables[3,1]
              d23 <- tables[2,3]*sum(tables[3,1:2])
              D <- d12 + d13 + d22 + d23
    gamma.stat <- (C-D)/(C+D)
    return(gamma.stat)
}
# 68 is the number of split, it varies according to kfold #
gene.freq <-  predaccuracy <- gamma <- NULL
for (kk in 1:68){
      gene.freq <- c(gene.freq, PenMixedCV[[kk]]$gene.name)
      predaccuracy <- c(predaccuracy,
      sum(diag(PenMixedCV[[kk]]$pred.table))/sum(PenMixedCV[[kk]]$pred.table))
      gamma <- c(gamma, Gamma.stat(PenMixedCV[[kk]]$pred.table))
      }
gene.freq <- table(gene.freq)
gene.freq1 <- gene.freq[order(gene.freq, decreasing=T)]
gene.freq2 <- round(gene.freq1/68, digits=3)
library(hgu133a2.db)
keys = as.character(names(gene.freq2))
for (ii in 1:length(keys)){
     keys[ii] <- substr(keys[ii], 2, nchar(keys[ii]))
     }
match <- select(hgu133a2.db, keys=keys, cols="SYMBOL")
```

I.6 High-dimensional Data Simulation

```
### Load functions and Define parameters ###
source("Compile_step1.R")
source("Compile_step2.R")
source("Compile_step3.R")
n <- 90  # number of samples #
p <- 1000 # number of features #
k <- 3       # number of categories #
sig.p <- 1  # number of important features #
n.simu <- 1 # number of simulation #
mean <- 0
mean2 <- 1.5
mean3 <- 3
sd <- 1
CVpredErr.AIC <- CVpredErr.BIC <- list()
CVgamma.AIC <- CVgamma.BIC <- list()
CV.nonzero.AIC <- CV.nonzero.BIC <- list()
# Define Goodman and Kruskal's gamma #
Gamma.stat <- function(tables) {
              c11 <- tables[1,1]*(sum(tables[2:3,2:3]))
              c12 <- tables[1,2]* sum(tables[2:3,3])
              c21 <- tables[2,1]* sum(tables[3,2:3])
              c22 <- tables[2,2]* tables[3,3]
              C <- c11 + c12 + c21 + c22
              d12 <- tables[1,2]*sum(tables[2:3,1])
              d13 <- tables[1,3]*sum(tables[2:3,1:2])
              d22 <- tables[2,2]*tables[3,1]
              d23 <- tables[2,3]*sum(tables[3,1:2])
              D <- d12 + d13 + d22 + d23
    gamma.stat <- (C-D)/(C+D)
    return(gamma.stat)
}
##############################################################################
### Construct Simulated Data ###
for (kk in 1:n.simu){
     x <- matrix(nc=p, nr=n)
     for (ii in 1:p) {
          x[,ii] <- rnorm(n, mean=mean, sd=sd)
     }
     sig.p.index <- sample(1:p, sig.p)
     nn <- n/k  #samples in each category#
     x.cat1 <- x.cat2 <- x.cat3 <- matrix(nr=n/k, nc=sig.p)
     mean2 <- mean2*sample(c(-1,1), size=sig.p, replace=TRUE)
     mean3 <- mean3*sample(c(-1,1), size=sig.p, replace=TRUE)
     for (ii in 1:length(sig.p.index)){
          x.cat1[,ii] <- rnorm(nn, mean=mean, sd=sd)
          x.cat2[,ii] <- rnorm(nn, mean=mean2[ii], sd=sd)
          x.cat3[,ii] <- rnorm(nn, mean=mean3[ii], sd=sd)
          }
     x.cat <- rbind(x.cat1, x.cat2, x.cat3)
     for (ii in 1:length(sig.p.index)){
          x[,sig.p.index[ii]] <- x.cat[,ii]
     }
y <- as.ordered(c(rep(1,nn), rep(2,nn), rep(3,nn)))
################################################################################
# N-fold Cross-Validation #
# Define Prediction Error and Gamma Statistics #
predErr.AIC <- predErr.BIC <- vector(length=n, mode="numeric")
gamma.AIC <- gamma.BIC <- vector(length=n, mode="numeric")
true.nonzero.AIC <-  true.nonzero.BIC <- vector(length=n, mode="numeric")
for (ii in 1:dim(x)[1]){
     x <- scale(x)
     train.sample <- x[-ii,]
     train.y <- y[-ii]
     eval.sample <- as.matrix(t(x[ii,]))
     eval.y <- y[ii]
     forwardstagewise <- forward.stagewise.cum(x=train.sample, y=train.y,
                                 epsilon=1e-4, tol=1e-4, 1)
     output <- output.CumFS(forwardstagewise, x=train.sample, y=train.y)
     step.index <- steps(output, 0, criteria="AIC")
     beta.list <- forwardstagewise$beta.list
     Num.nonzero <- unlist(forwardstagewise$num.nonzero)
     new.beta <- beta.gene <- beta.list[[step.index]]
     gene.name <- as.numeric(names(new.beta))
     fit <- logLL(train.sample, train.y, beta1=new.beta)
     new.alpha <- fit$par
     pred.y <- Predict(x, y, new.alpha, new.beta)
     pred.table <- table(pred.y, y)
     predErr.AIC[ii] <- 1 - sum(diag(pred.table))/sum(pred.table)
     gamma.AIC[ii] <- Gamma.stat(pred.table)
     true.nonzero.AIC[ii] <- length(intersect(sig.p.index, gene.name))
     }
     CVpredErr.AIC[[kk]] <- predErr.AIC
     CVgamma.AIC[[kk]] <- gamma.AIC
     CV.nonzero.AIC[[kk]] <- true.nonzero.AIC
     }
```

I.7 Longitudinal High-dimensional Data Simulation

```
### Load functions and Define parameters ###
source("Compile_step1.R")
source("Compile_step2.R")
source("Compile_step3.R")
################################################################################
# Create the data from longitudinal package #
library(longitudinal)
data(tcell)
# Rearrange data to sort by ID #
tcell <- NULL
for (ii in 1:10){
    tmp <- seq(ii, dim(tcell.10)[1], 10)
    temp <- tcell.10[tmp,]
    tcell <- rbind(tcell, temp)
    }
Time <- rep(c(0,2,4,6,8, 18, 24,32,48,72),10)
ID <- NULL
for (ii in 1:10){
     tmp.ID <- rep(ii,10)
     ID <- c(ID, tmp.ID)
     }
tcell <- data.frame(Time, ID, tcell)
################################################################################
CVpredErr.AIC <- CVpredErr.BIC <- list()
CVgamma.AIC <- CVgamma.BIC <- list()
CV.nonzero.AIC <- CV.nonzero.BIC <- list()
n.simu <- 5
for (kk in 1:n.simu){
     # Specify the beta and error to create ordinal response #
     beta.prior <- rep(0, 58)
     sig.p.index <- sample(1:58, 4)
     beta.prior[sig.p.index] <- sample(-5:5, 4)
     err <- rnorm(100, 0, 1)
     latent.y <- as.matrix(tcell[,3:60])%*%beta.prior  +  err
     cutoff <- quantile(latent.y, probs=seq(0,1,1/3))
     ordinal.y <- rep(0, 100)
     for (ii in 1:100){
         if(latent.y[ii] <= cutoff[2]) {ordinal.y[ii] <-1
         } else if ((cutoff[2] < latent.y[ii]) &&(latent.y[ii]<=cutoff[3])){ordinal.y[ii] <-2
         } else {ordinal.y[ii] <- 3}
     }
     ordinal.y <- as.ordered(ordinal.y)
################################################################################
# Cross-Validation #
# Define Prediction Error and Gamma Statistics #
predErr.AIC <- predErr.BIC <- vector(length=dim(tcell)[1], mode="numeric")
gamma.AIC <- gamma.BIC <- vector(length=dim(tcell)[1], mode="numeric")
true.nonzero.AIC <-  true.nonzero.BIC <- vector(length=dim(tcell)[1], mode="numeric")
for (ii in 1:dim(tcell)[1]){
     tcell <- as.matrix(data.frame(tcell[,1:2],scale(tcell[,3:60])))
     train.sample <- tcell[-ii,3:60]
     train.y <- ordinal.y[-ii]
     eval.sample <- as.matrix(t(tcell[ii, 3:60]))
     eval.y <- ordinal.y[ii]
 forwardstagewise <- forward.stagewise.cum(x=train.sample, y=train.y,
                                  epsilon=1e-4, tol=1e-4, 1)
     output <- output.CumFS(forwardstagewise, x=train.sample, y=train.y)
     step.index <- steps(output, 0, criteria="AIC")
     beta.list <- forwardstagewise$beta.list
     Num.nonzero <- unlist(forwardstagewise$num.nonzero)
     new.beta <- beta.gene <- beta.list[[step.index]]
     gene.name <- as.numeric(names(new.beta))
     fit <- logLL(train.sample, train.y, beta1=new.beta)
     new.alpha <- fit$par
     pred.y <- Predict(x, y, new.alpha, new.beta)
     pred.table <- table(pred.y, y)
     predErr.AIC[ii] <- 1 - sum(diag(pred.table))/sum(pred.table)
     gamma.AIC[ii] <- Gamma.stat(pred.table)
     true.nonzero.AIC[ii] <- length(intersect(sig.p.index, gene.name))
     }
     CVpredErr.AIC[[kk]] <- predErr.AIC
     CVgamma.AIC[[kk]] <- gamma.AIC
     CV.nonzero.AIC[[kk]] <- true.nonzero.AIC
     }
```

VITA

Jiayi Hou was born April 12, 1986 in Chengdu, Sichuan Province, People's Republic of China. She received her Bachelor's of Science in Theoretical Mathematics from Sichuan Unversity in May of 2008. In August of 2008, she moved to United States to pursue her Ph.D in Biostatistics at Virginia Commonwealth University in Richmond, VA.