5-2016

Inference for a zero-inflated Conway-Maxwell-Poisson regression for clustered count data.

Hyoyoung Choo-Wosoba
University of Louisville

INFERENCE FOR A ZERO-INFLATED CONWAY-MAXWELL-POISSON REGRESSION FOR CLUSTERED COUNT DATA

By

Hyoyoung Choo-Wosoba
B.A., KonKuk University, 2006
M.S., University of Nebraska-Lincoln, 2009

A Dissertation
Submitted to the Faculty of the
School of Public Health and Information Science
in Partial Fulfillment of the Requirements
for the Degree of

Doctor of Philosophy
in
Biostatistics: Decision Science

Department of Bioinformatics and Biostatistics
University of Louisville
Louisville, Kentucky

May 2016

INFERENCE FOR A ZERO-INFLATED CONWAY-MAXWELL-POISSON REGRESSION FOR CLUSTERED COUNT DATA

By

Hyoyoung Choo-Wosoba
B.A., KonKuk University, 2006
M.S., University of Nebraska-Lincoln, 2009

A Dissertation Approved on

April 14, 2016

by the following Dissertation Committee:

_____
Somnath Datta, Ph.D., Dissertation Director

_____
Susmita Datta, Ph.D.

_____
Maiying Kong, Ph.D.

_____
Jeremy Gaskins, Ph.D.

_____
Ryan Gill, Ph.D.

DEDICATION

This dissertation is dedicated to my parents

Mr. JaeSung Choo

and

Mrs. HyunMi Ma

who have given me invaluable educational opportunities

and to my husband

Mr. Adam Wosoba

who has supported me in achieving my academic goal.

ACKNOWLEDGMENTS

I would like to thank my academic advisor, Dr. Somnath Datta, for his sincere guidance and advice. I would also like to thank all the other committee members, Dr. Susmita Datta, Dr. Maiying Kong, Dr. Jemery Gaskins, and Dr. Ryan Gill, for their assistance and thoughtful comments regarding my research. I specially want to give my thanks to Dr. Gaskins for advising my last project (Chapter 4). I would also like to express my thanks to my husband, Adam, for his understanding and patience during my academic years for attaining a Ph.D.

ABSTRACT

INFERENCE FOR A ZERO-INFLATED CONWAY-MAXWELL-POISSON REGRESSION FOR CLUSTERED COUNT DATA

Hyoyoung Choo-Wosoba

April 14, 2016

This dissertation is directed toward developing a statistical methodology with applications of the Conway-Maxwell-Poisson (CMP) distribution (Conway, R. W., and Maxwell, W. L., 1962) to count data. The count data for this dissertation exhibit three different characteristics: clustering, zero inflation, and dispersion. Clustering suggests that observations within clusters are correlated, and the zero inflation phenomenon occurs when the data exhibit excessive zero counts. Dispersion implies that the mean is greater/smaller than the variance unlike a Poisson distribution.

The dissertation starts with an introduction of inference for a zero-inflated clustered count data in the first chapter. Then, it presents novel methodologies through three different statistical approaches (Chapters 2-4). A marginal regression approach as the second chapter which begins with a description of a zero-inflated CMP model and subsequently develops proper statistical methodologies for estimating marginal regression parameters. Furthermore, various types of simulations are conducted to investigate whether the marginal regression approach leads to the proper statistical inference. This chapter also provides an application to a dental dataset, which is clustered, zero inflated, and dispersed. Chapter 3 develops a mixed effects model including a cluster-specific random effect term. This chapter also addresses numerical challenges of a mixed effects model approach through extensive simulations. For the application of the zero-inflated mixed effects model, next generation sequencing (NGS) data from a maize hybrids experiement is analyzed.

While Chapter 3 applies a mixed effects model using the frequentist approach, Chapter 4 develops a Bayesian method to analyze such data under a mixed effects model sturucture. In that chapter, a hurdle model is applied to cope with a zero inflation phenomenon, rather than a zero-inflated model used in both Chapters 2 and 3. Furthermore, Chapter 4 provides the application to the same dental dataset used in Chapter 2. The application section introduces a new factor into a hurdle mixed effects model, which incorporates both fixed effects term and random effects term. Chapter 5 describes the future plan as the concluding chapter.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

This dissertation consists of three interconnected research projects (Chapters 2-4). The first project (Chapter 2) deals with marginal zero-inflated CMP regression models for clustered count data that has excessive zero values. The aim of the second project (Chapter 3) is the similar to that of the first project but the analysis is based on a joint model involving cluster specific random effects with zero-inflated CMP conditionals. The last project (Chapter 4) is to apply a Bayesian approach method with a hurdle CMP mixed effects model.

1.1 A marginal model approach to zero-inflated clustered count data

Some count datasets have more zero values than expected from a certain common count distribution such as Poisson or negative binomial. This phenomenon, called zero-inflation, takes place in diverse fields such as engineering, dentistry, health surveys, transport, genomics and so on. To this end, zero-inflated versions of these distributions and related inferential procedures have been derived (Böhning, 1998; McLachlan, 1997; Yau,Wang and Lee, 2003) .

A Poisson distribution is well known for modeling count data. It is a relatively simple distribution that belongs to an exponential family which makes it convenient for analysis within the generalized linear models (GLM) framework. However, a Poisson distribution may not be the best choice in certain cases when the data is under- or over- dispersed, which is violation of the property that the variance and the mean are equal. Negative binomial is a popular choice to model such data. However, while a negative binomial distribution fits reasonably well for overdispersion, that is not the case for underdispersion. The Conway-Maxwell Poisson (CMP) distribution introduced by Conway and Maxwell (1962) is a great tool to

overcome this difficulty, since it can model a wide range of dispersion. In addition, it belongs to an exponential family as well.

Often in practice not all the data values are independent. Instead they arise as independent groups called clusters. An illustrative dataset is provided in Project 1 and 2, where observations on teeth belonging to the same individual form a cluster. They are expected to exhibit some form of statistical dependence due to shared environmental factors. Currently, SAS version 13.1 has a procedure, PROC COUNTREG which allows us to perform a regression analysis based on the zero-inflated CMP distribution and the COMPoissonReg package in R performs a CMPoisson regression analysis based on a GLM framework (Sellers and Shmueli, 2010). However, all these procedures are only applicable to independent data. Thus, the motivation of the first project is to seek a proper statistical method for a count dataset that is clustered and fitted into a zero-inflated marginal CMP model. In this project, we illustrate two statistical methodologies: MES(Modified Expectation-Solution) algorithm and MPL(Maximum Pseudo Likelihood) method. The MES algorithm, one of our proposed methods is a modified version of an ES algorithm which is introduced by Rosen, Jiang and Tanner, 2000. The MES algorithm shows how to account for the clustered count data with excessive zeros based on a CMP regression model. The MPL method gives estimators under the inference where clustered data is considered as being independent but account for the clustering feature through a clustered-adjusted variance estimation. Hence, our methods not only consider the clustering features but also deal with a wide range of dispersion. Furthermore, we assess our methods has properties of asymptotic theory through various simulations and provide an application to a dental data with a proper interpretation.

1.2 A joint modeling approach to zero-inflated clustered count data

Introduction of the random effects into a regression model has been a useful statistical tool for the analysis of longitudinal/clustered data. Considering not only fixed effects but also random effects allows us to adjust for subject-level (within-subject) randomness. Typically, a joint model via a mixed effects model has more accuracy than a marginal model as long as a random effect structure with random factors is correctly

specified. Furthermore, a joint model is able to produce a full likelihood function so that we can make efficient inference with standard large sample statistical theory and methods.

Recently there have been a number of attempts to apply the joint modeling framework into clustered/correlated count data with excessive zeros (Fulton, Liu, Haynie and Albert, 2015; Hall, 2000; Hasan, Sneddon and Ma, 2009, ets). Furthermore, a zero-inflated mixed effect modeling regression analysis is able to handle a limited range of dispersion of count data (Yau, Wang and Lee, 2003; Rodrigues-Motta, Gianola and Heringstad, 2010; ets). We can easily perform a regression analysis on certain types of distributions and dispersion through statistical software programs such as the "GLIMMIX" procedure in SAS 9.2 version and the "glmmADBM" package in R. However, these articles and programs have a limited number of count distributions (Poisson and negative binomial) and can only manage overdispersed data. In this regard, our model, zero-inflated CMP enables us to cope with various types of dispersion for any count data because of the versatility of a CMP distribution mentioned in Section 1.1.

In this work, we adapt the Gaussian-Hermite (G-H) quadrature method to calculate an approximation of the likelihood function from a zero-inflated CMP mixed effects model since there is no closed explicit form of the true likelihood function. As a by product of our approach, we are able to construct a statistical test for zero inflation in the data. We also carry out a numerical power analysis for testing a covariate effect as well as that of a zero-inflation test. Further, we apply our methodology to a maize hybrids experiment dataset to illustrate our methodology for clustered zero-inflated count data with two types of dispersions (over and under).

1.3 A Bayesian approach to zero-inflated clustered count data

A Bayesian method may incorporate a mixed effects model without requiring the quadrature method or other approximations to calculate the likelihood function. While the frequestist-based estimates are obtained by maximizing the log-likelihood function, estimates in a Bayesian approach are generated from the posterior distributions by using Markov chain Monte Carlo (MCMC) sampling methods. In this project, a hurdle model framework is applied to account for zero inflation and clustering.

Barriga and Louzada (2014) performed a Bayesian approach to analyze a zero-inflated dispersed data based on a CMP distribution. Our work differs from theirs as they only consider independent data, not dependent/clustered data. Our model considers clustering by adding a random effects term. Moreover, we use a hurdle model rather than a zero-inflated model which was used for their paper. The hurdle model gives an easier interpretation in terms of the way to divide zero inflated data into two different parts. The hurdle model considers the data with zero and non-zero parts, and analyzes each part with the corresponding distribution. However, in the zero-inflated model, data is explained by a mixture of two different distributions so that zero counts can be explained either from a degenerate distribution at zero or a count distribution. This statistical methodology is partially motivated to analyze data from the Iowa Fluoride Study (IFS) on nine-year old children which is used for an application in Chapter 2.

CHAPTER 2

MARGINAL REGRESSION MODELS FOR CLUSTERED COUNT DATA BASED ON ZERO-INFLATED CONWAY-MAXWELL-POISSON DISTRIBUTION WITH APPLICATIONS

2.1 Methods and Materials

We begin with the probability mass function (pmf) of a CMP distribution,

$$p(y) = \frac{\lambda^y}{(y!)^v Z(\lambda, v)} \ , \ y = 0, 1, 2, \ldots , \tag{1}$$

where $Z(\lambda, v) = \sum_{s=0}^{\infty} \frac{\lambda^s}{(s!)^v}$. Here $\lambda > 0$ is a shape parameter and $v \geq 0$ is a dispersion parameter. If v is 1, a CMP distribution is the exactly same as a Poisson distribution which means there is no additional dispersion. It turns out that $v > 1$ represents underdispersion and $v < 1$ represents overdispersion. Note that this distribution belongs to an exponential family since $p(y) = exp\{ylog(\lambda) - vlog(y!)\}Z^{-1}(\lambda, v)$. The limiting cases of a CMP distribution also include a Bernoulli distribution ($v = \infty$), or a geometric distribution ($v = 0$ and $\lambda < 1$). Thus, a CMP distribution has great flexibility to include various types of count distributions. Another important feature of the CMP distribution is about the expectation function of Y. In general, the means behave independently from the dispersion parameters; in other words, the dispersion parameters do not affect the means. However, Equation (2) shows that the mean of Y in the CMP distribution* is not only a function of the shape parameter, λ, but also of the dispersion parameter, v.

$$EY = \sum_{s=0}^{\infty} \frac{s\lambda^s}{(s!)^v} / Z(\lambda, v). \tag{2}$$

---

*A penultimate version of this chapter appeared in Choo-Wosoba, Levy, and Datta, 2016.

A zero-inflated model consists of two components: the zero-degenerated distribution δ_0, and a particular count distribution W. The zero-degenerated part controls excessive zeros in the form of a binary distribution and a count distribution. The CMP distribution, in this case, controls counts including the expected number of zeros. Thus, the zero-inflated CMP (or ZICMP, hereafter) model has a probability mass function described by

$$P(Y=y)=\begin{cases} p+\frac{(1-p)}{Z(\lambda,v)}, & \text{if } y=0, \\ (1-p)\frac{\lambda_{ij}^{y_{ij}}}{(y_{ij}!)^v Z(\lambda_{ij},v)}, & \text{if } y\geq 1, \end{cases} \tag{3}$$

where $p \in [0,1]$ is a parameter of the distribution representing the mixing proportion of the degenerate at zero part.

We assume that data are clustered into N clusters. The size of the i^{th} cluster is denoted by n_i, $1 \leq i \leq N$. Furthermore, let Y_{ij} indicate the j^{th} observation in the i^{th} cluster, $1 \leq j \leq n_i$.

The expectation-maximization (EM) algorithm is widely used for estimating parameters in a zero-inflated model. However, the EM algorithm, by itself, is not a valid tool for clustered data.

In this chapter, two different methods are proposed to explain the mechanisms of marginal framework, accounting for not only zero-inflation but also dependency. One is MES (Modified Expectation-Solution) algorithm and the other one is MPL (Maximum Pseudo Likelihood) method.

2.1.1 MES algorithm based on a modified Newton-Raphson method

The MES algorithm is motivated from the ES (Expectation-Solution) algorithm (Hall and Zhang, 2004; Rosen, Jiang and Tanner, 2000) when the data are clustered. The ES algorithm combines elements of both GEE (Liang and Zeger, 1986) and the EM algorithms, so that one can account for dependency (clustering) in the data. However, the ES algorithm as prescribed by Rosen, Jiang and Tanner (2000) has a major limitation in that it is only applicable to an exponential dispersion family which has a form of $f(y_{ij}; \theta_{ij}, \phi) = h(y_{ij}, \phi) exp\{\frac{(\theta_{ij} y_{ij} - k(\theta_{ij}))w_{ij}}{\phi}\}$, where θ_{ij} is the canonical parameter, w_{ij} is a constant, and ϕ is a dispersion parameter. Unfortunately, the CMP distribution does not belong to the exponential dispersion

family. Since the Z function can not be factored into a function of $\lambda_{ij}\,(=\log(\theta_{ij}))$ and a function of v, it cannot be re-expressed in the exponential dispersion family form. As a consequence of this, the expectation of Y is not only related to λ but also to v making a regression formulation complicated.

Therefore, we propose the following modification of the standard ES algorithm to deal with the CMP family; we call it the MES algorithm. Note that given a specified value of v, the CMP distribution indexed by λ belongs to an exponential dispersion family with $h=(y!)^{-\nu}$, $k=\log Z(\lambda,v)$, $w=1$, and $\phi=1$. So, instead of using the ES algorithm for estimating all parameters, we applied the ES algorithm for only regression coefficients other than v. For estimating v, a log-likelihood function is applied instead of GEE. The parameters of interest based on this algorithm consist of $\theta=\{\beta,\gamma,v,\rho,\delta\}$: a dispersion parameter, v, both β and γ as coefficients of the count and zero-inflation parts from the GLM framework of $log(\lambda(\beta))=X_\beta\beta$ and $logit(p(\gamma))=X_\gamma\gamma$ and correlation coefficients, ρ and δ from correlation matrices corresponding to the count and zero-inflation parts in GEE formulation. X_β and X_γ are covariates in the CMP distribution and zero-degenerated distribution, respectively. The covariates are determined depending on researchers' interests.

An MES algorithm starts with the complete log-pseudo-likelihood of zero-inflated CMP (ZICMP) model given by

$$\ell^c(\beta,\gamma,v;y_{ij},\,u_{ij})=\sum_{i=1}^{N}\sum_{j=1}^{n_i}u_{ij}logp(\gamma_{ij})+\sum_{i=1}^{N}\sum_{j=1}^{n_i}(1-u_{ij})log(1-p(\gamma_{ij}))+$$

$$\sum_{i=1}^{N}\sum_{j=1}^{n_i}(1-u_{ij})(y_{ij}log\lambda_{ij}(\beta)-vlog(y_{ij}!)-logZ(\lambda_{ij}(\beta),v)),\qquad(4)$$

where u_{ij} are latent (i.e., unobserved) binary indicators of the degenerate at zero part. (We call it a pseudo-likelihood because it is a product over likelihoods of individual terms as if they were independent.) Subsequently, it alternates between two main steps: the expectation (E) step and the solution (S) step. The E-step is to calculate the expectation of the expressions in each side of equation (4) by replacing u_{ij} with

$E(u_{ij})$ leading to

$$Q = E(\ell^c(\beta, \gamma, v;\, \boldsymbol{y},\, \boldsymbol{u})) = \sum_{i=1}^{N}\sum_{j=1}^{n_i} \ell^c(\beta, \gamma, v; y_{ij},\, E(u_{ij}))\,,$$

where

$$E(u_{ij}) = P(u_{ij} = 1|\, y_{ij} = 0, \beta, \gamma, v)$$

$$= \frac{p_{ij}}{p_{ij} + (1 - p_{ij})/Z(\lambda_{ij}(\beta), v)}. \tag{5}$$

Let u_{ij}^h denote this value at the h^{th} iteration. In the solution step, given $E(\boldsymbol{u})\,(=\boldsymbol{u}^h\,)$, estimates of β, γ and v are obtained by solving there own linearized estimating equations leading to the following updating schemes:

$$\gamma^{h+1} = \gamma^h + \kappa\left[\sum_{i=1}^{N} \frac{\partial \boldsymbol{p}_i^T}{\partial \gamma}\{V_{u_i}\}^{-1}\frac{\partial \boldsymbol{p}_i}{\partial \gamma} + \Psi_{1i}(\gamma)\Psi_{1i}(\gamma)^T\right]^{-1}\Psi_1(\gamma),$$

$$\beta^{h+1} = \beta^h + \kappa\left[\sum_{i=1}^{N} \frac{\partial E(\boldsymbol{y}_i)^T}{\partial \beta}\{V_{y_i}\}^{-1} Diag(1 - \boldsymbol{u}_i^h)\frac{\partial E(\boldsymbol{y}_i)}{\partial \beta^T} + \Psi_{2i}(\beta)\Psi_{2i}(\beta)^T\right]^{-1}\Psi_2(\beta),$$

$$v^{h+1} = v^h$$

$$- \kappa\left[\sum_{i=1}^{N}\sum_{j=1}^{n_i}\frac{\partial \ell_{ij}^c(v^h|\beta^h, \gamma^h)}{\partial v}\right] \Big/ \left[\sum_{i=1}^{N}\sum_{j=1}^{n_i}\frac{\partial^2 \ell_{ij}^c(v^h|\beta^h, \gamma^h)}{\partial v^2} + \left(\sum_{i=1}^{N}\sum_{j=1}^{n_i}\frac{\partial \ell_{ij}^c(v^h|\beta^h, \gamma^h)}{\partial v}\right)^2\right], \tag{6}$$

where

$$\Psi_1(\gamma) = \sum_i \Psi_{1i}(\gamma) \;= \sum_{i=1}^{N}\frac{\partial \boldsymbol{p}_i(\gamma)^T}{\partial \gamma} V_{u_i}^{-1}(\boldsymbol{u}_i^h - \boldsymbol{p}_i(\gamma)), \tag{7}$$

and

$$\Psi_2(\beta) = \sum_i \Psi_{2i}(\beta) \;= \sum_{i=1}^{N}\frac{\partial E(\boldsymbol{y}_i)^T}{\partial \beta} V_{y_i}^{-1} Diag(1 - \boldsymbol{u}_i^h)(\boldsymbol{y}_i - E(\boldsymbol{y}_i(\beta))). \tag{8}$$

See Appendix 2.1 and 2.2 for the details of the estimating functions Ψ_1 and Ψ_2. Note that the estimating functions for γ and β are of the GEE form (as in a standard ES algorithm) whereas that for the v is from a complete data pseudo-likelihood. Further, the CMP distribution does not fall under an exponential dispersion family for changing v, and consequently we cannot apply the GEE methodology to estimate v.

Also note that a step-size parameter κ is introduced in the updating scheme as compared with the classical Newton-Raphson method so that the algorithm converges slowly and steadily. The iterative algorithm stops when the maximum componentwise difference of the estimates between two successive iterations falls below a threshold ϵ.

The working variance-covariance matrices for the zero-inflation and the count parts are specified as $V_{u_i} = A_i^{1/2} R(\delta) A_i^{1/2}$ and $V_{y_i} = D_i^{1/2} R(\rho) D_i^{1/2}$, respectively. Here $A_i = A_i(\boldsymbol{p}_i(\gamma)) = Var(\boldsymbol{u}_i) = Diag(\boldsymbol{p}_i(1-\boldsymbol{p}_i))$, $D_i = D_i((E(\boldsymbol{y}_i|\beta, v)) = Diag(Var(\boldsymbol{y}_i))$, and $R(\delta)$ and $R(\rho)$ are working correlation matrices.

For estimating the correlation coefficients, δ and ρ, the GEE formulations can be used; the corresponding estimating equations are given by (see Appendice 2.3 and 2.4 for details)

$$\Psi_3(\delta) = \sum_{i=1}^{N} \frac{\partial \rho_{\gamma i}(\delta)}{\partial \delta^T} W_{\gamma i}^{-1} (U_i^{\gamma} - \rho_{\gamma i}(\delta)) = 0\,, \tag{9}$$

$$\Psi_4(\rho) = \sum_{i=1}^{N} \frac{\partial \rho_{\beta i}(\rho)}{\partial \rho^T} W_{\beta i}^{-1} H_{\beta i} (U_i^{\beta} - \rho_{\beta i}(\rho)) = 0, \tag{10}$$

where $W_{\gamma i}$ and $W_{\beta i}$ are working variance covariance matrices for the zero-degenerating and the count (i.e., CMP) distributions, respectively. Note that all the estimated parameters are updated iteratively as explained before. We are sometimes suppressing the index h for notational simplicity.

2.1.2 The Maximum Pseudo-Likelihood (MPL)

Estimators from the MPL method are obtained by maximizing the observed log-pseudo-likelihood function,

$$\ell(\beta, \gamma, v; y_{ij}) = \sum_{i=1}^{N} \sum_{j=1}^{n_i} I(y_{ij} = 0) \log[p(\gamma_{ij}) + \{1 - p(\gamma_{ij})\} / Z(\lambda_{ij}(\beta), v)]$$

$$+ \sum_{i=1}^{N} \sum_{j=1}^{n_i} I(y_{ij} \geq 1) [\log\{1 - p(\gamma_{ij})\} + (y_{ij} \log\{\lambda_{ij}(\beta)\} - v \log(y_{ij}!)$$

$$- \log\{Z(\lambda_{ij}(\beta), v))\}], \tag{11}$$

with respect to β, γ associated with covariates, X_β and X_γ, and the dispersion parameter v. The above log-pseudo-likelihood is constructed under the independence assumption; so, an adjusted variance method is used to account for the dependency within clusters. The adjusted variance proposed in this project is based on the log-pseudo-likelihood-based sandwich variance.

2.1.3 Variance Estimations

We investigate two different variance estimation methods: a sandwich-variance based on the large sample approximation and one using a nonparametric bootstrap at the cluster level. Large sample sandwich variances are calculated both for the MPL estimators and the estimators obtained from the MES algorithm. The typical sandwich covariance matrix is of the form $B^{-1}MB^{T^{-1}}$. The matrices B and M for the MPL method for independent data are given by

$$B_{\text{MPL}} = E\widehat{B}_{\text{MPL}}, \text{ with } \widehat{B}_{\text{MPL}} = -\begin{pmatrix} \frac{\partial^2 \ell}{\partial\beta\partial\beta^T} & 0 & \frac{\partial^2 \ell}{\partial\beta\partial v} \\ 0 & \frac{\partial^2 \ell^c}{\partial\gamma\partial\gamma^T} & 0 \\ \frac{\partial^2 \ell}{\partial v\partial\beta^T} & 0 & \frac{\partial^2 \ell}{\partial v^2} \end{pmatrix}_{p_\beta * p_\gamma} \text{ and} \tag{12}$$

$$M_{\text{MPL}} = E\begin{pmatrix} \frac{\partial \ell}{\partial\beta} \\ \frac{\partial \ell}{\partial\gamma} \\ \frac{\partial \ell}{\partial v} \end{pmatrix}\begin{pmatrix} \frac{\partial \ell}{\partial\beta} \\ \frac{\partial \ell}{\partial\gamma} \\ \frac{\partial \ell}{\partial v} \end{pmatrix}^T = E\sum_{i=1}^{N}\sum_{j=1}^{n_i}\begin{pmatrix} \frac{\partial \ell_{ij}}{\partial\beta} \\ \frac{\partial \ell_{ij}}{\partial\gamma} \\ \frac{\partial \ell_{ij}}{\partial v} \end{pmatrix}\begin{pmatrix} \frac{\partial \ell_{ij}}{\partial\beta} \\ \frac{\partial \ell_{ij}}{\partial\gamma} \\ \frac{\partial \ell_{ij}}{\partial v} \end{pmatrix}^T, \tag{13}$$

where ℓ is the observed log-pseudo-likelihood function in (11), p_β and p_γ are the numbers of covariates used for the count part and the zero part, respectively (see Appendix 2.5 for details). However, the equation (13) does not account for dependence within a cluster; so an adjusted sandwich covariance matrix is applied in the complete log-pseudo-likelihood function. The adjusted sandwich covariance matrix for the MPL estimators is obtained in the form of $\widehat{B}^{-1}_{\text{MPL}}\widehat{M}^{*}_{\text{MPL}}\widehat{B}^{T^{-1}}_{\text{MPL}}$ using the independence of the clusters where

$$\widehat{M}^{*}_{\text{MPL}} = \sum_{i=1}^{N}\begin{pmatrix} \frac{\partial \ell_i}{\partial\beta} \\ \frac{\partial \ell_i}{\partial\gamma} \\ \frac{\partial \ell_i}{\partial v} \end{pmatrix}\begin{pmatrix} \frac{\partial \ell_i}{\partial\beta} \\ \frac{\partial \ell_i}{\partial\gamma} \\ \frac{\partial \ell_i}{\partial v} \end{pmatrix}^T.$$

The sandwich covariance matrix for $\widehat{\theta}_{\text{MES}} = (\widehat{\beta}^T, \widehat{\gamma}^T, \widehat{v})^T$ obtained from the MES algorithm is in the form of $Var\left(\widehat{\theta}_{\text{MES}}\right) = B^{-1}_{\text{MES}}M_{\text{MES}}B^{T^{-1}}_{\text{MES}}$, where the matrices B_{MES} and M_{MES} are defined in Appendix

1.6. We note that it is not possible to estimate B_{MES} based on the model assumptions since we do not specify the joint likelihood of the clustered observations. Therefore, using a bootstrap based covariance matrix is a natural option in this case. Of course, bootstrap could be used for obtaining the standard errors for the MPL estimators as well.

We employ a cluster bootstrap technique (Field and Welsh, 2007) to perform the resampling since the clusters are independent and the primary sampling units. This way, the intra-cluster correlation will be preserved for the resampled data. Thus, each bootstrap sample is generated by resampling at the cluster level with replacement. Mathematically, let $i^*_{1b}, \cdots, i^*_{Nb}$ be a random sample of indices drawn with replacement from $\{1, \cdots, N\}$, for $1 \leq b \leq B$. Then the b^{th} bootstrap dataset is given by $(y^*_{1b}, X^*_{\beta,1b}, X^*_{\gamma,1b}), \cdots, (y^*_{Nb}, X^*_{\beta,Nb}, X^*_{\gamma,Nb})$, where $y^*_{jb} = y_{i^*_{jb}}, X^*_{\beta,jb} = X_{\beta,i^*_{jb}}, X^*_{\gamma,jb} = X_{\gamma,i^*_{jb}}$. The bootstrap standard errors based on B bootstrap resamples are calculated as

$$SE_{\text{BS}}(\widehat{\theta}) = \sqrt{\frac{1}{B}\sum_{b=1}^{B} diag\Big\{(\widehat{\theta}^*_b - \widehat{\overline{\theta}}^*)(\widehat{\theta}^*_b - \widehat{\overline{\theta}}^*)^T\Big\}},$$

where $\widehat{\theta}^*_b$ is the vector of estimates obtained by either the MPL method or the MES algorithm from the b^{th} bootstrap sample and $\widehat{\overline{\theta}}^*$ is the mean of the B bootstrap estimates.

2.2 Applications

This section introduces two different count datasets that include both zero inflation and clustering characteristics. The first dataset is obtained from the Iowa Fluoride Study (Levy et al., 2003) that serves as an example of the overdispersion phenomenon; the second illustrative dataset is taken from an NGS assay on maze hybrids and provides an example of underdispersion in count data.

2.2.1 An application for the Iowa Fluoride Study (IFS)

We apply our marginal regression model to analyze a dataset on dental caries from the Iowa Fluoride Study (Levy et al., 2003). As mentioned before, this dataset possesses the characteristics of zero-inflation, overdispersion and clustered counts. IFS was a longitudinal study of Iowa children who were

recruited at age 5 (http://www.dentistry.uiowa.edu/preventive-fluoride-study). For this illustration, we looked at the data at the first follow-up when they were about nine years of age.

The response is the caries experience score (CES) that is obtained by summing the scores of individual dental surface scores for each tooth (scored 0, 1 or 2 depending on the caries severity). Eight potential risk/protective factors (covariates) are available in Table 2.1.

Altogether, 464 children are included in our analysis.

We treat the outcomes (i.e., CES) on teeth belonging to the same child to be clustered. It is likely that they will be correlated due to shared genetic and environmental factors. The cluster size varies between 16 and 24. Overall, there are 10,838 observations on the CES. A preliminary inspection of the CES values reveals that zero-inflation is a concern for this dataset (Figure 2.1).

We fit a clustered ZICMP model to these data where the parameters are estimated using both the MPL and MES methods. The ZIP estimates obtained from the R package '*pscl*' are used as the starting values in the MES algorithm. The standard errors of the MPL estimators are calculated by using the adjusted sandwich variance method mentioned before. For the standard errors of MES estimators, the bootstrap scheme (outlined in Section 2.1) is used with bootstrap size $B = 500$. Finally, p-values for each of the potential risk/protective factors are calculated using a large sample Wald test.

Before we describe the significance of the risk/protective factors, we want to note that $\widehat{v}$ turned out to be about 0.6 for both the MPL and MES methodologies (Table 2.2), indicating that the data are somewhat overdispersed. Because $\widehat{v} < 1$, it is important to test whether this apparent overdispersive pattern is statistically significant. The observed absolute *Z*-statistic corresponding to the MES estimator, $|\widehat{v} - 1|/\sqrt{\widehat{var}(\widehat{v})} = |0.5975 - 1|/0.1362 \approx 2.96$ is larger than $z_{0.025} \approx 1.96$, indicating statistical significance at the commonly applied 5% level. A similar conclusion is reached from $\widehat{v}_{\text{MPL}}$ as well. Therefore, the ZICMP model is recommended over the simpler ZIP model for analyzing this dataset. Furthermore, we also compare the ZICMP model with the ZIP model with adjusted sandwich variance accounting for the cluster dependence (Table 2.2).

Based on our fitted ZICMP model and the corresponding p-values (Table 2.2), it turns out that *AUCmhF5_9yrs*, *AUCSodaOz5_9yrs*, and *ToothBrushingFeq.Per_DayAvg* have statistically significant

effects (*p*-values are all less than 0.01) on the excessive zero part for 9-year-old children data for both the MPL and MES methodologies. According to the signs of the coefficients of these model terms, frequent tooth brushing and greater daily fluoride intake are protective against the development of caries, whereas soda pop intake is a risk factor for the same. *HomeFluorideppm.Avg* is the one which is a moderately significant factor for both the count part and the excessive zero part (just above 5% level) with the MPL method; however, the message is mixed. The result for the count part makes clinical sense and indicates that the presence of fluoride in tap water might reduce the severity of caries. Also noteworthy is that the data from the same mouth exhibited low correlation ($\widehat{\rho} \approx 0.27$, for the count part and $\widehat{\delta} \approx 0.11$, for the excessive zero part).

The standard ZIP (zero-inflated Poisson) model, which operates under the independence assumption, yields a different set of significant factors for both count and excessive zero parts. In addition to the three significant factors based on our marginal ZICMP model, Gender(Male=1), FluorideTreatmentPast6monthAvg, and HomeFluorideppm.Avg have significant effects in the count part (p-values < 0.01). Thus, overall, the significance results from the simpler ZIP model appear to be a bit too optimistic. This may be due, in part, to the fact that the ZIP analysis did not account for (positive) correlations within the cluster members and over-dispersion of the data. This leads to the consequence that the variance of the covariate effects are underestimated, leading to an inflated Z-statistic (and low p-value). On the other hand, the ZIP model with the adjusted sandwich variance obtained using a similar formula as (14) identifies the same set of significant factors as the ZICMP model (perhaps with the exception of ToothBrushingFeq.Per_DayAvg, which is borderline significant under the MES method). This consequence is natural because the ZIP with the adjusted sandwich variance reflects the dependency of data. However, dispersion characteristics cannot be captured by a ZIP model even with the adjusted sandwich variance and may lead to biased inference. Indeed we verify this to be the case in a simulation study in the next section.

It is perhaps worth mentioning that the CES values were all less than or equal to 10 because there were five surfaces for each tooth. Thus, use of a truncated ZICMP, say, may be more appropriate. However, we have calculated the probability of a response y exceeding 10 under the fitted model and found it to be too small to make a practical difference in this analysis.

2.2.2 An application to maize hybrids data

We also apply our marginal ZICMP methods to a next generation sequencing (NGS) dataset to demonstrate a case of zero inflated, clustered count data, with underdispersion. This dataset emerges from a maze hybrids experiment (Paschold et. al, 2014). A complete analysis of this dataset from a biological standpoint is not intended here which consists of 39,656 gene IDs with four different genotypes (B73, B73 $\times$ Mo17, Mo17 $\times$ B73 and Mo17), four different tissues of each experimental unit (in this case, a certain genotype of a maze) and four biological replications. Since four tissues are harvested from the same root, there could exist some correlation among tissues belonging to the corresponding root. Therefore, this data is clustered. Out of all gene IDs, "GRMZM2G042361" is selected for an providing an illustrative example of zero-inflation with underdispersion. For this specific gene ID, we have 64 observations (read counts) including 37 zeros, 23 ones, 3 twos and 1 three.

Both the MPL and MES methods are applied to fit a marginal ZICMP model to the data. Since differences in total numbers of read counts over genes exist across biological sample units or different lanes, we need to account for this additional characteristic of NGS data in our model. Hence, we include the total read counts as an offset term into our regression model for a normalization across the biological samples. Therefore, our count part link function is modified as $log\,\lambda = \text{genotype} + log(\text{offset})$. The number of clusters is not deemed to be large enough for us to use the normal based confidence interval calculations. Instead, we report the point estimates along with a first order bootstrap confidence interval using the cluster bootstrap scheme described in the previous section with $B = 100$.

The dispersion estimates $\hat{\upsilon} \approx 2$ for both methods (Table 2.3) and indicate that the expression data for GRMZM2G042361 is significantly underdispersed since the bootstrap confidence intervals do not include the value 1. All the coefficients for genotype effects are similar in both the MPL and MES methods. Only the Mo17 genotype has a significant effect on this specific gene ID for the count part since the corresponding bootstrap confidence interval excludes zero for both MPL and MES confidence intervals. Note that, for a full scale analysis of this dataset, additional considerations such as multiple hypotheses corrections need to be taken into account.

2.3 Simulation Studies

We perform two different sets of simulations to study the finite sample performance of our methodology. The first simulation study is guided by the dental data analyzed in the previous section. Here we study the bias and variance of our MPL and MES estimators as well as the performance of the adjusted sandwich based variance estimator and the bootstrap based variance estimator, respectively. Performances of the estimators based on ZIP model and both variance estimators are also included for comparison. The second simulation study is guided by a dataset on airfreight breakage which has only one covariate; however, the covariate is a subject level (rather than cluster level) covariate. In addition, we are able to study the effect of increasing the number of clusters on the performance of these estimators.

2.3.1 Simulation guided by the dental data

The CES dataset of the nine-year-old children from the Iowa Fluoride Study (Levy et al., 2003) is described in detail in the previous section, which is also used for application of our marginal ZICMP model. The present large simulation study is guided by that dataset. We generate the clustered CES scores using a correlated ZICMP regression model with four cluster level covariates for both parts of the model. These covariates were the significant factors (based on results from Section 3) for the zero part: AUCmhF5_9yrs, AUCSodaOz5_9yrs and ToothBrushingFeq.Per_DayAvg except HomeFluorideppmAvg which was borderline significant for both the count and the zero parts based on the MPL analysis. Noisy versions of these covariate vectors resampled from the original dataset were used to generate the CES scores using the subject specific parameters through the links explained in Section 2. We use parameter values $\beta = (1.00, 0.01, -0.01, -0.13, -0.16)$ for the count part, $\gamma = (2.00, 0.70, -0.07, 0.56, -0.30)$ for the zero part and $v = 0.6$. These are close to both MPL and MES estimates obtained for the dental data in Section 3.

In order to keep the computational burden in check, the total number of clusters, N, is taken to be 200 and a constant cluster size of $n_i = 15$ is used for all the clusters. That is, $X_{i,\beta} = X_{i,\gamma}$ is a 15×5 matrix including an intercept term for each of the count and zero parts. Following Kong et al. (2014), correlated Bernoulli variables to generate the zero values are simulated using the Cholesky decomposition of a

compound symmetric correlation matrix with a common correlation coefficient $\widetilde{\delta}$, whereas the correlated count (CMP) data are generated by the inverse CDF transformation technique starting with a multivariate normal distribution with zero mean and a compound symmetric correlation matrix with a common correlation $\widetilde{\rho}$. We consider both low ($\widetilde{\rho} = \widetilde{\delta} = 0.2$) and high ($\widetilde{\rho} = \widetilde{\delta} = 0.8$) intra-cluster correlation cases.

For each setting, we create 100 datasets, calculate the MPL and MES estimates for each dataset, and obtain the empirical bias and standard error for each parameter estimator. The adjusted sandwich variance estimates are calculated for the MPL method and the variance estimates for the MES estimators are obtained through the bootstrap scheme (Field and Welsh, 2007) based on 100 bootstrap resamples as detailed in Section 2. ZIP estimates with their asymptotic variance and the adjusted sandwich variance estimates are also obtained for each Monte Carlo dataset by applying the $pscl$ R package and using an analogous adjusted sandwich variance formula as (14), respectively.

Estimators obtained from the ZIP model have larger biases than both the MPL and MES estimators of the ZICMP model (Table 1.4). This is more notable in high intra-cluster correlation case. The bias for the high intra-correlation case is larger for almost all estimators compared to the low intra-cluster case in both the ZICMP and ZIP models, as expected.

The estimators based on the simpler ZIP model are accompanied by Hessian-based standard errors, SE (pscl), obtained by the 'zeroinfl' function in pscl R package. For the low correlation case, these are not too different from the true standard errors for both count and zero parts. However, in the high intra-cluster correlation case, ZIP standard errors (SE (pscl)) are considerably smaller than the true ones. This implies that the Hessian-based standard errors are deflated which leads to a more liberal interpretation of p-values. This happens because the ZIP estimators do not account for the dependency of data in a cluster. This issue turns out to be more apparent for the high intra-cluster correlation case. On the other hand, the adjusted sandwich variance estimators tend to be closer to the true standard errors (SE). Thus, the inference from a ZIP model along with an adjusted sandwich variance has an ability to account for the clustering characteristic of data but still lacks the ability to handle data dispersion (under or over). This aspect may

causes prominently larger bias, especially in the count part, which may lead to incorrect inference as shown by the probability-probability (p-p) plots (Figure 2.2 and Figure 2.3).

While the MES estimators yield smaller biases and true standard errors (SE) than MPL estimates (Table 2.4), they need to use the bootstrap-based standard errors for variance estimates which consumes a considerable amount of computational efforts.

In order to study the performance of the resulting inferences of the effects of covariates/factors, we created the p-p plots where we plot the targeted nominal coverage of a confidence interval in the horizontal axis and the corresponding true coverage, as measured by the Monte Carlo simulation, in the vertical axis. Thus, a diagonal p-p plot would indicate that the asymptotic normal approximation to various estimators is accurate so we can have proper inferences using them. Overall, we noticed that all the p-p plots obtained from both the MPL and MES methods are relatively close to the solid reference lines (see Figures 2.2 and 2.3) even for the high correlation case. However, none of the p-p plots based on the ZIP model with standard variance estimates is very linear even in the low correlation case. As mentioned earlier, the situation improves when we use the adjusted sandwich variance with the ZIP model. Nevertheless, the p-p plots for most of the regression parameters still exhibit varying extent of under coverage. Thus, the ZIP model may not be a satisfactory method for analyzing zero-inflated clustered data with overdispersion.

2.3.2 Simulation guided by the airfreight breakage data

In this simulation, we investigate the performance of our ZICMP marginal model with a subject (observation) level covariate by building a simulation plan around the airfreight breakage data (Kutner, Nachtsheim and Neter, 2003, page 35, Exercise 1.21) which consists of 10 observations and one scalar covariate. A CMP model for this data was fit by Sellers and Shmueli (2010), which yielded parameter estimates of $\beta = (13.8,\ 1.3)^T$ and $v = 5.7818$. Going forward, we use the same parameter values for generating the count part of our data, with covariates X_β resampled from the set of scalar covariates in the original dataset (to match the desired number of observations). The zero-inflated part is generated by a regression model as described in Section 2, with the same set of covariates, i.e., $X_\gamma = X_\beta$, but with the

regression parameters $\gamma = (2, -3)^T$. For generating clustered ZICMP data, we need to generate correlated zeros, as well as, correlated counts. These are generated as explained in the Section 4.1

In this simulation, we consider three different combinations of number of clusters and the cluster size, namely, $N = 30$ with $n = 20$, $N = 50$ with $n = 30$, and $N = 75$ with $n = 15$. For each condition, we generate data with low $(\widetilde{\rho} = \widetilde{\delta} = 0.2)$ or high $(\widetilde{\rho} = \widetilde{\delta} = 0.8)$ correlations within each cluster. Both MPL and MES methods are applied to each of the 100 simulated dataset and the results are averaged to compute the empirical bias and variances of our estimators. We also used the bootstrap to compute variance estimates for both estimators in addition to the adjusted sandwich variance estimate for the MPL estimator. In order to keep the computational resources in check, we have used a modest number of bootstrap resamples ($= 100$) which is still deemed to be sufficient for our purpose. As mentioned earlier, in order to calculate bootstrap variance, we resample 100 times at the cluster level with replacement so that the correlation structures are preserved within a cluster. Finally, bootstrap variance estimates are given by the empirical variances of the parameter estimates obtained for the 100 bootstrap resamples. The results for $N = 50$ are provided in Table 1.5; results for the other two cases are placed in the Tables 2.6 and 2.7.

Table 2.6 results show that, in the case of $N = 30$, the estimators obtained from both MPL and MES methods have comparable performances in terms of bias and standard errors for both low and high intra-cluster correlation cases. For the low intra-cluster correlation case, bootstrap standard errors of both MPL and MES estimators match the true standard errors fairly well. However, in the high correlation case, the accuracy of the bootstrap standard errors worsens in both the MPL and the MES methods. Similarly, the adjusted sandwich standard errors based on the MPL method are fairly close to the true standard errors in the low intra-cluster correlation case, but not in the high correlation case. The bias terms for both MPL and MES methods are similar to each other and the bias tends to be larger in the high intra-cluster correlation case, as expected.

When the number of clusters increases to 50 (Table 2.5), the variance results were again comparable for the two sets of estimators in the case of both low and high intra-cluster correlations. In addition, the bootstrap based standard errors for both sets of estimators are very close to the true standard errors in both low and high intra-cluster correlation cases. Moreover, the adjusted sandwich standard errors

based on the MPL method are quite comparable to the bootstrap standard errors in both low and high correlation cases, even though the bootstrap estimates are slightly closer to the true standard errors.

When the number of clusters N further increases to 75 in Table 2.7, the performance improves across the board. From the results based on all the three scenarios, both the MPL method and MES algorithm have similar performances with respect to bias and standard errors. Note, however, that the MPL method is generally easier to implement and comes with a closed form sandwich variance estimate. The standard errors obtained using the bootstrap method appear to be reasonably close to the true SE as obtained by the Monte Carlo method; the estimates obtained from the adjusted sandwich formula for the MPL estimator can be adequate when the number of clusters is large.

We would like to point out that the biases for the intercept terms from the count parts based on these two simulations (Table 2.5 and Table 2.6) appear to be large compared to those for the other terms. In fact, the true values of the intercept parameters are relatively large compared to the other regression coefficients and consequently the relative biases of the intercept terms are comparable to those for the other terms.

2.4 Discussion

The CMP model has received a great deal of attention in recent years in many fields of application. In particular, the article by Shmueli et al. (2005) advocating the use of CMP distributions has already been cited 165 times according to Google Scholar (accessed September 5, 2015). While Sellers and Shmueli (2010) developed regression modeling for CMP distributed data, in this chapter we provide two significant extensions of the CMP methodology for making frequentist inference, thereby making this applicable to a greater variety of problems. Our version of the methodology can handle excessive zeros (zero-inflation) in the data and also when the data are clustered so that not all observations are independent. In particular, we have analyzed a dataset from the Iowa Fluoride Study using our model and show that more reliable inference can be obtained using it than the ZIP regression.

In this chapter, we have introduced two methods to fit a ZICMP marginal model with clustered data that has over or under dispersion. In our simulations, the MES method produced slightly more efficient

estimators through the use of a working variance-covariance matrix like the GEE. However, the corresponding variance estimates are computationally more expensive. The MPL method, on the other hand, affords a close form variance estimator. Like any other numerical optimization/estimating equation based methods, these methods may have convergence issues for certain datasets and changing the initial values and the optimization method (e.g., use a different method rather than the default in the R function 'optim') or the updating scheme (e.g., acceleration constant, Cesàro updating) may help the situation.

We also demonstrated that a cluster bootstrap method is capable of producing reasonable variance estimates for both sets of estimators through two different simulations. With respect to this, it is important for the reader to note that certain R packages that are directly able to calculate the sandwich variances may not work as well as using bootstrap to estimate the variances. We also obtain a theoretical form of the asymptotic variance covariance matrix of the MES estimators that explains the variability in the estimation of the indicators of the zero part. However, it is not possible to obtain an empirical analogue of this for general clustered data, since we do not know the exact joint likelihood of the cluster-correlated observations. On the other hand, we can obtain a valid sandwich variance estimators for the MPL method even for clustered data by utilizing the independence of the cluster sums of the corresponding estimating functions.

The two real data examples demonstrate the scope of applications of our methodology to diverse fields and it is our hope that with time more applications to these models for clustered count data with zero inflation and wide range of dispersion will be discovered.

Table 2.1

The descriptions of eight different covariates including nondietary and dietary factors as a part of the dental dataset from the Iowa Fluorida Study

Gender	Gender of the child; Male is coded as 1.
DentalExamAge	Age in years at the time of the dental examination.
AUCmhF5_9yrs	Daily Fluoride intake (mg) from water, other beverages and selected foods, ingested dentifrice and fluoride supplements. Computed using AUC trapezoidal method using all available data within the time span 5 to 9 years.
AUCSodaOz5_9yrs	Daily soda pop intake (oz.) computed using AUC trapezoidal method using all available data within the time span 5 to 9 years.
ToothBrushingFeq.Per_DayAvg	Average of all tooth brushing frequencies reported for the period 5 to 9 years.
DentalVisitPast6monthAvg	Proportion of times a dental visit was indicated with each individual point assessing the previous 6 months.
FluorideTreatmentPast6monthAvg	Average proportion of times a professional dental fluoride treatment was received with each individual point assessing the previous 6 months.
HomeFluorideppmAvg	Average home tap water fluoride level for all returned questionnaires for the period 5 to 9 years.

Table 2.2

Results for the data of the nine-year-old children from the Iowa Fluoride Study

ZICMP (MPL)						
	Counts	SE (Adj_SW)	P-value	Zero-inflation	SE (Adj_SW)	P-value
Intercept	1.2571	0.4708	0.008**	2.1507	0.7839	0.006**
Gender(Male=1)	-0.0154	0.0545	0.777	0.1363	0.1235	0.270
DentalExamAge	-0.0360	0.0394	0.361	-0.0791	0.0795	0.320
AUCmhF5_9yrs	0.0137	0.1087	0.900	0.7741	0.2127	<0.0001**
AUCSodaOz5_9yrs	-0.0120	0.0116	0.302	-0.0721	0.0241	0.003**
ToothBrushingFeq.Per_DayAvg	-0.1476	0.0612	0.016*	0.5607	0.1249	<0.0001**
Dental VisitPat6moth Avg	0.1161	0.1522	0.446	-0.5297	0.2785	0.057
FluorideTreatmentPast6monthAvg	0.1010	0.1264	0.424	-0.0186	0.1951	0.924
HomeFluorideppm.Avg	-0.1551	0.0801	0.053	-0.3179	0.1710	0.063
v	0.6027	0.1060				
ZICMP (MES)						
	Counts	SE (BS)	P-value	Zero-inflation	SE (BS)	P-value
Intercept	0.9273	0.6253	0.138	2.2401	0.8463	0.008**
Gender(Male=1)	-0.0134	0.0548	0.807	0.1408	0.1266	0.266
DentalExamAge	-0.0081	0.0591	0.891	-0.0884	0.0902	0.327
AUCmhF5_9yrs	0.0135	0.1136	0.905	0.7041	0.2293	0.002**
AUCSodaOz5_9yrs	-0.0098	0.0119	0.409	-0.0704	0.0245	0.004**
ToothBrushingFeq.Per_DayAvg	-0.1225	0.0649	0.059	0.5664	0.1371	<0.0001**
Dental VisitPat6moth Avg	-0.0604	0.1745	0.729	-0.4932	0.3537	0.163
FluorideTreatmentPast6monthAvg	0.0765	0.1228	0.533	-0.0643	0.2138	0.7636
HomeFluorideppm.Avg	-0.1580	0.0854	0.064	-0.2877	0.1892	0.1283
v	0.5975	0.1362				
ZIP						
	Counts	SE (pscl, adj_sw)	P-value(pscl, adj_sw)	Zero-inflation	SE (pscl, adj_sw)	P-value(pscl, adj_sw)
Intercept	0.5737	0.2994, 0.4886	0.055, 0.2403	2.1560	0.4469, 0.7578	<0.0001**, 0.0044**
Gender(Male=1)	-0.0174	0.0369, 0.0690	0.639, 0.8017	0.1540	0.0648, 0.1224	0.018*, 0.2084
DentalExamAge	0.0771	0.0296, 0.0480	0.009**, 0.1080	-0.0814	0.0451, 0.0773	0.071, 0.2928
AUCmhF5_9yrs	-0.0185	0.0688, 0.1355	0.788, 0.8911	0.4701	0.1129, 0.1963	<0.0001**, 0.0166*
AUCSodaOz5_9yrs	-0.0055	0.0075, 0.0146	0.466, 0.7073	-0.0560	0.0129, 0.0235	<0.0001**, 0.0114*
ToothBrushingFeq.Per_DayAvg	-0.0745	0.0418, 0.0708	0.075, 0.2926	0.6070	0.0697, 0.1221	<0.0001**, < 0.0001**
Dental VisitPat6moth Avg	0.0962	0.1045, 0.1791	0.358, 0.5912	-0.2886	0.1678, 0.2862	0.085, 0.3133
FluorideTreatmentPast6monthAvg	0.0029	0.0677, 0.1397	0.967, 0.9837	-0.2490	0.1135, 0.2055	0.028*, 0.2256
HomeFluorideppm.Avg	-0.1863	0.0477, 0.0961	<0.0001**, 0.0525	-0.1573	0.0768, 0.1503	0.041*, 0.2952

*: $0.01 <$ p-value < 0.05, **: p-value < 0.01

The result includes the Maximum Pseudo Likelihood (MPL) estimators with adjusted sandwich standard error (Adj_SW) and the Modified Expectation-Solution (MES)

estimators with a $B = 500$ size bootstrap standard error (BS). Results from a standard zero-inflated Poisson analysis with a Hessian-based standard error from the pscl

package (SE (pscl)) are also shown for comparison.

Table 2.3

Results for the GRMZM2G042361 gene from the maze hybrids data: Parameter estimates are reported along with cluster bootstrap based (nonasymptotic) confidence intervals (BS_CI).

MPL				
	Count part	BS_CI	Zero part	BS_CI
Intercept	-16.6529	(-16.70, -15.93)	-11.2535	(-24.46, -2.04)
B73 $\times$ Mo17	0.5264	(-0.29, 0.71)	-1.9622	(-18.75, 6.86)
Mo17	1.4999	(0.66, 3.24)	-3.9165	(-10.27, 15.54)
Mo17 $\times$ B73	0.6317	(-1.22, 2.97)	-4.9595	(-14.67, 13.64)
v	2.1020	(1.86, 4.93)		
MES				
	Count part	BS_CI	Zero part	BS_CI
Intercept	-16.6490	(-16.69, -15.93)	-11.2748	(-24.46, -2.04)
B73 $\times$ Mo17	0.5150	(-0.28, 0.71)	-1.9740	(-18.75, 6.85)
Mo17	1.5027	(0.65, 3.23)	-3.8774	(-10.26, 15.53)
Mo17 $\times$ B73	0.6273	(-1.22, 2.96)	-4.9362	(-14.67, 13.63)
v	2.1056	(1.86, 4.93)		

Table 2.4

Empirical bias and variance of our estimators in a simulation study guided by the dental data of nine-year-old children from the Iowa Fluoride Study: The number of clusters is 200; the size of each cluster is 15. Each entry is based on 100 Monte Carlo iterations. The performances of ZIP estimators are also added.

MPL			Low intraclass correlation			High intraclass correlation		
		True	Bias	SE	SE (Adj_SW)	Bias	SE	SE (Adj_SW)
Count part	Intercept	1.00	0.0645	0.2040	0.2835	0.3191	0.5564	0.5916
	AUCmhF5_9yrs	0.01	-0.0006	0.1388	0.1274	0.0462	0.3889	0.2966
	AUCSodaOz5_9yrs	-0.01	-0.0011	0.0201	0.0167	-0.0076	0.0558	0.0430
	ToothBrushingFeq.Per_DayAvg	-0.13	-0.0117	0.0920	0.0879	-0.0290	0.2439	0.2086
	HomeFluorideppm.Avg	-0.16	0.0080	0.1000	0.0999	-0.0631	0.3174	0.2453
	v	0.60	0.0427	0.1030	0.1277	0.1892	0.2052	0.2461
			Low intraclass correlation			High intraclass correlation		
		True	Bias	SE	SE (Adj_SW)	Bias	SE	SE (Adj_SW)
Zero part	Intercept	2.00	0.0533	0.4238	0.4117	0.0274	0.7473	0.8477
	AUCmhF5_9yrs	0.70	0.0044	0.3068	0.3156	0.0356	0.5244	0.6381
	AUCSodaOz5_9yrs	-0.07	0.0091	0.0489	0.0433	0.0129	0.0879	0.0902
	ToothBrushingFeq.Per_DayAvg	0.56	-0.0417	0.2324	0.2235	-0.0157	0.4599	0.4592
	HomeFluorideppm.Avg	-0.30	-0.0006	0.2491	0.2317	0.0328	0.5282	0.4942

MES			Low intraclass correlation			High intraclass correlation		
		True	Bias	SE	SE (BS)	Bias	SE	SE (BS)
Count part	*Intercept*	1.00	-0.0045	0.0687	0.1016	-0.0068	0.1481	0.1593
	AUCmhF5_9yrs	0.01	-0.0025	0.0487	0.0820	0.0133	0.1235	0.1282
	AUCSodaOz5_9yrs	-0.01	-0.0003	0.0074	0.0099	-0.0019	0.0177	0.0182
	ToothBrushingFeq.Per_DayAvg	-0.13	-0.0006	0.0353	0.0531	-0.0033	0.0717	0.0862
	HomeFluorideppm.Avg	-0.16	0.0032	0.0359	0.0558	-0.0154	0.0934	0.0982
	v	0.60	-0.0069	0.0539	0.0686	-0.0034	0.0640	0.0430
			Low intraclass correlation			High intraclass correlation		
		True	Bias	SE	SE (BS)	Bias	SE	SE (BS)
Zero part	*Intercept*	2.00	0.0240	0.1537	0.1623	0.0115	0.3154	0.3664
	AUCmhF5_9yrs	0.70	-0.0046	0.1061	0.1182	0.0048	0.2075	0.2734
	AUCSodaOz5_9yrs	-0.07	0.0027	0.0161	0.0158	0.0072	0.0335	0.0358
	ToothBrushingFeq.Per_DayAvg	0.56	-0.0140	0.0820	0.0816	-0.0052	0.1786	0.1913
	HomeFluorideppm.Avg	-0.30	-0.0026	0.0767	0.0877	0.0176	0.1798	0.2022

ZIP			Low intraclass correlation			High intraclass correlation		
		True	Bias	SE	SE (*pscl*, Adj_SW)	Bias	SE	SE (*pscl*, Adj_SW)
Count part	*Intercept*	1.00	0.7203	0.2361	0.1665, 0.2038	0.7002	0.5395	0.1922, 0.4104
	AUCmhF5_9yrs	0.01	-0.0063	0.1782	0.1374, 0.1646	0.0721	0.4532	0.1597, 0.3348
	AUCSodaOz5_9yrs	-0.01	-0.0044	0.0272	0.0182, 0.0215	-0.0102	0.0654	0.0231, 0.0484
	ToothBrushingFeq.Per_DayAvg	-0.13	-0.0566	0.1253	0.0919, 0.1123	-0.0620	0.2790	0.1067, 0.2312
	HomeFluorideppm.Avg	-0.16	-0.0418	0.1372	0.1038, 0.1265	-0.1092	0.3700	0.1249, 0.2710
			Low intraclass correlation			High intraclass correlation		
		True	Bias	SE	SE (*pscl*, Adj_SW)	Bias	SE	SE (*pscl*, Adj_SW)
Zero part	*Intercept*	2.00	0.0596	0.4234	0.3117, 0.4086	0.0372	0.7874	0.3220, 0.8389
	AUCmhF5_9yrs	0.70	-0.0069	0.3026	0.2446, 0.3122	0.0335	0.5423	0.2532, 0.6292
	AUCSodaOz5_9yrs	-0.07	0.0114	0.0490	0.0333, 0.0429	0.0135	0.0880	0.0354, 0.0882
	ToothBrushingFeq.Per_DayAvg	0.56	-0.0258	0.2333	0.1671, 0.2219	-0.0047	0.4678	0.1733, 0.4534
	HomeFluorideppm.Avg	-0.30	0.0174	0.2469	0.1801, 0.2288	0.0388	0.5456	0.1957, 0.4911

SE: Monte Carlo; SE (BS): bootstrap estimated standard error, SE (Adj_SW): square root of adjusted sandwich variance estimate,
SE (*pscl*): standard errors obtained from the Hessian matrix.

Table 2.5

Empirical bias and variance of our ZICMP estimators in a simulation study guided by the airfreight breakage data : The number of clusters is 50; the size of each cluster is 30. Each entry is based on 100 Monte Carlo iterations.

MPL									
		Low intraclass correlation				High intraclass correlation			
	True	Bias	SE	SE (BS)	SE (Adj_SW)	Bias	SE	SE (BS)	SE (Adj_SW)
β_0	13.8	0.2484	0.8425	0.8341	0.8144	0.9945	2.1666	2.2882	2.2576
β_1	1.3	0.0260	0.0847	0.0816	0.0797	0.0965	0.2088	0.2173	0.2139
γ_0	2	0.0130	0.1703	0.1727	0.2089	0.0691	0.3803	0.3469	0.4054
γ_1	-3	-0.0384	0.1538	0.1622	0.1918	-0.0726	0.3309	0.3358	0.3788
v	5.7818	0.1040	0.3490	0.3467	0.3384	0.4152	0.9048	0.9540	0.9403

MES							
		Low intraclass correlation			High intraclass correlation		
	True	Bias	SE	SE (BS)	Bias	SE	SE (BS)
β_0	13.8	0.2498	0.8430	0.8419	0.9978	2.1712	2.1816
β_1	1.3	0.0254	0.0848	0.0838	0.0952	0.2069	0.3223
γ_0	2	0.0151	0.1704	0.1702	0.0732	0.3910	0.4453
γ_1	-3	-0.0381	0.1534	0.1626	-0.0763	0.3356	0.3394
v	5.7818	0.1040	0.3491	0.3493	0.3937	0.9372	0.9446

SE: Monte Carlo; SE (BS): bootstrap estimated standard error, SE (Adj_SW): square root of adjusted sandwich variance estimate, SE (*pscl*): standard errors obtained from the Hessian matrix.

Table 2.6

Empirical bias and variance of our ZICMP estimators in a simulation study guided by the airfreight breakage data: The number of clusters is 30; the size of each cluster is 20. Each entry is based on 100 Monte Carlo iterations.

MPL									
		Low intraclass correlation				High intraclass correlation			
	True	Bias	SE	SE (BS)	SE (Adj_SW)	Bias	SE	SE (BS)	SE (Adj_SW)
β_0	13.8	0.2368	1.2988	1.2472	1.2059	1.3290	3.4226	2.8328	2.8935
β_1	1.3	0.0236	0.1388	0.1240	0.1200	0.1225	0.3239	0.2696	0.2741
γ_0	2	-0.0008	0.2553	0.2509	0.3063	0.0485	0.3975	0.6196	0.5227
γ_1	-3	-0.0174	0.2497	0.2428	0.2825	-0.0763	0.3960	0.6763	0.4995
v	5.7818	0.1004	0.5514	0.5201	0.5031	0.5567	1.4247	1.1796	1.2053

MES							
		Low intraclass correlation			High intraclass correlation		
	True	Bias	SE	SE (BS)	Bias	SE	SE (BS)
β_0	13.8	0.2383	1.3009	1.2592	1.3175	3.4271	2.3657
β_1	1.3	0.0230	0.1376	0.1395	0.1232	0.3217	0.4115
γ_0	2	0.0062	0.2553	0.2512	0.0587	0.4229	0.7440
γ_1	-3	-0.0141	0.2546	0.2397	-0.1044	0.3932	0.4364
v	5.7818	0.1005	0.5515	0.5338	0.5533	1.4188	0.9935

SE: Monte Carlo based empirical standard error; SE (BS): bootstrap estimated standard error, SE (Adj_SW): square root of adjusted sandwich variance estimate.

Table 2.7

Empirical bias and variance of our ZICMP estimators in a simulation study guided by the airfreight breakage data: The number of clusters is 75; the size of each cluster is 15. Each entry is based on 100 Monte Carlo iterations.

MPL									
		Low intraclass correlation				High intraclass correlation			
	True	Bias	SE	SE (BS)	SE (Adj_SW)	Bias	SE	SE (BS)	SE (Adj_SW)
β_0	13.8	0.0991	0.9378	0.8927	0.8666	0.1844	1.8638	1.9151	1.8809
β_1	1.3	0.0105	0.0827	0.0874	0.0855	0.0177	0.1773	0.1799	0.1776
γ_0	2	0.0046	0.1683	0.1713	0.2094	0.0448	0.2876	0.2818	0.3433
γ_1	-3	-0.0223	0.1895	0.1769	0.2089	-0.0435	0.3031	0.2834	0.3359
v	5.7818	0.0432	0.3874	0.3710	0.3608	0.0774	0.7797	0.7960	0.7827

MES							
		Low intraclass correlation			High intraclass correlation		
	True	Bias	SE	SE (BS)	Bias	SE	SE (BS)
β_0	13.8	0.1004	0.9380	0.8607	0.1993	1.8663	1.7956
β_1	1.3	0.0100	0.0821	0.0860	0.0160	0.1783	0.2519
γ_0	2	0.0072	0.1679	0.1676	0.0382	0.2604	0.2848
γ_1	-3	-0.0233	0.1891	0.1731	-0.0405	0.2927	0.2826
v	5.7818	0.0433	0.3874	0.3587	0.0825	0.7873	0.7761

SE: Monte Carlo based empirical standard error; SE (BS): bootstrap estimated standard error, SE (Adj_SW): square root of adjusted sandwich variance estimate.

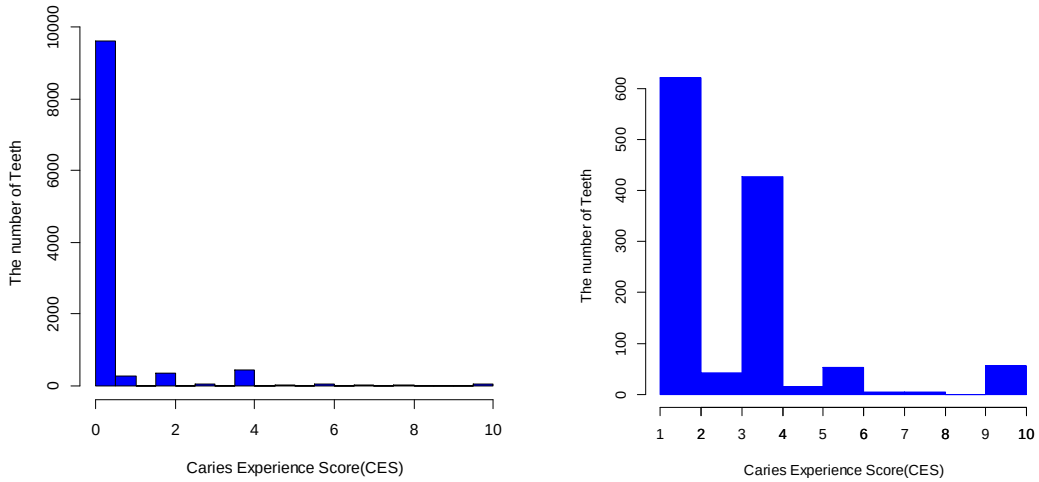
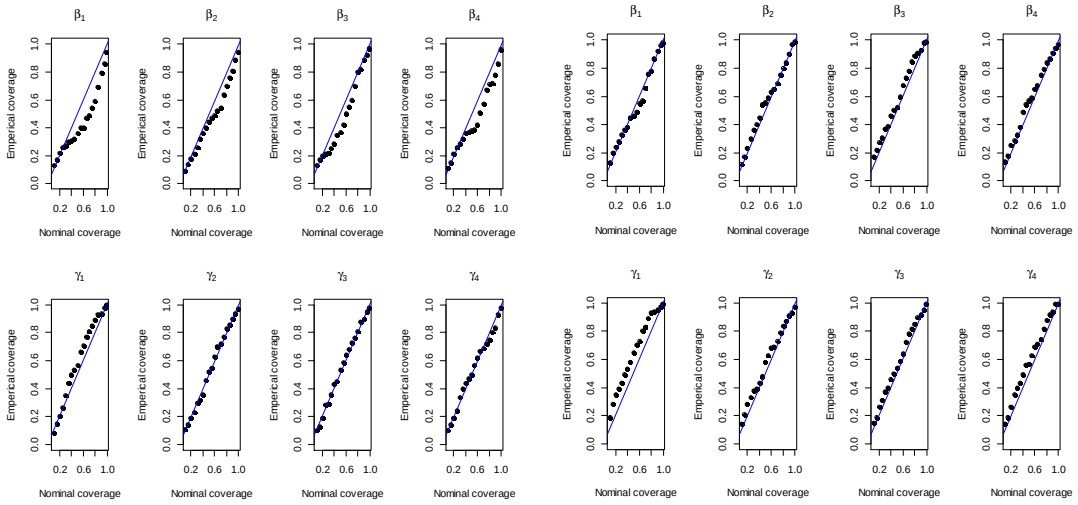


Figure 2.1. Summary plots of the data of the nine-year-old children from the Iowa Fluoride Study: Frequency histogram of caries experience scores (CES) summarized over all teeth and children in our sample (left panel), and the frequency histogram of CES excluding zero counts summarized over all teeth and children in our sample (right panel).

(a) Coverage probabilities based on ZICMP/MPL (b) Coverage probabilities based on ZICMP/MES



(c) Coverage probabilities based on ZIP (d) Coverage probabilities based on ZIP with Adj_SW

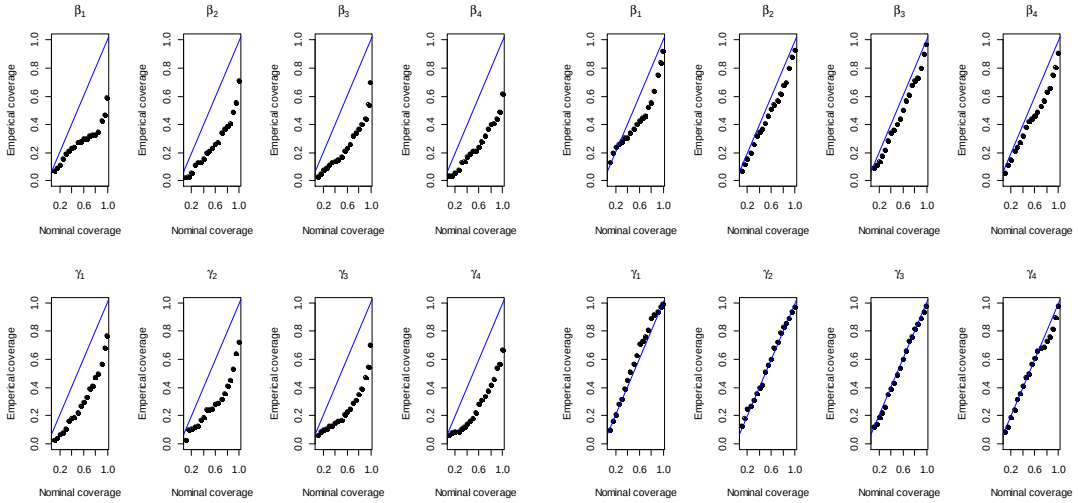
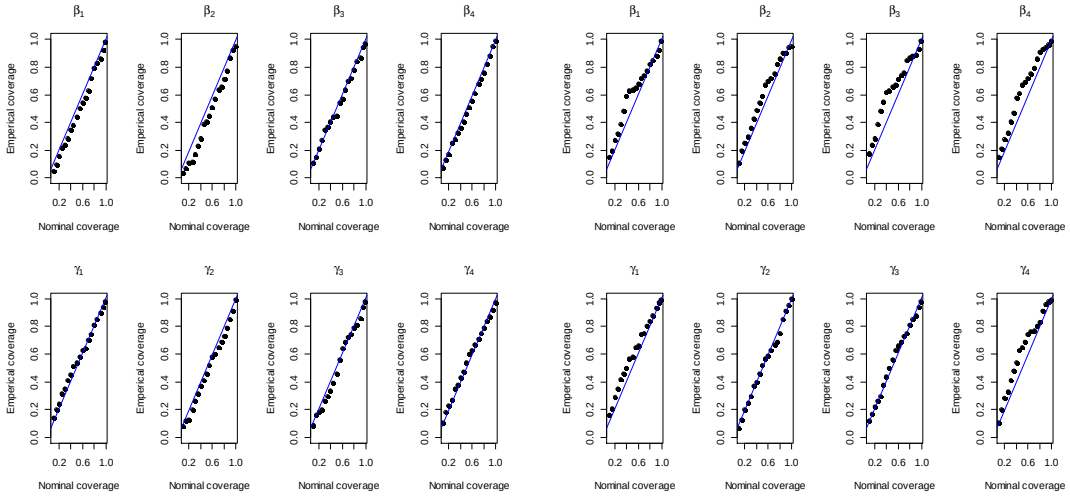


Figure 2.2. Empirical coverage of the confidence intervals in a simulation study guided by the dental data of nine-year-old children from the Iowa Fluoride Study. The p-p plots are for $N = 200$ and $n = 15$ when intra-cluster correlation is high. Three sets of plots are provided for the regression parameters corresponding to the four covariates: ZICMP/MPL (upper left panel), ZICMP/MES (upper right panel), ZIP (bottom left panel) and ZIP with Adj_SW (bottom right panel).

(a) Coverage probabilities based on ZICMP/MPL (b) Coverage probabilities based on ZICMP/MES



(c) Coverage probabilities based on ZIP (d) Coverage probabilities based on ZIP with Adj_SW

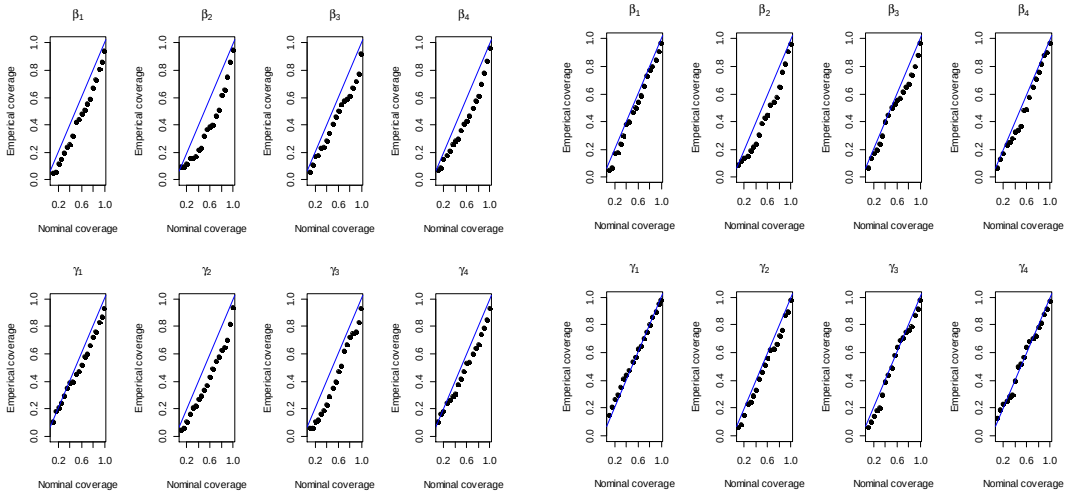


Figure 2.3. Empirical coverage of the confidence intervals in a simulation study guided by the dental data of nine-year-old children from the Iowa Fluoride Study. The p-p plots are for $N = 200$ and $n = 15$ when intra-cluster correlation is low. Three sets of plots are provided for the regression parameters corresponding to the four covariates: ZICMP/MPL (upper left panel), ZICMP/MES (upper right panel), ZIP (bottom left panel) and ZIP with Adj_SW (bottom right panel).

TECHNICAL DETAILS

Appendix 2.1: Estimating functions based on the MES algorithm for the zero degenerated part and the count part (CMP).

Equation (7) is the estimating function of the zero part and below is given for the derivative term with respect to γ.

$$\frac{\partial p_i}{\partial \gamma} = \frac{X_{\gamma,i} e^{X_{\gamma,i}\gamma}}{(1+e^{x_{\gamma,i}\gamma})^2},$$

and the derivative term for the count part from Equation (8) is given as

$$\frac{\partial E(y_i)^T}{\partial \beta} = \begin{pmatrix} \frac{\partial E(y_i)^T}{\partial \beta_1} \\ \frac{\partial E(y_i)^T}{\partial \beta_2} \\ \cdots \\ \frac{\partial E(y_i)^T}{\partial \beta_{p_\beta}} \end{pmatrix}_{p_\beta \times n_i} \quad \text{and each component is calculated as}$$

$$\frac{\partial E(y_{ij})}{\partial \beta_k} = x_{\beta,ijk} \left(\sum_{s=0}^{\infty} \frac{s^2 \lambda_{ij}^s}{(s!)^v} / Z(\lambda_{ij}, v) - \left[\sum_{s=0}^{\infty} \frac{s \lambda_{ij}^s}{(s!)^v} / Z(\lambda_{ij}, v) \right]^2 \right)$$

$$Var(y_{ij}) = \sum_{s=0}^{\infty} \frac{s^2 \lambda_{ij}^s}{(s!)^v} / Z(\lambda_{ij}, v) - \left[\sum_{s=0}^{\infty} \frac{s \lambda_{ij}^s}{(s!)^v} / Z(\lambda_{ij}, v) \right]^2.$$

Here, $x_{\beta,ijk}$is the k^{th}covariate corresponding to the count part in the j^{th}element within the i^{th}subject and $Var(y_{ij})$ is the j^{th}diagonal element of D_i.

Appendix 2.2: Estimating a dispersion parameter, v, based on the MPL method for a CMP distribution of y.

We estimating the dispersion parameter, v, from Equation (6) where

$$\frac{\partial \ell_{ij}^c}{\partial v} = \left(-log(y_{ij}!) + \sum_{s=0}^{\infty} \frac{\lambda_{ij}^s log(s!)}{(s!)^v} / Z(\lambda_{ij}, v) \right)(1-u_{ij}),$$

$$\frac{\partial \ell_{ij}^c}{\partial v^2} = \left(-\sum_{s=0}^{\infty} \frac{\lambda_{ij}^s (log(s!))^2}{(s!)^v} / Z(\lambda_{ij}, v) + \left[\sum_{s=0}^{\infty} \frac{\lambda_{ij}^s log(s!)}{(s!)^v} / Z(\lambda_{ij}, v) \right]^2 \right)(1-u_{ij}).$$

Appendix 2.3: Estimating a correlation coefficient, δ, based on the MES

algorithm for zero-degenerated distribution of u.

Estimation of the common off-diagonal correlation coefficient δ is carried out from Equation (9) leading to

$$\widehat{\delta} = \frac{\frac{1}{N*}\sum_{i=1}^{N}\sum_{s<t}\frac{(u_{is}-p_{is})(u_{it}-p_{it})}{\sqrt{p_{is}(1-p_{is})p_{it}(1-p_{it})}}}{\frac{1}{N_{total}}\sum_{i=1}^{N}\sum_{j=1}^{n_i}\frac{(u_{ij}-p_{ij})^2}{p_{ij}(1-p_{ij})}},$$

where $N* = \sum_{i=1}^{N} n_i(n_i-1)/2$ and $N_{total} = \sum_{i=1}^{N} n_i$. See, e.g., Kong et al. (2014).

Appendix 2.4: Estimating a correlation coefficient, ρ based on the MES algorithm for a CMP distribution of y.

Estimation of the common off-diagonal correlation coefficient ρ is carried out from Equation (10) leading to

$$\widehat{\rho} = \frac{\frac{1}{N*}\sum_{i=1}^{N}\sum_{s<t}\frac{(1-u_{is})(1-u_{it})(y_{is}-E(y_{is}))(y_{it}-E(y_{it}))}{\sqrt{Var(y_{is})Var(y_{it})}}}{\frac{1}{N_{total}}\sum_{i=1}^{N}\sum_{j=1}^{n_i}\frac{(1-u_{ij})^2(y_{ij}-E(y_{ij}))^2}{Var(y_{ij})}},$$

where $N* = \sum_{i=1}^{N}\sum_{s<t}(1-u_{is})(1-u_{it})$ and $N_{total} = \sum_{i=1}^{N}\sum_{j=1}^{n_i}(1-u_{ij})^2$. See, e.g., Kong *et al.* (2014).

Appendix 2.5: (a modified/adjusted) Sandwich covariance matrix of $\widehat{\beta}, \widehat{\gamma}$ and $\widehat{v}$

based on the MPL method.

The partial derivatives of ℓ with respect to β, γ and v from Equations (12) and (13) are given follows:

• For β,

$$\frac{\partial \ell}{\partial \beta_k} = \sum_{i=1}^{N}\sum_{j=1}^{n_i} I(y_{ij}=0) x_{\beta,ijk} \frac{-(1-p_{ij})/Z(\lambda_{ij},v)^2 \sum_{s=0}^{\infty} \frac{s\lambda_{ij}^s}{(s!)^v}}{p_{ij}+(1-p_{ij})/Z(\lambda_{ij},v)} +$$

$$\sum_{i=1}^{N}\sum_{j=1}^{n_i} I(y_{ij}\geq 1) x_{\beta,ijk}(y_{ij} - \sum_{s=0}^{\infty} \frac{s\lambda_{ij}^s}{(s!)^v}/Z(\lambda_{ij},v)).$$

$$\frac{\partial^2 \ell}{\partial \beta \partial \beta^T} = X_\beta^T diag\Big[I(y_{ij}=0) \times$$

$$\frac{\Big[\Big(2/Z(\lambda_{ij},v)^3 \sum_{s=0}^{\infty} \frac{s\lambda_{ij}^s}{(s!)^v} - (1-p_{ij})/Z(\lambda_{ij},v)^2 \sum_{s=0}^{\infty} \frac{s^2\lambda_{ij}^s}{(s!)^v}\Big)\Big\{p_{ij}+(1-p_{ij})/Z(\lambda_{ij},v)\Big\} - \Big\{\frac{(1-p_{ij})}{Z(\lambda_{ij},v)^2}\sum_{s=0}^{\infty}\frac{s\lambda_{ij}^s}{(s!)^v}\Big\}^2}{\Big\{p_{ij}+(1-p_{ij})/Z(\lambda_{ij},v)\Big\}^2}\Big] X_\beta -$$

$$X_\beta^T\, diag\Big[I(y_{ij}\geq 1)\Big[\sum_{s=0}^{\infty}\frac{s^2\lambda_{ij}^s}{(s!)^v}/Z(\lambda_{ij},v) \; - \Big(\sum_{s=0}^{\infty}\frac{s\lambda_{ij}^s}{(s!)^v}/Z(\lambda_{ij},v)\Big)^2\Big]\Big] X_\beta$$

where X_β is a $\Big(\sum_{i=1}^{N} n_i \times p_\beta\Big)$matrix.

• For γ,

$$\frac{\partial \ell}{\partial \gamma_k} = \sum_{i=1}^{N}\sum_{j=1}^{n_i} I(y_{ij}=0) x_{\gamma,ijk} \frac{p_{ij}(1-p_{ij})(1-1/Z(\lambda_{ij},v))}{p_{ij}+(1-p_{ij})/Z(\lambda_{ij},v)} - \sum_{i=1}^{N}\sum_{j=1}^{n_i} I(y_{ij}\geq 1) x_{\gamma,ijk} p_{ij}.$$

$$\frac{\partial^2 \ell}{\partial \gamma \partial \gamma^T} =$$

$$X_\gamma^T\, diag\Big[I(y_{ij}=0) \frac{p_{ij}(1-p_{ij})(1-1/Z(\lambda_{ij},v))(1-2\times p_{ij})(p_{ij}+(1-p_{ij})/Z(\lambda_{ij},v)) - p_{ij}^2(1-p_{ij})^2(1-1/Z(\lambda_{ij},v))^2}{\Big\{p_{ij}+(1-p_{ij})/Z(\lambda_{ij},v)\Big\}^2}\Big] X_\gamma -$$

$$X_\gamma^T\, diag\Big[I(y_{ij}\geq 1) p_{ij}(1-p_{ij})\Big] X_\gamma,$$

where X_γ is a $\Big(\sum_{i=1}^{N} n_i \times p_\gamma\Big)$matrix.

• For v,

$$\frac{\partial \ell}{\partial v} = \sum_{i=1}^{N}\sum_{j=1}^{n_i} I(y_{ij}=0) \frac{(1-p_{ij})\sum_{s=0}^{\infty}\frac{log(s!)\lambda_{ij}^s}{(s!)^v}/Z(\lambda_{ij},v)^2}{p_{ij}+(1-p_{ij})/Z(\lambda_{ij},v)} +$$

$$\sum_{i=1}^{N}\sum_{j=1}^{n_i} I(y_{ij}\geq 1)\Big(- log(y_{ij}!) + \sum_{s=0}^{\infty}\frac{\lambda_{ij}^s logs!}{(s!)^v}/Z(\lambda_{ij},v)\Big).$$

$$\frac{\partial^2\ell}{\partial v^2} = \sum_{i=1}^{N}\sum_{j=1}^{n_i} I(y_{ij}=0)\frac{\Big[\mathcal{H}_{ij}\times\Big\{p_{ij}+(1-p_{ij})/Z(\lambda_{ij},v)\Big\} - \Big\{(1-p_{ij})^2\sum_{s=0}^{\infty}\frac{log(s!)\lambda_{ij}^s}{(s!)^v}\Big\}^2/Z(\lambda_{ij},v)^4\Big]}{\Big\{p_{ij}+(1-p_{ij})/Z(\lambda_{ij},v)\Big\}^2} -$$

$$\sum_{i=1}^{N}\sum_{j=1}^{n_i} I(y_{ij}\geq 1)\Big[\sum_{s=0}^{\infty}\frac{(log(s!))^2\lambda_{ij}^s}{(s!)^v}/Z(\lambda_{ij},v) - \Big(\sum_{s=0}^{\infty}\frac{log(s!)\lambda_{ij}^s}{(s!)^v}/Z(\lambda_{ij},v)\Big)^2\Big],$$

where

$$\mathcal{H}_{ij} = \Big[2(1-p_{ij})\Big\{\sum_{s=0}^{\infty}\frac{log(s!)\lambda_{ij}^s}{(s!)^v}\Big\}^2/Z(\lambda_{ij},v)^3 - (1-p_{ij})\sum_{s=0}^{\infty}\frac{(log(s!))^2\lambda_{ij}^s}{(s!)^v}/Z(\lambda_{ij},v)^2\Big].$$

• For off-diagonal elements,

$$\frac{\partial^2\ell}{\partial\beta\partial\gamma^T} = X_\beta^T diag\Bigg[I(y_{ij}=0)$$

$$\frac{p_{ij}(1-p_{ij})\Big\{\sum_{s=0}^{\infty}\frac{s\lambda_{ij}^s}{(s!)^v}/Z(\lambda_{ij},v)^2\Big\}\Big\{(p_{ij}+(1-p_{ij})/Z(\lambda_{ij},v)) + (1-p_{ij})(1-1/Z(\lambda_{ij},v))\Big\}}{\Big\{p_{ij}+(1-p_{ij})/Z(\lambda_{ij},v)\Big\}^2}\Bigg]X_\gamma.$$

$$\frac{\partial^2\ell}{\partial\beta_k\partial v} = \sum_{i=1}^{N}\sum_{j=1}^{n_i} I(y_{ij}=0)x_{ijk,\beta}\frac{\Big\{\mathcal{W}_{ij}\times\Big\{p_{ij}+(1-p_{ij})/Z(\lambda_{ij},v)\Big\} - (1-p_{ij})^2/Z(\lambda_{ij},v)^4\sum_{s=0}^{\infty}\frac{s\lambda_{ij}^s}{(s!)^v}\sum_{s=0}^{\infty}\frac{\lambda_{ij}^s logs!}{(s!)^v}\Big\}}{\Big\{p_{ij}+(1-p_{ij})/Z(\lambda_{ij},v)\Big\}^2}$$

$$+\sum_{i=1}^{N}\sum_{j=1}^{n_i} I(y_{ij}\geq 1)x_{ijk,\beta}\Big[\sum_{s=0}^{\infty}\frac{slog(s!)\lambda_{ij}^s}{(s!)^v}/Z(\lambda_{ij},v) - \sum_{s=0}^{\infty}\frac{log(s!)\lambda_{ij}^s}{(s!)^v}\sum_{s=0}^{\infty}\frac{s\lambda_{ij}^s}{(s!)^v}/(Z(\lambda_{ij},v))^2\Big],$$

where

$$\mathcal{W}_{ij} = -2(1-p_{ij})/Z(\lambda_{ij},v)^3\sum_{s=0}^{\infty}\frac{\lambda_{ij}^s logs!}{(s!)^v}\sum_{s=0}^{\infty}\frac{s\lambda_{ij}^s}{(s!)^v} - (1-p_{ij})/Z(\lambda_{ij},v)^2\sum_{s=0}^{\infty}\frac{slog(s!)\lambda_{ij}^s}{(s!)^v}.$$

$$\frac{\partial^2\ell}{\partial\gamma_k\partial v} =$$

$$\sum_{i=1}^{N}\sum_{j=1}^{n_i} I(y_{ij}=0)x_{\gamma,ijk}\frac{\Big\{-p_{ij}(1-p_{ij})/Z(\lambda_{ij},v)^2\sum_{s=0}^{\infty}\frac{\lambda_{ij}^s logs!}{(s!)^v}\Big\}\Big\{p_{ij}+(1-p_{ij})/Z(\lambda_{ij},v)+(1-p_{ij})(1-1/Z(\lambda_{ij},v))\Big\}}{\Big\{p_{ij}+(1-p_{ij})/Z(\lambda_{ij},v)\Big\}^2}.$$

Appendix 2.6: A modified(adjusted) sandwich covariance matrix of $\widehat{\beta}$ and $\widehat{\gamma}$ based on the MES algorithm. B_1 and B_2 in the modified sandwich variance matrices are given below. Here, B_2 accounts for variability in the u_i; see, e.g., Satten and Datta (2000).

$$B_1 = E\sum_{i=1}^{N}\begin{pmatrix} -\frac{\partial E(\boldsymbol{y}_i)^T}{\partial\beta}V_{y_i}^{-1}Diag(\mathbf{1}-\boldsymbol{u}_i)\frac{\partial E(\boldsymbol{y}_i)}{\partial\beta} & 0 & -\sum_{j=1}^{n_i}\frac{\partial^2\ell_{ij}^c}{\partial\beta\partial v} \\ 0 & -\frac{\partial \boldsymbol{p}_i(\gamma)^T}{\partial\gamma}V_{u_i}^{-1}\frac{\partial \boldsymbol{p}_i(\gamma)}{\partial\gamma} & 0 \\ -\frac{\partial}{\partial v}(\Psi_2(\beta))^T & 0 & -\sum_{j=1}^{n_i}\frac{\partial^2\ell_{ij}^c}{\partial v^2} \end{pmatrix}.$$

$$B_2 = E\sum_{i=1}^{N}\begin{pmatrix} -\frac{\partial E(\boldsymbol{y}_i)^T}{\partial\beta}V_{y_i}^{-1}Diag(\mathbf{1}-\boldsymbol{u}_i)(\boldsymbol{y}_i-E(\boldsymbol{y}_i(\beta))) \\ -\frac{\partial \boldsymbol{p}_i(\gamma)^T}{\partial\gamma}V_{u_i}^{-1}(\boldsymbol{u}_i-\boldsymbol{p}_i(\gamma)) \\ -(1-\boldsymbol{u}_i)\{-log(\boldsymbol{y}_i!)+\sum_{s=0}^{\infty}\frac{\boldsymbol{\lambda}_i^s logs!}{(s!)^v}/Z(\boldsymbol{\lambda}_i,v)\}^T \end{pmatrix}\widetilde{S}_i(\boldsymbol{u}_i,\boldsymbol{y}_i|\theta=\beta,\gamma,v) -$$

$$\sum_{i=1}^{N}\begin{pmatrix} -\frac{\partial E(\boldsymbol{y}_i)^T}{\partial\beta}V_{y_i}^{-1}Diag(\mathbf{1}-E(\boldsymbol{u}_i))(\boldsymbol{y}_i-E(\boldsymbol{y}_i(\beta))) \\ -\frac{\partial \boldsymbol{p}_i(\gamma)^T}{\partial\gamma}V_{u_i}^{-1}(E(\boldsymbol{u}_i)-\boldsymbol{p}_i(\gamma)) \\ -(1-E(\boldsymbol{u}_i))\{-log(\boldsymbol{y}_i!)+\sum_{s=0}^{\infty}\frac{\boldsymbol{\lambda}_i^s logs!}{(s!)^v}/Z(\boldsymbol{\lambda}_i,v)\}^T \end{pmatrix}\widetilde{S}_i(\boldsymbol{y}_i|\theta=\beta,\gamma,v),$$

where $\widetilde{S}_i(\boldsymbol{u}_i,\boldsymbol{y}_i|\theta=\beta,\gamma,v)=\frac{\partial logf(\boldsymbol{u}_i,\boldsymbol{y}_i|\theta)}{\partial\theta^T}$, $\widetilde{S}_i(\boldsymbol{u}_i|\theta=\beta,\gamma,v)=\frac{\partial logf(\boldsymbol{y}_i|\theta)}{\partial\theta^T}=\int logf(\boldsymbol{u}_i,\boldsymbol{y}_i|\theta^T)dF(u_{i1},...,u_{in_i}|y_{i1},...,y_{in_i})$, and $M_{\text{MES}}=$

$$\sum_{i=1}^{N}\begin{pmatrix} \frac{\partial E(\boldsymbol{y}_i)^T}{\partial\beta}V_{y_i}^{-1}Diag(\mathbf{1}-\boldsymbol{u}_i)(\boldsymbol{y}_i-E(\boldsymbol{y}_i(\beta))) \\ \frac{\partial \boldsymbol{p}_i(\gamma)^T}{\partial\gamma}V_{u_i}^{-1}(\boldsymbol{u}_i-\boldsymbol{p}_i(\gamma)) \\ \sum_{j=1}^{n_i}\frac{\partial\ell_{ij}^c}{\partial v} \end{pmatrix}\Bigg|_{(\widehat{\beta},\widehat{\gamma},\widehat{v})}\begin{pmatrix} \frac{\partial E(\boldsymbol{y}_i)^T}{\partial\beta}V_{y_i}^{-1}Diag(\mathbf{1}-\boldsymbol{u}_i)(\boldsymbol{y}_i-E(\boldsymbol{y}_i(\beta))) \\ \frac{\partial \boldsymbol{p}_i(\gamma)^T}{\partial\gamma}V_{u_i}^{-1}(\boldsymbol{u}_i-\boldsymbol{p}_i(\gamma)) \\ \sum_{j=1}^{n_i}\frac{\partial\ell_{ij}^c}{\partial v} \end{pmatrix}^T\Bigg|_{(\widehat{\beta},\widehat{\gamma},\widehat{v})}.$$

Note that we use the final estimate of the u_{ij} in all of the above.

R-code

```
##############################################################################################
# This r-code is a part of the IFS (Iowa Fluoride Study) simulation section in Chapter 2     #
# This r-code describes MPL/MES estimates and adjusted SE for MPL estimates                  #
# This r-code includes codes for Cardinal Research Clueter (CRC) server lines                #
##############################################################################################

# CRC code
args <- commandArgs(trailingOnly = TRUE)
rho <- as.numeric(args[1])
seq <- as.numeric(args[2])

# Z function as a part of a pmf of a CMP distribution
Z<- function(xmat,b,v, max) {

  lambda <- exp(xmat%*%b)
   # Compute the terms used to sum for the (in)finite summation
      forans <- matrix(0,ncol=max+1,nrow=length(lambda))
   for (j in 1:max){
     temp <- matrix(0,ncol=j,nrow=length(lambda))
     for (i in 1:j){temp[,i] <- lambda/(i^c(v))}
     for (k in 1:length(lambda)){forans[k,j+1] <- prod(temp[k,])}
     }
   forans[,1] <- rep(1,length(lambda))


# Determine the (in)finite sum
   ans <- rowSums(forans)

return(ans)
}

eu_cmp <- function(xmat,zmat,beta,gamma,y,v,max) {
            eu <- matrix(0,nrow(y),1)
            p <- exp(zmat%*%gamma)/(1+exp(zmat%*%gamma))
            exp_u <- p/(p+(1-p)/Z(xmat,beta,v,max))
            for (i in 1:nrow(y)){
                eu[i,] <- ifelse(y[i]==0,exp_u[i],0)
                }
            return(eu) }

  RowbyRow<-function(A, b){  # b is a vector, A is a matrix
              temp <-A
              for (i in 1:length(A[,1]))
                 {temp[i,]<-A[i,]*b[i]}
              return(temp)
              }
 prod.b2 <- function(xmat, b, v,max) {
               l <- exp(xmat%*%b)
             smat <- matrix(0,length(l), max)
             for(i in 1:length(l)) {
                for( j  in 1: max) {
                    smat[i,j] <- l[i]/j^v
                    }}
             temp <- matrix(NA, length(l),max)
```

```
        for (i in 1:length(l)) {
           for (k in 1: max) {
                temp[i,k] <- k^2*prod(smat[i,1:k])
                }
                }
        res <- apply(temp, 1, sum)
        return(res)
          }
prod.b1 <- function(xmat, b, v,max) {
          l <- exp(xmat%*%b)
        smat <- matrix(0,length(l), max)
        for(i in 1:length(l)) {
           for( j  in 1: max) {
               smat[i,j] <- l[i]/j^v
               }}
        temp <- matrix(NA, length(l),max)
        for (i in 1:length(l)) {
           for (k in 1: max) {
                temp[i,k] <- k*prod(smat[i,1:k])
                }
                }
        res <- apply(temp, 1, sum)
        return(res)
          }
 prod.dlv1 <- function(xmat, b,v,max) {
        l <- exp(xmat%*%b)
        smat <- matrix(0,length(l), max)
        for(i in 1:length(l)) {
           for( j  in 1: max) {
               smat[i,j] <- l[i]/j^v
               }}
        temp <- matrix(NA, length(l),max)
        for (i in 1:length(l)) {
           for (k in 1: max) {
                temp[i,k] <- k*(log(factorial(k)))*prod(smat[i,1:k])
                }
                }
        res <- apply(temp, 1, sum)
        return(res)
           }

  prod.v2 <- function(xmat, b,v,max) {
        l <- exp(xmat%*%b)
        smat <- matrix(0,length(l), max)
        for(i in 1:length(l)) {
           for( j  in 1: max) {
               smat[i,j] <- l[i]/j^v
               }}
        temp <- matrix(NA, length(l),max)
        for (i in 1:length(l)) {
           for (k in 1: max) {
                temp[i,k] <- (log(factorial(k)))^2*prod(smat[i,1:k])
                }
                }
        res <- apply(temp, 1, sum)
```

```
            return(res)
               }
   prod.v1 <- function(xmat, b,v,max) {
            l <- exp(xmat%*%b)
            smat <- matrix(0,length(l), max)
            for(i in 1:length(l)) {
               for( j  in 1: max) {
                  smat[i,j] <- l[i]/j^v
                  }}
            temp <- matrix(NA, length(l),max)
            for (i in 1:length(l)) {
               for (k in 1: max) {
                   temp[i,k] <- (log(factorial(k)))*prod(smat[i,1:k])
                   }
                   }
            res <- apply(temp, 1, sum)
            return(res)
               }
# variance of CMP
  var.w <- function(xmat,beta,v,max){

            temp1 <- prod.b2(xmat,beta,v,max)/Z(xmat,beta,v,max)
            temp2 <- prod.b1(xmat,beta,v,max)/Z(xmat,beta,v,max)
            temp <- temp1-temp2^2
            res <- diag(temp)
            return(res)
            }
# mean of CMP
 e.w <- function(xmat,beta,v,max){
           prod.b1(xmat,beta,v,max)/Z(xmat,beta,v,max)
           }
 # To estimate v, use loglikelihood based form
 dlvv<- function(xmat,zmat,b,g,y,v,max) {
              p <- exp(zmat%*%g)/(1+exp(zmat%*%g))
              yidx <- ifelse(y==0,1,0)
              temp1 <- Z(xmat,b,v,max)^(-2)*prod.v1(xmat,b,v,max)*(1-p)
              temp1_1 <- temp1/(p+(1-p)/Z(xmat,b,v,max))*yidx
              temp2 <- -log(factorial(y))+(prod.v1(xmat, b,v,max)/Z(xmat,b,v,max))
              temp2_1 <- (1-yidx)*temp2
              res <- temp1_1+temp2_1
              return(t(res))
              }
var_gamma <- function(zmat,gamma) {
                p <-  exp(zmat%*%gamma)/(1+exp(zmat%*%gamma))
                return(diag(c(RowbyRow(p,1-p))))
                }
  prod.bv <- function(xmat,b,v,max){
            l <- exp(xmat%*%b)
            smat <- matrix(0,length(l), max)
            for(i in 1:length(l)) {
               for( j  in 1: max) {
                  smat[i,j] <- l[i]/j^v
                  }}
            temp <- matrix(NA, length(l),max)
            for (i in 1:length(l)) {
```

```
            for (k in 1: max) {
                temp[i,k] <- k*(log(factorial(k)))*prod(smat[i,1:k])
                }
                }
          res <- apply(temp, 1, sum)
          return(res)
            }
bv_inf <- function(xmat,zmat,b,g,y,v,max){
          uhat <- eu_cmp(xmat,zmat,b,g,y,v,max)
          wbv1 <- prod.bv(xmat, b,v, max)/Z(xmat,b,v,max)
          wbv2 <- (prod.b1(xmat,b,v,max)/Z(xmat,b,v,max))*(prod.v1(xmat, b,v,max)/Z(xmat, b,v,max))
          wbv <- wbv1-wbv2
          res <- t(xmat)%*% ((1-uhat)*wbv)
          return(res)
          }
 dlgamma <- function(xmat,zmat,b,g,y,v,max) {
           p <- exp(zmat%*%g)/(1+exp(zmat%*%g))
           temp1 <- p*(1-p)*(1-1/Z(xmat,b,v,max))/(p+(1-p)/Z(xmat,b,v,max))
           temp1_2 <- RowbyRow(zmat,temp1)
           temp2 <- RowbyRow(zmat,-p)
           yidx <- ifelse(y==0,1,0)
           y0 <- RowbyRow(temp1_2, yidx)
           y1 <- RowbyRow(temp2, (1-yidx))
           res <- y0+y1
           return(t(res))
           }
 dlbeta <- function(xmat,zmat,b,g,y,v,max) {
           p <- exp(zmat%*%g)/(1+exp(zmat%*%g))
           yidx <- ifelse(y==0,1,0)
           temp1 <- y-prod.b1(xmat,b,v,max)/Z(xmat,b,v,max)
           temp1_1 <- RowbyRow(xmat,temp1)
           temp2 <- -(1-p)/Z(xmat,b,v,max)^2*prod.b1(xmat,b,v,max)
           temp2_1 <- RowbyRow(xmat, temp2/(p+(1-p)/Z(xmat,b,v,max)))
           y0 <- RowbyRow(temp2_1,yidx)
           y1 <- RowbyRow(temp1_1, (1-yidx))
           res <- y0+y1
           return(t(res))
           }
 # Adjusted Meat matrix for calculating adjusted sandwich variance for MPL estimates
 M_adj <- function(b,g,v,max) {
         res <- matrix(0,nrow(b)+nrow(g)+1,nrow(b)+nrow(g)+1)
         for ( i in 1:N ) {
            x_ind <- as.matrix(allx[which(allx[,1]==i),-1])
            z_ind <- x_ind
            #z_ind <- matrix(1,nrow(x_ind),1)
            y_ind <- ally[which(ally[,1]==i),-1]

            temp1 <-
rbind(dlbeta(x_ind,z_ind,b,g,y_ind,v,max),dlgamma(x_ind,z_ind,b,g,y_ind,v,max),dlvv(x_ind,z_ind,b,g,y_i
nd,v,max))
            temp2 <- apply(temp1,1,sum)
            res <- res + temp2%*%t(temp2)

                }
         return(res)
```

```
        }

# create a function for correlation coeff. for zero-inf. part
corr_zero1 <- function(eu,p) {
          uist <- NULL
          for(i in 1:(n-1)) {
              for (j in (i+1):n) {
                  uist0  <- (eu[i,]-p[i,])*(eu[j,]-p[j,])/sqrt(p[i,]*(1-p[i,])*p[j,]*(1-p[j,]))
                  uist <- rbind(uist0,uist)
                      }
                  }
          return(sum(uist))
           }
corr_zero2 <- function(eu,p) {
            std_u <- (eu-p)^2/(p*(1-p))
            return(sum(std_u))
            }
# create a function for correlation coeff. for CMP part

 corr_cmp2 <- function(eu,beta0,v0) {
            l <- e.w(xmat,beta0,v0,max)
            res <- (1-eu)^2*(y-l)^2/as.matrix(diag(var.w(xmat,beta0,v0,max)))
            return(sum(res))
            }

gencor.bi<-function(n, mu, delta)
          { X<-rnorm(n)
          R<-matrix(delta, nrow=n, ncol=n)
          diag(R)<-1

          X.cor<-X%*%chol(R)
          x.bi<-qbinom(p=pnorm(X.cor),size=1, prob=mu)
          return(x.bi)
          }
# Peudo loglikelihood function for MPL etimates
logL_cmp <- function(parm) {
        bhat <- as.matrix(parm[1:5])
        ghat <- as.matrix(parm[6:10])
        vhat <- as.matrix(parm[11])
        p <- exp(zmat%*%ghat)/(1+exp(zmat%*%ghat))
        l <- exp(xmat%*%bhat)
        yidx <- ifelse(y==0,1,0)
        l_0 <- log(p+(1-p)/Z(xmat,bhat,vhat,100))
        l_c <- log((1-p)*l^y/((factorial(y)^c(vhat))*Z(xmat,bhat,vhat,100)))
        lsum <- sum(yidx*l_0)+sum((1-yidx)*l_c)
        return(-lsum)
        }
library(MASS)
library(pscl)
library(matrixcalc)

n <- 15
delta <- rho
sigma_rho <- matrix(rho,n,n)+ diag(1-c(rho),n,n)
sigma_del <- matrix(delta,n,n)+ diag(1-c(delta),n,n)
```

```
# calculate the derivative term of rho and delta
  dde1 <- NULL
      for( i in 2:(n-1)) {
           a1 <- matrix(c(NA,rep(1,n-i)),n-i+1,1)
           dde1 <- rbind(dde1, a1)
           }
      dde2 <- matrix(c(NA,rep(1,n-1)),n,1)
      ddel <- rbind(dde2,dde1,NA)

max <- 100
caries <- read.csv("/home/h0choo01/simcari_allzicmp/caries.csv", header=TRUE)
#caries <- read.csv("D:/caries.csv", header=TRUE)
N <- 200
x <- caries[,c(1,16,9:15)]


gamma <- matrix(c(2,0.7,-0.07,0.56,-0.3),5,1) # low proprortion of zero in each subject
beta <- matrix(c(1,0.01,-0.01,-0.13,-0.16),5,1)
v <- 0.6

seedi <- ifelse(rho==0.2,1880,521)
set.seed(seedi+seq)
    Nidx <- sample(unique(x$SID),N)
    x1 <- NULL
    for(i in 1:N){
      x2 <- x[which(x$SID==Nidx[i])[1:n],]
      x1 <- rbind(x1,x2)
     }
     xxx<- apply(x1[,c(4:6,9)],2,function(y) tapply(y,x1[,1], function(x)
unique(x)+round(rnorm(1,0,0.2),2)))
     xx <- matrix(apply(xxx, 2, function(x) rep(x, each=n)),n*N,4)

      onevec <- rep(1,nrow(xx))
      newx <- cbind(onevec,xx)
       allx <- data.frame(sub=as.matrix(rep(seq(1:N),each=n)),int=newx[,1], AUC=newx[,2],
soda=newx[,3],toothbrush=newx[,4],homefluoride=newx[,5])
       allz <- allx
       xmat <- as.matrix(allx[,-1])
       zmat <- xmat

       lambda <- exp(xmat%*%beta)

       p <- exp(zmat%*%gamma)/(1+exp(zmat%*%gamma))
       p_id <- data.frame(id=allz[,1],p=as.matrix(p))

      newy <- NULL
          for ( k in 1:N) {
                  z <- mvrnorm(1,rep(0,n),sigma_rho)
                  u <- pnorm(z)
                  tempx <-  as.matrix(allx[which(allx$sub==k),-1])
                  l <- exp(tempx%*%beta)
                  y1 <- matrix(NA, n,1)
                  for (i in 1:n){
                     y <- 0
                     zinv <- (1/Z(tempx,beta,v,max))[i]
```

```
                    py <- zinv # p(y=0)=Z^-1
                    while( py < u[i]) {
                        y <- y+1
                        py <- py + (l[i]^y/(factorial(y))^v)*zinv
                        }
                    y1[i] <- y
                    }
                  newy <- rbind(newy, y1)
                  }

     u <- NULL
     for (i in 1:N) {
        u <- rbind(u,matrix(gencor.bi(n,p_id[which(p_id$id==i),2],delta),n,1))
             }
      w <- newy
      y <- matrix(NA, n*N,1)
      for (i in 1:(n*N)) {
        y[i,] <- ifelse(u[i]==0, w[i], 0) # zero-inflated data
         }
      ally <- cbind(as.matrix(rep(1:N,each=n)),y)   # index subjects for each data
      all_data <- data.frame(allx,y=ally[,2])
      all_data <- na.omit(all_data)

mZIP <- zeroinfl(formula=y~AUC+soda+toothbrush+homefluoride, dist = "poisson", data=all_data)
beta0 <- beta      # use true parameter values as initial values to obtain MPL/MES estimates
gamma0 <- gamma
v0 <- v
mle_parm <- optim(c(beta0,gamma0,v0), logL_cmp,control=list(maxit=30000), hessian=TRUE)
mpl_bhat <- as.matrix(mle_parm$par[1:5])
mpl_ghat <- as.matrix(mle_parm$par[6:10])
mpl_vhat <- mle_parm$par[11]

adj_sw <-
solve(mle_parm$hessian)%*%M_adj(mpl_bhat,mpl_ghat,mpl_vhat,100)%*%solve(mle_parm$hessian)
adj_swse <- sqrt(diag(adj_sw))

################## MES Algorithm ##########################################################

ddlv <- function(xmat,zmat,b,g,y,v,max){
            uhat <- eu_cmp(xmat,zmat,b,g,y,v,max)
            wv1 <- prod.v2(xmat, b,v,max)/Z(xmat,b,v,max)
            wv2 <- (prod.v1(xmat, b,v,max)/Z(xmat,b,v,max))^2
            temp1 <- wv2-wv1
            res <- temp1*(1-uhat)
            return(sum(res))
            }
  dlvv<- function(xmat,zmat,b,g,y,v,max) {
            p <- exp(zmat%*%g)/(1+exp(zmat%*%g))
            uhat <- eu_cmp(xmat,zmat,b,g,y,v,max)
            temp1 <- -log(factorial(y))+(prod.v1(xmat, b,v,max)/Z(xmat,b,v,max))
            res <- (1-uhat)*temp1
            return(sum(res))
            }
  dpg <- function(zmat, gamma) {      # derivative term of gamma
         temp <- exp(zmat%*%gamma)
```

```
        temp1 <- RowbyRow(zmat,temp)
        temp2 <- (1+temp)^2
        res <- RowbyRow(temp1,1/temp2)
        return(res)
        }
 dlb <- function(xmat,beta,v,max) {          # derivative term of beta
            temp1 <- prod.b2(xmat,beta,v,max)/Z(xmat,beta,v,max)
            temp2 <- prod.b1(xmat,beta,v,max)/Z(xmat,beta,v,max)
            temp3 <- temp1-(temp2)^2
            res <- RowbyRow(xmat, temp3)
            return(res)
            }
 dlv<- function(xmat, beta,v,max) {              # derivative term of v
            temp1 <- as.matrix(prod.dlv1(xmat,beta,v,max)/Z(xmat,beta,v,max))
            temp2 <-
as.matrix(prod.b1(xmat,beta,v,max))*as.matrix(prod.v1(xmat,beta,v,max))/((Z(xmat,beta,v,max))^2)
            res <- temp2-temp1
            return(res)
            }
 r <- 0.4    # tuning parameter for a modified Newton-Rapson algorithm
 delta0 <- 0.5
 rho0 <- 0.5
 all_ddg1 <- matrix(0, 5,5)
 all_geeg1 <- matrix(0,5,1)
 all_ddb1 <- matrix(0,5,5)
 all_geeb1 <- matrix(0,5,1)

# MES algorithm for clustered data

  N1 <- n*(n-1)/2*N
  nt <- n*N
  all_iter <- matrix(NA,13,100)
  for (j in 1: 100 ) {
      all_ddg <- matrix(0,nrow(gamma0),nrow(gamma0))
      all_geeg <- matrix(0,nrow(gamma0),1)
      all_ddb <- matrix(0,nrow(beta0),nrow(beta0))
      all_geeb <- matrix(0,nrow(beta0),1)
      all_del1 <- 0
      all_del2 <- 0
      all_rho1 <- 0
      all_rho2 <- 0
      N2 <- 0
      ntot <- 0
      for (k in 1:N) {
            xmat <- as.matrix(as.data.frame(split(allx,allx[,1])[k])[,-1])
            zmat <- xmat
            y <- as.matrix(matrix(unlist(split(ally,ally[,1])[k]),n,2)[,2])

            p0 <- exp(zmat%*%gamma0)/(1+exp(zmat%*%gamma0))

            eu0 <- eu_cmp(xmat,zmat,beta0,gamma0,y,v0,max)

            ntot1 <- sum((1-eu0)^2)
            ntot <- ntot+ntot1
            N20 <- as.vector(1-eu0)%*%t(as.vector(1-eu0))
```

```
            N21 <- upper.triangle(N20)
            diag(N21) <- 0
            N22 <- sum(N21)
            N2 <- N2 + N22
            ew0 <- e.w(xmat,beta0,v0,max)
            cor_u <- matrix(delta0,n,n)+ diag(1-c(delta0),n,n)
            wu0 <- sqrt(var_gamma(zmat,gamma0))%*%cor_u%*%sqrt(var_gamma(zmat,gamma0))
            cor_y <- matrix(rho0,n,n)+ diag(1-c(rho0),n,n)
            wy0 <- sqrt(var.w(xmat,beta0,v0,max))%*%cor_y%*%sqrt(var.w(xmat,beta0,v0,max))

            # update gamma estimate
            temp_g <- t(dpg(zmat,gamma0))%*%solve(wu0)
            geeg <- as.matrix(apply(RowbyRow(t(temp_g),eu0-p0),2,sum))
           temp_g1 <- t(dpg(zmat,gamma0))%*%solve(wu0)%*%dpg(zmat,gamma0)+ geeg%*%t(geeg)

            all_ddg <- all_ddg + temp_g1 ## sum upto N for derivative of gee for gamma
            all_geeg <- all_geeg + geeg  ## sum upto N for gee for gamma

            # update delta estimate
             all_del1 <- all_del1 + corr_zero1(eu0,p0)
             all_del2 <- all_del2 + corr_zero2(eu0,p0)

             # update beta estimate
            temp_b <- dlb(xmat,beta0,v0,max)
            temp_b1 <- t(temp_b)%*%solve(wy0)%*%diag(c(1-eu0))
            geeb <- as.matrix(apply(RowbyRow(t(temp_b1),(y-e.w(xmat,beta0,v0,max))),2,sum))
            temp_b2 <- t(temp_b)%*%solve(wy0)%*%diag(c(1-eu0))%*%temp_b + geeb%*%t(geeb)

            all_geeb <- all_geeb + geeb
            all_ddb <- all_ddb + temp_b2

             # update rho estimate (cor.coefficient for y)
             vary <- as.matrix(diag(var.w(xmat,beta0,v0,max)))
             temp_rho <- cbind(ddel,vech((1-eu0)%*%t(1-eu0)),vech((y-e.w(xmat,beta0,v0,max))%*%t(y-
                              e.w(xmat,beta0,v0,max))),vech(vary%*%t(vary)) )
             temp_rho <- na.omit(temp_rho)
             temp_rho1 <- temp_rho[,2]*temp_rho[,3]/sqrt(temp_rho[,4])

             all_rho1 <- all_rho1 + sum(temp_rho1)
             all_rho2 <- all_rho2 + corr_cmp2(eu0,beta0,v0)

            }

      all_xmat <- as.matrix(allx[,-1])
      all_zmat <- all_xmat
      all_ys <- as.matrix(ally[,2])
      v1 <- v0 -
r*sum(dlvv(all_xmat,all_zmat,beta0,gamma0,all_ys,v0,max))/(ddlv(all_xmat,all_zmat,beta0,gamma0,all_ys
,v0,max)+ sum(dlvv(all_xmat,all_zmat,beta0,gamma0,all_ys,v0,max))^2)

      ########## updating scheme #############
      if(j==1){ all_geeg1 <- all_geeg} else{ all_geeg1 <- ((j-1)*all_geeg1+all_geeg)/j }
      if(j==1){ all_ddg1 <- all_ddg} else{ all_ddg1 <- ((j-1)*all_ddg1+all_ddg)/j }
      if(j==1){ all_geeb1 <- all_geeb} else{ all_geeb1 <- ((j-1)*all_geeb1+all_geeb)/j }
      if(j==1){ all_ddb1 <- all_ddb} else{ all_ddb1 <- ((j-1)*all_ddb1+all_ddb)/j }
```

```
    gnew0 <- gamma0 + r*solve(all_ddg1)%*%all_geeg1
    gnew <- (j*gamma0+gnew0)/(j+1)
    del_new <- (all_del1/N1)/(all_del2/nt)


    beta11 <- beta0 + r*solve(all_ddb1)%*%all_geeb1
    beta1 <- (j*beta0+beta11)/(j+1)
    rho_new <- (all_rho1/N2)/(all_rho2/ntot)
    vnew <- (j*v0+v1)/(j+1)

    all_iter[,j] <- matrix(c(j,beta1,gnew,vnew, max(abs(gnew-gamma0),abs(beta1-beta0),abs(vnew-
                        v0))),length(c(beta,gamma,v))+2,1)
    if(max(abs(gnew-gamma0),abs(beta1-beta0),abs(vnew-v0)) <  0.01 | j==100) {
                    break
                    }
    gamma0 <- gnew
    beta0 <- beta1
    delta0 <- del_new
    rho0 <- rho_new
    v0 <- vnew
    cat(all_iter[,j],"\n")
              }

file.name <- paste("carisim_CorrectZ_avgiterT","_rho=delta=",rho,"_",seq,".RData", sep="")
save.image(file.name)
q()
```

CHAPTER 3

ANALYZING CLUSTERED COUNT DATA WITH A CLUSTER SPECIFIC RANDOM EFFECT ZERO-INFLATED CONWAY-MAXWELL-POISSON DISTRIBUTION

3.1 Methods and Materials

A zero-inflated model is composed of two parts: a zero-degenerated distribution and a certain count distribution. The zero-degenerated distribution governs excessive zero values with a Bernoulli distribution and the count distribution, a CMP in this case, governs counts including an expected number of zeros. In particular, the probability mass function (pmf) of a zero-inflated Conway-Maxwell-Poisson (ZICMP) distribution is given by

$$P(Y=y)=\begin{cases} p+\frac{(1-p)}{Z(\lambda,v)}, & \text{if } y=0, \\ (1-p)\frac{\lambda^y}{(y!)^v Z(\lambda,v)}, & \text{if } y\geq 1, \end{cases} \tag{1}$$

where λ is a positive rate parameter, v is a positive dispersion parameter, and $Z(\lambda,v)=\sum_{s=0}^{\infty}\lambda^s/(s!)^v$ in a normalizing constant in a regular CMP distribution. It turns out that for a CMP distribution (Shmueli et al., 2005), v between 0 and 1 indicates overdispersion of data (i.e., variance is greater than the mean) and $v > 1$ indicates underdispersion of data. Furthermore, the ZICMP distribution reduces to a zero-inflated Poisson (ZIP) distribution when $v=1$. In (1), p is the probability that a sample value of the response variable, y, is from a degenerated zero distribution.

In this chapter, we let both the excessive zero probability p and the rate parameter λ depend on covariates through appropriate link function. The shape parameter v is held as unknown constant that needs to be estimated from the data as well. We build up correlation in the data by introducing a cluster level random effects in the count data part which is the main component of the distribution for most applications. We feel this modeling offers enough richness for applications while keeping the likelihood computationally

manageable. In order to keep track of clustered data, we use two different indices, i as a cluster indicator and $j = j(i)$ as an observation indicator within the cluster i. The link functions of both parts for a ZICMP joint model is now described as below

$$log\lambda_{ij} = X_{\beta,ij}^T\beta + \sigma_b b_i,$$

$$log\frac{p_{ij}}{1-p_{ij}} = X_{\gamma,ij}^T\gamma, \tag{2}$$

where $X_{\beta,ij}^T \in \Re^{1\times p_\beta}$ and $X_{\gamma,ij}^T \in \Re^{1\times p_\gamma}$ are fixed effect covariates corresponding to the j^{th} observation in the i^{th} cluster for the count and zero part, respectively. β and γ are fixed effect parameters for their corresponding part. b_i is a random intercept of the i^{th} cluster and is normally distributed with mean 0 and standard deviation σ_b. Therefore, a set of parameters to be estimated in a ZICMP joint model is, then, $\theta = (\beta, \gamma, \upsilon, \sigma_b)$.

3.1.1 Calculation of the approximate likelihood function and estimation of parameters

The likelihood contribution of the i^{th}cluster in a mixed effect ZICMP model is given by

$$L_i = \int_{-\infty}^{\infty}\left[\prod_{j=1}^{n_i}\left\{I(Y_{ij}=0)\left(p_{ij}(\gamma) + \frac{1-p_{ij}(\gamma)}{Z(\lambda_{ij}(\beta,\sigma_b,b_i),\upsilon)}\right) + I(Y_{ij}\geq 1)\Big((1-p_{ij}(\gamma))\right.\right.$$

$$\left.\left.\times \frac{\lambda_{ij}(\beta,\sigma_b,b_i)^{y_{ij}}}{(y_{ij}!)^\upsilon Z(\lambda_{ij}(\beta,\sigma_b,b_i),\upsilon)}\Big)\right\}\right] p(b_i)\, db_i. \tag{3}$$

Note that L_i is calculated not only from multiplying all within-subject (cluster) observations but also by integrating out the subject (cluster) specific random effect. As a result, the observed likelihood function of a cluster in Equation (3) is difficult to compute in a theoretical fashion since there is no closed form of L_i to integrate out with respect to b_i. Thus, we employ a Gaussian-Hermite (G-H) quadrature method to approximate L_i, which is a popular numerical approximation to an integral given by

$$\int_{-\infty}^{\infty} f(x)e^{-x^2}dx \approx \sum_{q=1}^{Q} f(x_q)*w_q, \tag{4}$$

where x_q is a quadrature grid point and w_q is the corresponding weight. We use an R package called 'fastGHQuad ' to obtain w_q and x_q. For a direct application of (4) toward calculating L_i, we let $b_i \sim N(0, 0.5)$ which has $\frac{1}{\sqrt{\pi}}e^{-b_i^2}$ as its pdf. In addition, the log link function is reparametrized as

$log(\lambda) = X_\beta\beta + \sigma_b\sqrt{2}\,b_i$ which allows us to estimate variance of the random effect term from the count part link function.

Applying this G-H quadrature method in equation (4), an approximation of L_i is given by

$$\widetilde{L}_i := \frac{1}{\sqrt{\pi}}\sum_{q=1}^{Q} w_q \times \left[\prod_{j=1}^{n_i}\left\{ I(Y_{ij}=0)\Big(p_{ij}(\gamma) + \frac{1-p_{ij}(\gamma)}{Z(\lambda_{ij}(\beta,\sigma_b,b_{iq}),v)}\Big) + \right.\right.$$

$$\left.\left. I(Y_{ij}\geq 1)\Big((1-p_{ij}(\gamma)) \times \frac{\lambda_{ij}(\beta,\sigma_b,b_{iq})^{y_{ij}}}{(y_{ij}!)^v \times Z(\lambda_{ij}(\beta,\sigma_b,b_{iq}),v)}\Big)\right\}\right], \quad (5)$$

and consequently, an approximate observed log-likelihood function of a mixed effects ZICMP model is obtained as $\log\widetilde{L} = \sum_{i=1}^{N} \log(\widetilde{L}_i)$. Parameter estimates are obtained by maximizing the approximate log-likelihood function. We have used the 'optim' function in R (version 3.1.2) to this end throughout this chapter.

3.1.2 Variance Estimation

Variance estimation is an integral part of statistical inference based on the approximate MLE obtained above. We have considered both an approximate inverse Fisher information matrix and a sandwich variance estimate. The former is given by

$$\widehat{\text{A-Var}}(\widehat{\theta}) = = \left(-\sum_{i=1}^{N}\widetilde{H}_i(\widehat{\theta})\right)^{-1},$$

where $\widetilde{H}_i(\widehat{\theta}) = \frac{\partial^2 log\widetilde{L}_i(\widehat{\theta})}{\partial\widehat{\theta}\partial\widehat{\theta}^T}$ is an approximate Hessian matrix contribution of the i^{th}cluster; it can be directly outputted from the '*optim*' function in R.

The estimated sandwich variance-covariance matrix is calculated using the G-H quadrature approximation is given by

$$\widehat{\text{A-Var}}_S(\widehat{\theta}) = \widehat{B}^{-1}\widehat{M}\widehat{B}^{-1},$$

where $\widehat{B} = \sum_{i=1}^{N}\widetilde{H}_i(\widehat{\theta})$ and $\widehat{M} = \sum_{i=1}^{N}\widetilde{S}_{\,i}(\widehat{\theta})\widetilde{S}_{\,i}(\widehat{\theta})^T$. Here S_i is obtained as

$$S_i(\,\theta) = E_{b_i}\Big[S_i(\,\theta; y_i, b_i)|y_i\Big] = \frac{\int S_i(\,\theta; y_i, b_i)\psi_i(\theta)\, f_{b_i}(b_i)db_i}{\int \psi_i(\theta)\, f_{b_i}(b_i)db_i}$$

where $\psi_i(\theta) = \prod_{j=1}^{n_i}\Big\{ I(Y_{ij}=0)\Big(p_{ij}(\gamma) + \frac{1-p_{ij}(\gamma)}{Z(\lambda_{ij}(\beta,\sigma_b,b_i),v)}\Big) + I(Y_{ij}\geq 1)\times$

$$(1-p_{ij}(\gamma))\frac{\lambda_{ij}(\beta,\sigma_b,b_i)^{y_{ij}}}{(y_{ij}!)^v \times Z(\lambda_{ij}(\beta,\sigma_b,b_i),v)}\Big\},$$

and $f_{b_i}(b_i) = \frac{1}{\sqrt{\pi}}\, e^{-b_i^2}$ and $\widetilde{S}_{\,i}$ is its approximation obtained using $\frac{\partial \widetilde{L}_i(\,\theta)}{\partial\theta}\Big/\widetilde{L}_i(\,\theta)$. Note that

$$\frac{\partial L_i(\theta)}{\partial\theta} = \frac{\partial}{\partial\theta}\frac{1}{\sqrt{\pi}}\int\prod_{j=1}^{n_i}\Big\{ I(Y_{ij}=0)\Big(p_{ij}(\gamma) + \frac{1-p_{ij}(\gamma)}{Z(\lambda_{ij}(\beta,\sigma_b,b_i),v)}\Big) +$$

$$I(Y_{ij}\geq 1)(1-p_{ij}(\gamma))\frac{\lambda_{ij}(\beta,\sigma_b,b_i)^{y_{ij}}}{(y_{ij}!)^v \times Z(\lambda_{ij}(\beta,\sigma_b,b_i),v)}\Big\} \times e^{-b_i^2} db_i. \tag{6}$$

$$= \frac{1}{\sqrt{\pi}}\int \frac{\partial}{\partial\theta}\,\psi_i(\theta)\, e^{-b_i^2} db_i$$

$$= \frac{1}{\sqrt{\pi}}\int \frac{\partial}{\partial\theta} log\psi_i(\theta) \times \psi_i(\theta)\, e^{-b_i^2} db_i$$

$$\approx \frac{1}{\sqrt{\pi}}\sum_{q=1}^{Q}\frac{\partial}{\partial\theta}\, log\psi_i(\theta; b_{iq}) \times \psi_i(\theta; b_{iq}). \tag{7}$$

Thus, $\frac{\partial \widetilde{L}_i(\theta)}{\partial\theta}\Big|_{\widehat{\theta}}$ is finally obtained from Equation (7) (see Appendix 3.1 for the details).

In general, it is preferable to use a sandwich variance since it is robust to model misspecification.

3.1.3 A Statistical Test for Zero-Inflation

Besides testing for a significant regression effect, we may be interested in testing whether there exists zero-inflation in the data. In other words, to determine whether a ZICMP model is statistically necessary over a simpler CMP model under the same mixed effects framework for a given dataset. The parameter associated with zero-inflation is the marginal probability p of excessive zero part. Indeed, it is a testable set of hypotheses given by $H_0 : p = 0$ (no zero inflation) versus $H_1 : p > 0$ (zero-inflation) under the setting where $p \in [0,1]$ is not covariate dependent.

Note that testing for zero-inflation is different from testing for zero-modification. In fact, likelihood based score tests are applicable and widely used for testing zero-modification but not for zero-inflation. A zero-modified model allows a range of negative values of p so that the value of the null

parameter being on the boundary for testing for zero inflation is avoided (Jansakul and Hinde, 2002). In the zero-inflation test, one of the proper methods for handling the boundary issue is to use likelihood ratio test (LRT) statistics, which asymptotically follows an equal mixture of point mass at 0 and a chi-squared distribution with one degree of freedom (Fulton et. al., 2015). That is, $\Lambda = 2\log(L_{\text{ZICMP}}/L_{\text{CMP}}) \xrightarrow{d} 0.5 \times \delta_0 + 0.5 \times \chi^2_1$. However, due to numerical issues in computing the likelihood functions and different fitting procedures of ZICMP and CMP models, one may run into numerical issues in applying the LRT in practice when using the large sample approximation for calibrating the rejection region. Instead, we employ a null bootstrap mechanism to get approximate p-values for the zero-inflation test using the LRT.

The null bootstrap method starts by creating bootstrap (re)samples from the null distribution which asserts the presents of no zero inflation in the data. Thus, while we estimate the model parameters using the original sample, we only resample from the corresponding estimated CMP part obtained by setting p to zero and using the sample estimates for the remaining parameters. For each null bootstrap sample, we can calculate the LRT $\Lambda_{\text{b}} = 2\log(L_{\text{b, ZICMP}}/L_{\text{b, CMP}})$ as above using the two maximized log-likelihoods one with ZICMP and the other using CMP, where $\text{b} = 1, \ldots, B$, where B is a large positive integer, called the bootstrap replication size. Now a bootstrap based p-value is calculated by $\text{p}^* = \sum_{\text{b=1}}^{B} I(\Lambda_{\text{bs}} > \Lambda_{obs})/B$, where Λ_{obs} is the value of the LRT for the original sample. One would reject the null hypothesis if $\text{p}^* < \alpha$, α being the nominal level.

3.2 Simulation Studies

This simulation section contains three types of investigations into the finite sample behaviors of our inferential methodology. First, we study the behaviors of the point estimators in terms of their bias and variance. We also study the performance of approximating normal distribution based confidence intervals for the parameter of interest using two types of variance estimates. Next we conduct a power analysis for a statistical test for a regression effect based on an approximate Wald test; one again both variance estimates have been attempted to standardize the test statistic. The third subsection contains the simulation results for the zero-inflation test. We study the size/power of the test for three choices of the dispersion parameter corresponding to three types of dispersion patterns.

3.2.1 Study of bias and variance

We considered three different combinations of number of clusters with the same cluster size, namely, $N = 30$, $N = 50$, and $N = 75$ each with $n_i = 10$. Both X_β and X_γ are set as the same design matrix with an intercept term and one continuous covariate randomly generated from a normal distribution with mean 0.94 and variance 0.25, which were arbitrarily chosen. For all three scenarios, we considered the same set of parameter values given by $\beta_0 = 13.8,\ \beta_1 = -1.3, \gamma_0 = 2,\ \gamma_1 = -3,\ v = 6,$ and $\sigma_b^2 = 4$; these values were motivated by an application of CMP regression to airfreight breakage (Kutner, Nachtsheim and Neter, 2003, page 35, Exercise 1.21) by Sellers and Shmueli (2010) although their modeling was were less complex than ours. Given the true parameter values under each scenario, count values including excessive zeros were created. The count values of the i^{th} cluster from the zero part were randomly sampled from a Bernoulli distribution with success probability, $p_i = exp(X_{i,\gamma}\gamma)/(1 + exp(X_{i,\gamma}\gamma))$, and the values from the count part were generated by using the inverse CDF transformation method from a CMP distribution with $\lambda_i = exp(X_{i,\beta}\beta + \sigma_b\sqrt{2}b_i)$ and v. Here b_i was randomly generated from $N(0, 0.5)$ implying that the true variance of the within cluster random effect was 4 in this simulation.

We have used a Monte-Carlo sample size of $M = 500$ to empirically estimate the bias and the variance of our estimators in each setting. For a single data set, the Gaussian-Hermite (G-H) quadrature method (see Section 3.1 for details) was used to approximate L_i via $\widetilde{L}_i$. One needs to make a decision about Q, the number of grid points in order to apply Equation 5; for a given Q, both b_{iq} and w_q can be obtained from a R package fastGHQuad. Currently, there is no standard statistical criteria to determine an optimal value of Q and a large value of Q may add to the computational burden of the entire process. Following Lesaffre and Spiessens (2001), we have used $Q = 25$ in all our subsequent calculations. The results are given in Table 3.1, where we report the bias, the true standard error (as obtained form Monte Carlo), and the averages of estimated standard error, using both the information matrix and the sandwich estimator, over all 500 Monte Carlo runs.

As the number of clusters increases, the approximation bias due to the use of a quadrature method seems to dominate the statistical bias since the empirically computed total bias does not converge to zero

but rather stay flat. Generally speaking, the empirically estimated true standard deviation (SE) values reduce with sample size. The estimated standard errors for the regression parameters from both methods were reasonably close to the true standard deviations for all scenarios as can be seen from their ratios. However, the sandwich variance method is tends to be more conservative than the Fisher information variance method throughout.

Furthermore, we have created probability-probability (p-p) plots to explore the behaviors of the inference based on the asymptotic normality and the two estimates of asymptotic variance in a more precise way. A p-p plot consists of a range of nominal coverage rate along the x-axis and the corresponding empirical coverage rate along the y-axis and it is a useful tool for checking if a large sample normal approximation of a parameter estimator is effective. The p-p plots from the case of $N = 50$ are shown in Figure 3.1. For the Fisher information method (the left panel), the p-p plots for most of all the parameters are close to the reference solid line except for σ_b. On the other hand, the p-p plot of σ_b with the sandwich variance method performs noticeably better (the right panel in Figure 3.1). In addition, the p-p plots corresponding to the remaining parameters behave well in terms of all the points being close to the reference lines. Likewise, for both $N = 30$ and 75 cases, all the p-p plots from the sandwich variance method include points closer to the reference lines than those from the Fisher information method, especially, for σ_b (Figures 3.3 and 3.4).

3.2.2 Power analysis

We explore the behaviors of statistical powers of the Wald test for $H_0 : \beta_1 = 0$, using each of the two variance estimation methods, in all three dispersion cases. Figure 3.2 describes the power curves corresponding to the two different variance estimation methods. When the true value of β_1 is 0, the empirical size for the sandwich variance method ("sandwich variance" in Figure 3.2) is 0.054 which is extremely close to the targeted nominal size $\alpha = 0.05$. However, the Fisher information method ("Fisher information" in Figure 3.2) produces a more inflated size of 0.082. Thus, we can conclude that it is more appropriate to use the sandwich variance method than the inverse of the Fisher information for the approximate maximum likelihood inference in the mixed ZICMP regression.

3.2.3 A power analysis for the zero inflation test

Next, we present the power analysis for the zero-inflation test in three different choices of the dispersion parameters corresponding to underdispersion, overdispersion and equidispersion for the count part of the model. All the performance of the power analysis is based on $N = 30$ with the same design matrix used for the previous simulation study. Since these analyses are computationally expensive, only two different alternative values of p_s ($= 0.03$ and 0.1) are considered for the power of a test along with the null value of p_s (= 0) for each the three dispersion choices.

In this analysis, we use different sets of regression parameters for those three dispersion cases. For the underdispersion case, the parameter values are $\{\beta = (5, -2), v = 4, \sigma_b = 2\}$. For the overdispersion and equidispersion cases, the values are $\{\beta = (-1.5, 0.5), v = 0.6, \sigma_b = 1\}$ and $\{\beta = (0.7, -2), v = 1, \sigma_b = 1\}$, respectively. In each case, the parameter values are chosen to maintain a stable range of observations across 500 generated datasets.

Table 3.4 shows how the power of the ZI test behaves as p increases from its null value of 0 in each of the three different dispersion cases. The empirical size of a test seems to attain the nominal size, $\alpha = 0.05$ for both equidispersion and overdispersion cases. However, the underdispersion case shows the empirical size marginally inflated. In addition, compared to p_s, power values increase more dramatically in the underdispersion case than the other two.

3.3 Applications

This section provides an example to demonstrate the usefulness of a CMP distribution in terms of handling a wide range of dispersion. We use the maize hybrids experiment (Paschold et. al, 2014) recording the expression values of genes in terms of the read counts using a next generation sequencing (NGS) platform. This dataset consists of 39,656 genes with four different genotypes of corns (B73, B73 $\times$ Mo17, Mo17 $\times$ B73 and Mo17). Four experimental units are assigned to the same genotype (in this case, total sixteen corns are used) with four different tissues for each unit. Therefore, there are total 64 observations for each gene ID. Since these tissues are four different parts of the same root, the tissues within the same root are possibly correlated leading to clustered count data.

3.3.1 An underdispersion case

Out of 39,656 genes, the gene ID, "GRMZM2G042361" is selected to represent the case of $v > 1$. This gene ID consists of 64 observations including 37 zeros, 23 ones, 3 twos and 1 three. A joint model is applied to analyze the data with a ZICMP framework including an random intercept accounting for correlations within each root. Since the total numbers of read counts over genes are different among biological units (in this case, individual replicate for each root), the model needs to be adjusted for normalizing the data by adding an offset term into the link function of the count part. The offset term is calculated by summing all read counts for each biological unit. Thus, our adjusted count part link function becomes $log(\lambda_{ij}) = X_{ij}^T\beta + \sigma_b\sqrt{2}b_i + log(\text{offset}_{ij})$, where $i = 1, \ldots, 16$, $j = 1, \ldots, 4$. The fixed effect terms on both the count part and zero part are set to be the same as four different genotypes.

A naive bootstrap confidence interval (typically, 95%) is used since a normal-based confidence interval may not be appropriate due to such a small number (four) of clusters. A naive bootstrap confidence interval is obtained by starting with resampling with replacement by cluster level, not by observation level in order to preserve the original correlations within the same roots. For more variability, we randomly re-assign four different genotypes to the sixteen bootstrap sample roots. Based on a bootstrap dataset corresponding to the re-assigned genotypes, one set of bootstrap-based-estimates is obtained after fitting into a joint model based on a ZICMP framework. After iterating this procedure, BS bootstrap datasets are obtained along with BS sets of bootstrap-based-estimates. Finally, the cluster-based naive bootstrap confidence interval is calculated based on those estimates (in this case, BS=500).

The result of a ZICMP joint model analysis on this gene is provided in Table 3.2. Note that the point estimate of the dispersion parameter, v, is larger than one and the 95% bootstrap confidence interval of v excludes one. The standard error of the random intercept as the random component is extremely small. Thus, we refit the same data using an independent ZICMP model for comparison as shown in the table on the right side in Table 3.2. In other words, this independent ZICMP model does not include the random intercept as a part of the count link function and consider that all the observations are independent, not clustered. As the result from the independent case, this data is also underdispersed since $\hat{v}$ is greater than one with the confidence interval that excludes one and is the same value with the one from the joint ZICMP

model rounding to four decimal places. In fact, most of all the coefficients from both count and zero parts are relatively similar to these two models. The effects of this gene do not appear to be statistically significant.

3.3.2 An over-dispersion case

We select the gene ID, "GRMZM2G106026" to represent the overdispersion case. The settings are exactly the same as the underdispersion case in Section 3.4.1 except for the different set of observations due to using a different gene ID. Table 3.3 summarizes the results including all the estimates based on a mixed ZICMP model with the 95% naive bootstrap confidence intervals. According to the results, it is clear that $\hat{v}$ is less than 1 which indicates that this dataset is evidently overdispersed and the standard error of the random intercept, $\hat{\sigma}_b$ is relatively large. Based on the confidence intervals, the genetic effects are not statistically significant.

3.4 Disscussion

We have considered analyzing clustered count data based on a mixed effects ZICMP model. However, it is numerically cumbersome to calculate the likelihood function based on a ZICMP mixed effects model due to the absence of an explicit closed form of the likelihood function. While using the G-H quadrature method for the likelihood approximation, we notice the importance of choosing an optimal value for the total number of quadrature points, Q described in Section 3.1 and simulations in Section 3.2. Through the simulations, we notice that the approximate likelihood results in biased estimation, while the standard errors get smaller with the number of clusters, as expected. The overall inference from this methodology, specially using the sandwich variance calculation is still adequate for the regression parameters as shown by the p-p plots and the power analysis. The likelihood framework is also suitable for developing a test for zero inflation. However, a bootstrap calibration is recommended instead of the asymptotic distribution to overcome the problem of numerical approximation of the likelihood function.

As alternative way to analyze clustered count data in the ZICMP framework is to perform a marginal regression analysis (Choo-Wosoba, Levy and Datta, 2016). Unlike our mixed effects model in this

chapter, the marginal model does not specify any random effects because the primary concern of marginal modeling is to get marginal inference over a target population. On the other hand, the mixed effects model focuses on individual's inference in estimating both fixed effects, β, and the random effect, σ_b. Hence, these two models for zero-inflated clustered data do not support the same statistical inference. In fact, it is not feasible to compare these two sets of estimators statistically.

A limitation of the mixed effects approach is the difficulty of incorporating a larger number of random effects term since the approximation by G-H quadrature to the likelihood function will be even more problematic. A Bayesian approach may be a way around in such situations. We explore this possibility in the next chapter.

TABLE 3.1

Finite sample behavior of parameter estimators obtained from fitting ZICMP mixed effects model when the likelihood is approximated by G-H quadrature with 25 grid points: Here SE_{Fisher} indicates estimated standard errors based on the inverse of the Fisher information matrix and SE_{sand} refers to estimated standard errors from a sandwich covariance variance matrix. Also, SE stands for the standard error based on the Monte Carlo replicates and is taken to be the gold standard for comparison.

		$N = 30$						$N = 50$					
	True	bias	SE	SE_{Fisher}	SE_{sand}	SE/SE_{Fisher}	SE/SE_{sand}	bias	SE	SE_{Fisher}	SE_{sand}	SE/SE_{Fisher}	SE/SE_{sand}
β_0	13.8	-0.296	1.904	1.500	2.125	1.270	0.896	-0.474	1.531	1.153	1.676	1.327	0.913
β_1	-1.3	0.014	0.230	0.207	0.272	1.113	0.848	0.037	0.193	0.158	0.209	1.216	0.921
γ_0	2	0.081	0.369	0.355	0.349	1.040	1.058	0.003	0.270	0.269	0.265	1.003	1.019
γ_1	-3	-0.099	0.432	0.400	0.396	1.080	1.091	-0.002	0.304	0.302	0.297	1.005	1.022
v	6	-0.060	0.743	0.649	0.935	1.146	0.795	-0.159	0.624	0.495	0.705	1.259	0.885
σ_b	2	-0.134	0.340	0.227	0.450	1.496	0.757	-0.183	0.276	0.172	0.351	1.601	0.786
		$N = 75$											
	True	bias	SE	SE_{Fisher}	SE_{sand}	SE/SE_{Fisher}	SE/SE_{sand}						
β_0	13.8	-0.537	1.302	0.936	1.324	1.392	0.984						
β_0	-1.3	0.046	0.153	0.129	0.167	1.190	0.915						
γ_0	2	0.048	0.267	0.222	0.222	1.202	1.202						
γ_0	-3	-0.053	0.304	0.250	0.250	1.217	1.218						
v	6	-0.196	0.531	0.401	0.561	1.323	0.947						
σ_b	2	-0.217	0.223	0.140	0.311	1.592	0.716						

TABLE 3.2

Results for the GRMZM2G042361 gene from the maize hybrids data, for which the estimated dispersion parameter was larger than 1 representing underdispersion in the fixed parameter count data part: Parameter estimates are reported along with a cluster bootstrap based(nonasymptotic) naive confidence intervals. Here, the bootstrap replication size was 500.

Mixed model					Independent			
	Count part	BS_CI	Zero part	CI (Bootstrap)	Count part	BS_CI	Zero part	CI (Bootstrap)
Intercept	-16.653	(-16.956,-13.882)	-12.550	(-15.272, 0.236)	-16.653	(-16.872, -12.376)	-12.550	(-15.058, 0.910)
B73 × Mo17	0.525	(-1.960, 2.040)	-1.933	(-9.969, 10.698)	0.525	(-3.064, 3.164)	-1.933	(-11.910, 12.415)
Mo17	1.498	(-1.860, 2.130)	-2.573	(-11.030, 10.302)	1.499	(-2.805, 3.446)	-2.573	(-11.946, 12,216)
Mo17 × B73	0.631	(-2.285, 2.081)	-1.247	(-10.427, 10.696)	0.631	(-2.964, 3.327)	-1.247	(-11.868, 12.077)
v	2.100	(1.470, 4.569)			2.100	(1.556, 40.650)		
σ_b	0.001	(0.001,0.951)			N/A			

TABLE 3.3

Results for the GRMZM2G106026 gene from the maize hybrids data, for which the estimated dispersion parameter was smaller than 1 representing overdispersion in the fixed parameter count data part: Parameter estimates are reported along with a cluster bootstrap based (nonasymptotic) naive confidence intervals. Here, the bootstrap replication size was 500.

Mixed model				
	Count part	BS_CI	Zero part	BS_CI
Intercept	-15.981	(-16.116, -14.736)	-0.296	(-0.717, 0.793)
B73 $\times$ *Mo17*	0.523	(-0.813, 0.734)	0.018	(-1.220, 1.197)
Mo17	0.460	(-0.768, 0.811)	0.499	(-1.153, 1.086)
Mo17 $\times$ *B73*	0.549	(-0.773, 0.846)	0.777	(-1.103, 1.098)
v	0.154	(0.095, 0.601)		
σ_b	0.141	(0.001, 0.539)		

TABLE 3.4

Power of a zero inflation test (nominal size is 5%) based on a ZICMP model for 30 clusters: Three different choices of the dispersion parameter v was used. Each entry is based on a Monte Carlo sample size of 500.

		Choice of v	
Proportion of zero inflation	Underdispersion	Equidispersion	Overdispersion
0	0.092	0.042	0.040
0.03	0.872	0.216	0.120
0.1	1.000	0.476	0.280

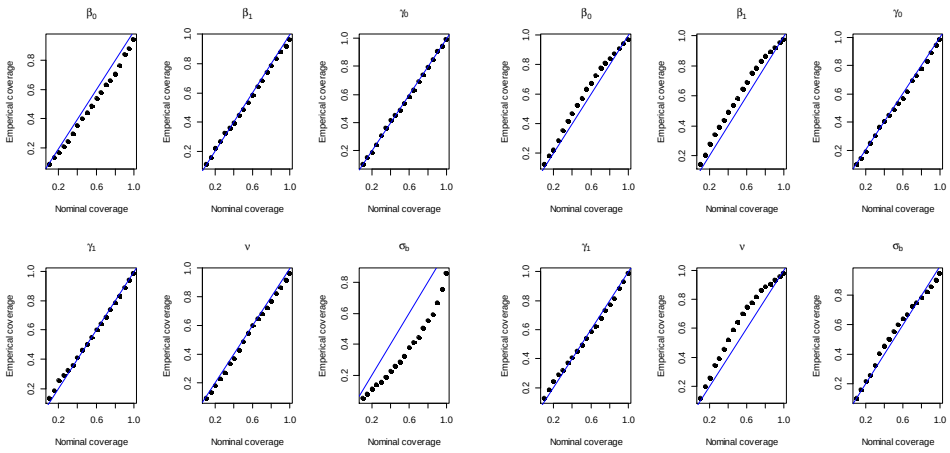


FIGURE 3.1. The p-p plot of confidence intervals of all parameters based on our simulation models when the number of clusters is 30. The left panel is based on the inverse of the Fisher information matrix and the right panel is based on the sandwich variance estimator.

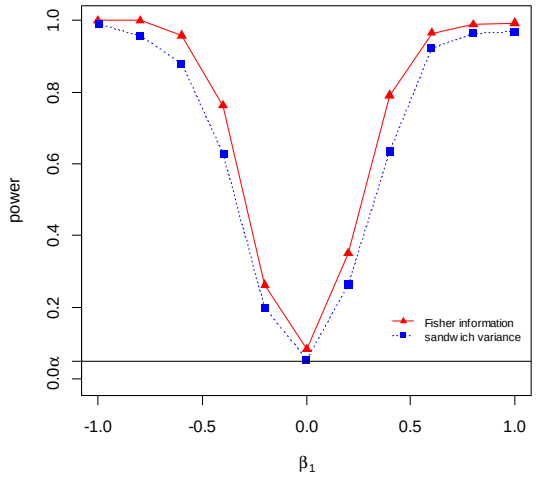


FIGURE 3.2. The power plots for testing the effect of X_1 using two types of asymptotic variance estimation methods. The red curve corresponds to using the inverse of the Fisher information and the blue curve corresponds to using a sandwich variance estimator. The horizontal black line denotes the nominal size.

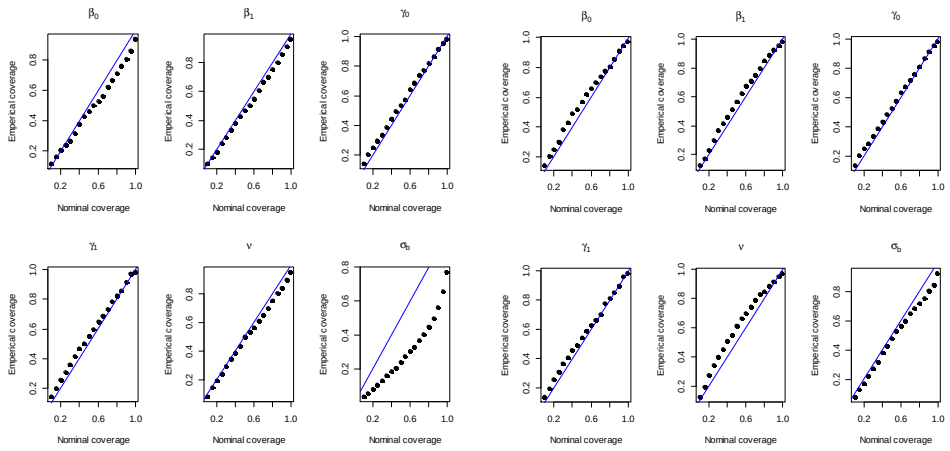


FIGURE 3.3. The p-p plot of confidence intervals of all parameters based on our simulation models when thenumber of clusters is 50. The left panel is based on the inverse of the Fisher information matrix and the right panel is based on the sandwich variance estimator.

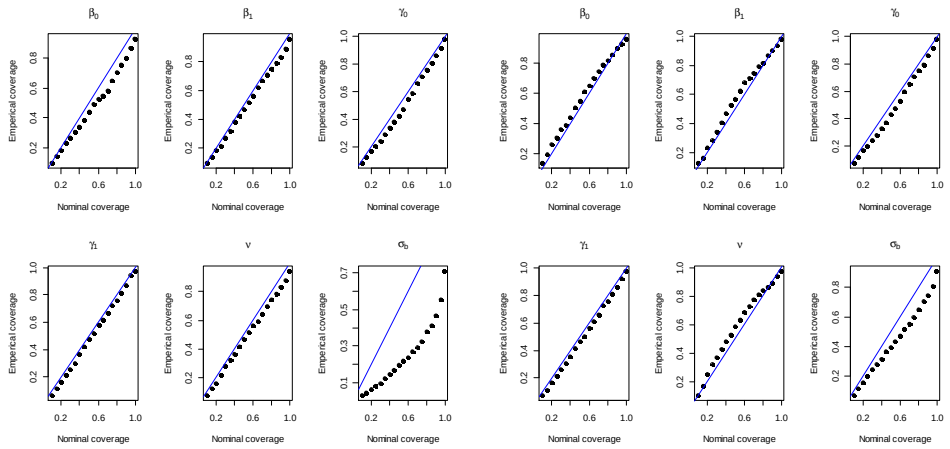


FIGURE 3.4. The p-p plot of confidence intervals of all parameters based on our simulation models when the number of clusters is 75. The left panel is based on the inverse of the Fisher information matrix and the right panel is based on the sandwich variance estimator.

Appendix 3.1: Score function of the mixed effect (joint) model based on a ZICMP framework for θ: β, γ, υ and σ_b

$$
\begin{aligned}
1.\ \frac{\partial L_i}{\partial \beta} &= \frac{1}{\sqrt{\pi}} \int \Bigg\{ \frac{\partial}{\partial \beta} \sum_{j=1}^{n_i} I(Y_{ij}=0) log\Big(p_{ij}(\gamma) + \frac{1-p_{ij}(\gamma)}{Z(\lambda_{ij}(\beta,\sigma_b,b_i),v)} \Big) \ + I(Y_{ij} \geq 1) \\
&\quad \Big(log(1-p_{ij}(\gamma)) + y_{ij} log\lambda_{ij}(\beta,\sigma_b,b_i) - logZ(\lambda_{ij}(\beta,\sigma_b,b_i),v) \\
&\quad - vlog(y_{ij}!) \Big) \Bigg\} \psi_i(\theta) e^{-b_i^2} db_i \\
&\approx \frac{1}{\sqrt{\pi}} \sum_{q=1}^{Q} \Bigg[w_q \Bigg\{ \sum_{j=1}^{n_i} I(Y_{ij}=0) \frac{(1-p_{ij}(\gamma)) \sum_{s=0}^{\infty} \frac{s\lambda_{ij}(\beta,\sigma_b,b_{iq})^s}{(s!)^v} X_{ij}}{-\Big(p_{ij}(\gamma) + \frac{1-p_{ij}(\gamma)}{Z(\lambda_{ij}(\beta,\sigma_b,b_{iq}),v)}\Big) Z(\lambda_{ij}(\beta,\sigma_b,b_{iq}),v)^2} + I(Y_{ij} \geq 1) \\
&\quad \Big(y_{ij} - \frac{\sum_{s=0}^{\infty} \frac{s\lambda_{ij}(\beta,\sigma_b,b_{iq})^s}{(s!)^v}}{Z(\lambda_{ij}(\beta,\sigma_b,b_{iq}),v)} \Big) X_{ij} \Bigg\} \psi_i(\theta; b_{iq}) \Bigg]
\end{aligned}
$$

$$
\begin{aligned}
2.\ \frac{\partial L_i}{\partial \gamma} &= \frac{1}{\sqrt{\pi}} \int \Bigg\{ \sum_{j=1}^{n_i} I(Y_{ij}=0) \Big(\frac{p_{ij}(\gamma)(1-p_{ij}(\gamma))(1-Z^{-1}(\lambda_{ij}(\beta,\sigma_b,b_i),v)))}{p_{ij}(\gamma) + \frac{1-p_{ij}(\gamma)}{Z(\lambda_{ij}(\beta,\sigma_b,b_i),v)}} Z_{ij} \Big) \ + I(Y_{ij} \geq 1) \times \\
&\quad - \frac{p_{ij}(\gamma)(1-p_{ij}(\gamma))}{1-p_{ij}(\gamma)} Z_{ij} \Bigg\} \psi_i(\theta) e^{-b_i^2} db_i \\
&\approx \frac{1}{\sqrt{\pi}} \sum_{q=1}^{Q} w_q \Bigg[\Bigg\{ \sum_{j=1}^{n_i} I(Y_{ij}=0) \Big(\frac{p_{ij}(\gamma)(1-p_{ij}(\gamma))(1-Z^{-1}(\lambda_{ij}(\beta,\sigma_b,b_{iq}),v)))}{p_{ij}(\gamma) + \frac{1-p_{ij}(\gamma)}{Z(\lambda_{ij}(\beta,\sigma_b,b_{iq}),v)}} Z_{ij} \Big) \ + I(Y_{ij} \geq 1) \times \\
&\quad - \frac{p_{ij}(\gamma)(1-p_{ij}(\gamma))}{1-p_{ij}(\gamma)} Z_{ij} \Bigg\} \psi_i(\theta; b_{iq}) \Bigg]
\end{aligned}
$$

$$
\begin{aligned}
3.\ \frac{\partial L_i}{\partial v} &= \frac{1}{\sqrt{\pi}} \int \Bigg\{ \sum_{j=1}^{n_i} I(Y_{ij}=0) \frac{-(1-p_{ij}(\gamma)) \sum_{s=0}^{\infty} \frac{log(s!)\lambda_{ij}(\beta,\sigma_b,b_i)^s}{(s!)^v}}{-\Big(p_{ij}(\gamma) + \frac{1-p_{ij}(\gamma)}{Z(\lambda_{ij}(\beta,\sigma_b,b_i),v)}\Big) Z(\lambda_{ij}(\beta,\sigma_b,b_i),v)^2} \ + I(Y_{ij} \geq 1) \times \\
&\quad \Bigg(\frac{\sum_{s=0}^{\infty} \frac{log(s!)\lambda_{ij}(\beta,\sigma_b,b_i)^s}{(s!)^v}}{Z(\lambda_{ij}(\beta,\sigma_b,b_i),v)} - log(y_{ij}!) \Bigg) \Bigg\} \psi_i(\theta) e^{-b_i^2} db_i \\
&\approx \frac{1}{\sqrt{\pi}} \sum_{q=1}^{Q} w_q \Bigg[\Bigg\{ \sum_{j=1}^{n_i} I(Y_{ij}=0) \frac{-(1-p_{ij}(\gamma)) \sum_{s=0}^{\infty} \frac{log(s!)\lambda_{ij}(\beta,\sigma_b,b_{iq})^s}{(s!)^v}}{-\Big(p_{ij}(\gamma) + \frac{1-p_{ij}(\gamma)}{Z(\lambda_{ij}(\beta,\sigma_b,b_{iq}),v)}\Big) Z(\lambda_{ij}(\beta,\sigma_b,b_{iq}),v)^2} \ + I(Y_{ij} \geq 1) \times \\
&\quad \Bigg(\frac{\sum_{s=0}^{\infty} \frac{log(s!)\lambda_{ij}(\beta,\sigma_b,b_{iq})^s}{(s!)^v}}{Z(\lambda_{ij}(\beta,\sigma_b,b_{iq}),v)} - log(y_{ij}!) \Bigg) \Bigg\} \psi_i(\theta; b_{iq}) \Bigg]
\end{aligned}
$$

$$
\begin{aligned}
4.\ \frac{\partial L_i}{\partial \sigma_b} &= \frac{1}{\sqrt{\pi}} \int \Bigg\{ \sum_{j=1}^{n_i} I(Y_{ij}=0) \frac{(1-p_{ij}(\gamma))\sqrt{2}b_i \sum_{s=0}^{\infty} \frac{s\lambda_{ij}(\beta,\sigma_b,b_i)^s}{(s!)^v}}{-\Big(p_{ij}(\gamma) + \frac{1-p_{ij}(\gamma)}{Z(\lambda_{ij}(\beta,\sigma_b,b_i),v)}\Big) Z(\lambda_{ij}(\beta,\sigma_b,b_i),v)^2} + I(Y_{ij} \geq 1) \times \\
&\quad \sqrt{2}\, b_i \Big\{ y_{ij} - \frac{\sum_{s=0}^{\infty} \frac{s\lambda_{ij}(\beta,\sigma_b,b_i)^s}{(s!)^v}}{Z(\lambda_{ij}(\beta,\sigma_b,b_i),v)} \Big\} \Bigg\} \psi_i(\theta) e^{-b_i^2} db_i
\end{aligned}
$$

$$\approx \frac{1}{\sqrt{\pi}} \sum_{q=1}^{Q} w_q \left[\left\{ \sum_{j=1}^{n_i} I(Y_{ij}=0) \frac{(1-p_{ij}(\gamma))\sqrt{2}\, b_{iq} \sum_{s=0}^{\infty} \frac{s\lambda_{ij}(\beta,\sigma_b,b_{iq})^s}{(s!)^v}}{-\left(p_{ij}(\gamma)+\frac{1-p_{ij}(\gamma)}{Z(\lambda_{ij}(\beta,\sigma_b,b_{iq}),v)}\right) Z(\lambda_{ij}(\beta,\sigma_b,b_{iq}),v)^2} + I(Y_{ij}\geq 1) \times \right.$$
$$\left. \sqrt{2}\, b_{iq} \left\{ y_{ij} - \frac{\sum_{s=0}^{\infty} \frac{s\lambda_{ij}(\beta,\sigma_b,b_{iq})^s}{(s!)^v}}{Z(\lambda_{ij}(\beta,\sigma_b,b_{iq}),v)} \right\} \right\} \psi_i(\theta; b_{iq}) \Bigg]$$

R-code

```
##############################################################################################
# This r-code is given as a underdispersion case of the maize hybrids data in Chapter 3      #
# This r-code describes non-parametric bootstrap method for obtaining 95 % naive confidence interval  #
# This r-code includes codes for Cardinal Research Clueter (CRC) server lines                #
##############################################################################################
# CRC code #
args <- commandArgs(trailingOnly = TRUE)
seq <- as.numeric(args[1])

Z<- function(lambda,v, max) {

  # Compute the terms used to sum for the (in)finite summation
     forans <- matrix(0,ncol=max+1,nrow=length(lambda))
  for (j in 1:max){
   temp <- matrix(0,ncol=j,nrow=length(lambda))
   for (i in 1:j){temp[,i] <- lambda/(i^c(v))}
   for (k in 1:length(lambda)){forans[k,j+1] <- prod(temp[k,])}
   }
  forans[,1] <- rep(1,length(lambda))
# Determine the (in)finite sum
  ans <- rowSums(forans)

return(ans)
}


lik_zicmp <- function(parm) {
          bhat <- as.matrix(parm[1:4])
          ghat <- as.matrix(parm[5:8])
          vhat <- as.matrix(parm[9])
          bsigma_hat <- parm[10]

          logLi <- numeric(N)
          i <- 1
          for(k in unique(corn_gene$id)){
            Li <- numeric(length(bi))
            for(b in 1:length(bi)){
              xmati <- as.matrix(allx[which(allx[,1]==k),-1])
              zmati <- xmati
              pij <- exp(zmati%*%ghat)/(1+exp(zmati%*%ghat))
              lhat <- exp(xmati%*%bhat+sqrt(2)*bsigma_hat*bi[b]+log(off[which(corn_gene$id==k)]))
              yi <- as.matrix(ally[which(ally[,1]==k),2])
              Lij <- ifelse(yi==0,(pij+(1-pij)/Z(lhat,vhat,100)),(1-
              pij)*lhat^yi/((factorial(yi)^c(vhat))*Z(lhat,vhat,100)))
             Li[b] <- prod(Lij)*wi[b]
                      }
              logLi[i] <- log(sum(Li)*1/sqrt(pi))
              i <- 1+1
               }
        return(-sum(logLi))
        }

library(pscl)
library(MASS)
```

```
library(matrixcalc)
library(fastGHQuad)
 rulebi <- gaussHermiteData(25)
 bi <- rulebi$x
 wi <- rulebi$w

corn <- read.table("/home/h0choo01/corn_glmm_ibs/genodata.txt",sep="\t",header=T)
dim(corn)
idxcorn <- apply(corn[,7:70],1, function(x)ifelse(mean(x==0)>=0.5 & mean(x==0) < 0.7 & max(x)
                <=4,1,0))
zerocorn <- which(idxcorn==1)
n <- 4
i <- 26
off <- apply(corn[,7:70],2,sum)
y <- matrix(as.numeric(corn[zerocorn[i],7:70]),64,1)  # pick arbitrary gene that is suspiciously zero-inflated
idx_maze <- rep(1:16, each=n)
y_all <- cbind(as.matrix(idx_maze), y)
onevec <- as.matrix(rep(1,nrow(y)))
BS <- 1
w <- 1
all_estbs_zicmp <- NULL
N <- length(y)
set.seed(350+seq)
 while( BS <6) {
            idx_bs <- sample(1:16, 16, replace=TRUE)
            onevec <- as.matrix(rep(1,nrow(y)))
            idx_xbs <- sample(rep(1:n,each=n),16,replace=FALSE)

            ybs <- matrix(sapply(idx_bs, function(x) y_all[which(y_all[,1]==x),2]), 64,1)
            fbs <- matrix(as.factor(rep(idx_xbs,each=4)), 64,1)
            gtype_fbs <- data.frame(gtype=fbs)
            xmat <- model.matrix(~ gtype, gtype_fbs)  # genotype effect
            zmat <- xmat
            corn_gene <- data.frame(GeneID=as.matrix(rep(corn[zerocorn[i],1],
            length(y))),id=as.matrix(rep(1:16, each=4)),read=ybs, gtype_fbs, row.names=NULL)
            allx <- as.matrix(data.frame(id=corn_gene[,2],xmat))
            ally <- as.matrix(corn_gene[,2:3])
            val <- tryCatch({
                 mZIP <- zeroinfl(formula=read ~ gtype+offset(log(off))|gtype, data=corn_gene,dist =
                                "poisson" )}
                                  ,error= function(e) e$message)
              error_idx <- list(w,value=val)

        if(length(error_idx$value)>1){
                beta0 <- as.matrix(summary(val)$coefficients$count[,1])
                gamma0 <-as.matrix(summary(val)$coefficients$zero[,1])
                v0 <- 1
                bsigma_hat0 <- 1
          val_zicmp <- tryCatch({
                     glmm_parm <- optim(c(beta0,gamma0,v0,bsigma_hat0), lik_zicmp,
                                        control=list(maxit=30000),method="L-BFGS-B", lower=c(rep(-
                                              Inf,8),0.001,0.001),upper=c(rep(Inf,10)))
                                    }, error= function(e) e$message)

          error_idx_zicmp <- list(w,value=val_zicmp)
```

```
        if(length(error_idx_zicmp$value)>1){
              bhat <- as.matrix(val_zicmp$par[1:4])
              ghat <- as.matrix(val_zicmp$par[5:8])
              vhat <- c(val_zicmp$par[9])
              bsigma_hat <- c(val_zicmp$par[10])
             if(val_zicmp$convergence==0){
                  all_estbs_zicmp <- rbind(all_estbs_zicmp,matrix(c(BS,bhat,ghat,vhat,bsigma_hat),1,11))
                  BS <- BS+1
                  w <- w+1
                 } else {all_estbs_zicmp <- rbind(all_estbs_zicmp,matrix(c(w,rep(NA,10)),1,11))
                       all_estbs_zicmp <- na.omit(all_estbs_zicmp)
                       w <- w+1
                     }} else {all_estbs_zicmp <- rbind(all_estbs_zicmp,matrix(c(w,rep(NA,10)),1,11))
                           all_estbs_zicmp <- na.omit(all_estbs_zicmp)
                           BS <- BS
                           w <- w+1
                        }} else {all_estbs_zicmp <- rbind(all_estbs_zicmp,matrix(c(w,rep(NA,10)),1,11))
                              all_estbs_zicmp <- na.omit(all_estbs_zicmp)
                              BS <- BS
                              w <- w+1
                            }
                    }

file.name <- paste("mazei_glmm_bs=",seq,".RData", sep="")
save(all_estbs_zicmp, val_zicmp,val,corn_gene, file=file.name)
q()
```

CHAPTER 4

A BAYESIAN APPROACH TO ZERO-INFLATED CLUSTERED COUNT DATA WITH DISPERSION

4.1 Bayesian Model

Our Bayesian hurdle model consists of two different parts, called presence model and severity model. The presence model considers a binary random variable for the non-zero outcome and the severity model describes the positive counts using a CMP distribution. It is important to be aware that while the presence model governs the existence of nonzero outcome, the severity model only governs the positive outcome to account for how severe the outcomes can be, given covariates.

The presence model is based on a probit regression with both fixed effects and random effects terms. The probability that an outcome is positive (non-zero) is modeled through probit regression, $P(Y_{ij} > 0) = \Phi(X_{ij,\beta}^T \beta + U_{ij,\delta}^T \delta_i)$. This model is associated with fixed effect covariates, X_{ij} and random effect covariates, U_{ij}, corresponding to the j^{th} observation in the i^{th} cluster. The q-dimensional random effects, δ_i, are assumed to be generated by a multivariate normal distribution with a mean vector with zeros and a variance-covariance matrix, Σ. A multivariate random effect allows flexibility in the dependence of observations within clusters.

The severity model starts with defining the usual probability mass function (pmf) of CMP,

$$P(Y = y) = \frac{\lambda^y}{(y!)^v Z(\lambda, v)},\ y = 0, 1, 2, \cdots,$$

where $Z(\lambda, v) = \sum_{s=0}^{\infty} \frac{\lambda^s}{(s!)^v}$ is the normalizing constant, and λ is a positive shape parameter. The parameter v indicates that the data are underdispersed if $v > 1$, overdispersed if $0 < v < 1$, or equidispersed if $v = 1$. When $v = 1$, the Z function becomes e^{λ} which implies that CMP distribution is equivalent to the Poisson distribution with mean λ. Then, the severity model, conditioning on the outcome being nonzero, is

$$P(Y_{ij} = y|\, y > 0) = \frac{(\lambda_{ij})^y}{(y!)^v \sum_{s=1}^{\infty} \frac{(\lambda_{ij})^s}{(s!)^v}},\ i = 1, \cdots,\ N,\ j = 1,\ \cdots, n_i, \tag{1}$$

where $log(\lambda_{ij}) = X_{ij,\alpha}^T \alpha$ is the linear predictor. It is important to be aware that the severity model uses a truncated CMP distribution which excludes zeros (Equation 1). The full distribution of Y_{ij} after combining the presence and severity models is, then,

$$\begin{cases} P(Y_{ij} = 0) = 1 - \Phi(X_{ij,\beta}^T \beta + U_{ij,\delta}^T \delta_i) \\ P(Y_{ij} = y) = \Phi(X_{ij,\beta}^T \beta + U_{ij,\delta}^T \delta_i) \times \frac{e^{y(X_{ij}^T \alpha)}}{(y!)^v \sum_{s=1}^{\infty} e^{s(X_{ij}^T \alpha)} (s!)^{-v}},\ y \geq 1 \end{cases} . \tag{2}$$

Clustering across outcomes is induced by the random effects $\delta_i \sim MVN(0, \Sigma)$ in the binary/presence model.

The prior distributions for β and Σ in the presence model are taken to be conjugate priors for the multivariate normal distributions:

$$\beta \sim MVN(0, \Omega_\beta), \tag{3}$$

$$\Sigma \sim IW(c, \Psi). \tag{4}$$

Likewise, the prior distributions for the severity model parameters are also defined as

$$\alpha \sim MVN(0, \Omega_\alpha) \tag{5}$$

$$v \sim logN(0, \sigma_v^2) \tag{6}$$

As Z function of a CMP distribution is not a closed form, conjugate priors are not readily available. We recommend that the prior distribution of the dispersion parameter v is to be a lognormal distribution (Equation 6) with a median/mode of v at 1, centered at equidispersion. We choose the variance of log(v) to be 0.5^2 so that 95% values are between 0.38 to 2.05.

4.2 MCMC (Markov chain Monte Carlo) Sampling

Inference under this model is performed by using MCMC sampling methods. Due to the conjugacy in the presence model, Gibbs Sampling steps are used to generate the samples. In sampling steps for the severity

model, conjugacy for the CMP distribution is not available, and the samples are generated by Metropolis-Hastings steps.

In the presence model, instead of estimating β and δ_i directly from the probit model, a data augmentation scheme (Chib and Greenburg, 1998) is applied by introducing a normally distributed latent vairable, $z_{ij} \sim N(X_{ij}^T\beta + U_{ij}^T\delta_i, 1)$. The value of Y_{ij} is determined by the sign of the latent variable: $Y_{ij} > 0$ if $z_{ij} > 0$ and $Y_{ij} = 0$ if $z_{ij} < 0$. In fact, the data augmentation allows Gibbs Sampling and gives better Monte-Carlo performance.

The sampling distributions for the presence model are obtained as below.

$$z_{ij}|\beta, \delta_i, y_{ij} \sim \begin{cases} TN(X_{ij,\beta}^T\beta + U_{ij,\delta}^T\delta_i,\ 1)_{(0,\infty)},\ y_{ij} > 0 \\ TN(X_{ij,\beta}^T\beta + U_{ij,\delta}^T\delta_i,\ 1)_{(-\infty,0),}\ y_{ij} = 0 \end{cases} \tag{7}$$

$$\beta|z, \delta \sim MVN\Big(\Big\{\Big(\sum_{i=1}^{N} X_i^T X_i\Big) + \Omega_\beta^{-1}\Big\}^{-1}\Big\{\sum_{i=1}^{N}(X_i^T z_i - X_i^T\delta_i U_i)\Big\}, \Big(\sum_{i=1}^{N} X_i^T X_i\Big) + \Omega_\beta^{-1}\Big\}^{-1}\Big) \tag{8}$$

$$\delta_i|\beta \sim MVN\Big(\{\Sigma + U_i^T U_i\}^{-1}(U_i^T z_i - U_i^T X_i\beta),\ \{\Sigma + U_i^T U_i\}^{-1}\Big) \tag{9}$$

$$\Sigma|\delta \sim IW(c + N, \sum_{i=1}^{N}\delta_i\delta_i^T + \Psi) \tag{10}$$

The notation, TN, in Equation 7 stands for a truncated normal distribution.

For generating posterior samples of the regression coefficients α in the severity model, we use both a global step and a local step to update. The global step seeks to change the full α vector, and in the local step only one component is updated at a time. For the global step, we propose the candidate value $\alpha^c \sim MVN(\alpha,\ \Omega_{q,\alpha})$ and accept with probability,

$$A_\alpha = min\left(1, \frac{\Big\{\prod_{i,j:y_{ij}>0} P(y_{ij}|\alpha^c, y_{ij} > 0, v)\Big\}\Pi(\alpha^c)}{\Big\{\prod_{i,j:y_{ij}>0} P(y_{ij}|\alpha, y_{ij} > 0, v)\Big\}\Pi(\alpha)}\right). \tag{11}$$

For the local step that updates element p, we propose α^c by taking $\alpha_p^c \sim N(\alpha_p, \sigma_{q,\alpha_p}^2)$ and $\alpha_j^c = \alpha_j \, (j \neq p)$. We accept the proposed α^c with probability,

$$A_{\alpha_p} = min\left(1, \frac{\left\{\prod_{i,j\,:y_{ij}>0} P(y_{ij}|\alpha_p^c, y_{ij}>0, v)\right\}\Pi(\alpha_p^c)}{\left\{\prod_{i,j\,:y_{ij}>0} P(y_{ij}|\alpha_p, y_{ij}>0, v)\right\}\Pi(\alpha_p)}\right). \tag{12}$$

For v, we use a pseudo random walk by drawing $v^c \sim logN(v, \sigma_{q,v}^2)$ and accept with probability,

$$A_v = min\left(1, \frac{\left\{\prod_{i,j:\,y_{ij}>0} P(y_{ij}|v^c, y_{ij}>0, \alpha)\right\}\Pi(v^c)q(v|v^c)}{\left\{\prod_{i,j:\,y_{ij}>0} P(y_{ij}|v, y_{ij}>0, \alpha)\right\}\Pi(v)q(v^c|v)}\right). \tag{13}$$

Variance parameters $\Omega_{q,\alpha}$, σ_{q,α_p}^2, and $\sigma_{q,v}^2$ are chosen so that accectence rate is about 25% for univariate steps and between 25% and 40% for global step (Robert et al., 1997).

To speed computation, we run the presence and severity models in parallel because their posterior are independent such as

$$\Pi(\beta, \delta, \Sigma, \alpha, v|y) = \Pi(\beta, \Sigma|y)\Pi(\alpha, v|y). \tag{14}$$

To see this, note

$$\begin{aligned}
\Pi(\beta, \alpha, \Sigma, v|y) &= \int \Pi(\beta, \alpha, \delta, \Sigma, v|y) d\delta \\
&\propto \int \Bigg\{ \prod_{i,j} \Big[1 - \Phi(X_{ij}^T\beta + U_{ij}^T\delta_i)\Big]^{I(Y_{ij}=0)} \\
&\qquad \times \Big[\Phi(X_{ij}^T\beta + U_{ij}^T\delta_i)\Big]^{I(Y_{ij}>0)} \Pi(\beta, \delta_i, \Sigma) \\
&\qquad \times \prod_{i,j} \Big[P(Y_{ij}|\alpha, v)\Big]^{I(Y_{ij}>0)} \Pi(\alpha, v) \Bigg\} d\delta \\
&\propto \left[\int \Pi(\beta, \delta, \Sigma|y) d\delta\right] \Pi(\alpha, v|y) \\
&= \Pi(\beta, \Sigma|y)\Pi(\alpha, v|y).
\end{aligned} \tag{15}$$

Hence, we can run MCMC on the presence model independently from the MCMC chain on the severity model.

4.3 Applications

The Iowa Fluoride Study is a longitudinal study for identifying risk and protective factors to teeth from children. It is obvious that teeth within the same child share the same dental environment, which implies clustering characteristics. This clustering feature is statistically incorporated by introducing random effects into a mixed effects model framework. In this section, we choose the nine-year old children's dataset which is the same dataset applied in Chapter 2.

In the dental field, the location of a tooth inside the mouth is believed to have a great effect on the likelihood of cavities or caries. In fact, cavities are more likely to occur in the molars than canines and incisors because molars have occlusal surfaces which are more easily able to retain foods. Additionally, the distal and mesial surfaces of molars are also more likely to keep foods between the surfaces. To account for this, a unique intercept for each tooth class is used in both presence and severity models. The eight covariates previously considered are also used. To induce clustering, we consider three correlated random effects corresponding to three tooth locations. Then, the design matrix for the random effects is given below.

$$U_{ij}^T = \begin{cases} (1,0,0) & : \text{Molar} \\ (0,1,0) & : \text{Canine and Premolar} \\ (0,0,1) & : \text{Incisor} \end{cases} . \tag{16}$$

We include premolars with the canines in the second group (Equation 16) due to the location of premolars next to canines at that premolars appear in the dataset as many nine-year-old children still have some primary teeth.

For this analysis, the hyperparameters for the priors are chosen to be $\Omega_\alpha = \Omega_\beta = 10 \cdot \mathrm{I}_{11}$, $c = 3$, $\Psi = 3^{-1} \cdot \mathrm{I}_3$, and $\sigma_v = 0.5$, and MCMC sampling is performed as described in Section 3. The posterior samples of the presence model parameters are obtained by running MCMC algorithm for 150,000 iterations, and the posterior samples of the severity model parameters are also generated with Markov chain of length 50,000 (the presence model displayed slower mixing). The samples are collected after first 5,000 burn-in and keeping every 10^{th} thinning. At the end, there are 14,500 samples for the presence model parameters

and 4,500 samples for the severity model parameters to be used for inference. The convergence in parameters is assessed through trace plots. The log-likelihood values for both models are also evaluated and display good mixing. The effective sample size for the log-likelihood functions from each model is founded to be greater than 1,000. The posterior means and 95% credible intervals are given in Table 4.1.

Coinciding with our expectations, we find molars to be much more likely to have cavities than canines or incisors ($Pr(\beta_1 > \beta_2|y) \approx 1.00$), $Pr(\beta_1 > \beta_3|y) \approx 1.00$). Furthermore, there is strong evidence that the daily fluoride intake (AUCmhF5_9yrs) and the frequency of brushing teeth per day (ToothBrushingFeq.Per_DayAvg) are protective factors for teeth. The daily carbonated beverage intake (AUCSodaOz5_9yrs) is found to be a risk factor ($Pr(\beta_7 > 0|y) = 0.99$). Similarly, in the severity model, the molar intercept (α_1) is found to be larger than the other teeth locations (α_2 and α_3). This indicates that molars with cavities have higher CES (caries experience score), on average, than canines or incisors with cavities. There is some evidence that older dental exam age (DentalExamAge) tends to have higher CES, but the effect size is small. Finally, the amount of fluoride in home tap water (HomeFluorideppmAvg) appears to be a protective factor in the severity model but was not significant in the presence model. This implies that the more fluoride intake supplied from home faucet water reduces the severity of cavities for decayed teeth, while the fluoride intake may not be preventing the tooth cavities. This result is important in light of ongoing debate within both the dental community and the public concerning the usefulness of fluoride in the public water.

We estimate the random effect covariance matrix, $\widehat{\Sigma} = [E(\Sigma^{-1}|y)]^{-1}$ (Yang and Berger, 1994), to be

$$\widehat{\Sigma} = \begin{bmatrix} 0.792 & 0.763 & 0.470 \\ 0.667 & 1.655 & 0.925 \\ 0.533 & 0.727 & 0.979 \end{bmatrix}, \tag{17}$$

where the variance-covariance components are displayed in the upper triangular part and the correlation coefficients for the lower off-diagonal part. The off-diagonal entries are positive indicating that all the cluster-specific tooth location effects have positive relationship with each other. In the severity model, we find strong evidence of overdispersion ($Pr(v < 1) \approx 1$).

4.4 Discussion

Through the model specification and application to the dental data (Sections 3.1 and 3.3), this chapter demonstrates that the Bayesian method has a great advantage of giving more flexible model structures and avoids the calculation of an approximated likelihood function.

In this chapter, we only consider random effects in the presence model, not in the severity model. As a part of future work with the dental dataset, an expended modeling approach will be performed such that random effects will be also incorporated into the severity model. The severity model will include a cluster-specific random effect that is correlated with δ_i form the presence model. Consequently, the variance-covariance matrix for all the random effect components accounts for random effects not only from the presence model but also from the severity model. Thus, posterior sampling must be performed jointly without running seperate MCMC chains. Preliminary results have been hindered by poor mixing, but alternative sampling methods may lead to improvements. This work is ongoing.

Table 4.1

Posterior means (post mean) and credible intervals (C.I.) for both presence and severity models

	Presence Model		Severity Model	
	post mean	C.I.	post mean	C.I.
Molar	-0.543	(-1.106, 0.027)	0.435	(0.218, 0.651)
Canine	-3.252	(-4.536, -2.245)	-0.119	(-0.455, 0.204)
Incisor	-4.252	(-5.925, -2.910)	0.168	(-0.347, 0.597)
Gender	-0.215	(-0.430, 0.001)	-0.012	(-0.071, 0.048)
DentalExamAge	0.121	(-0.030, 0.272)	0.056	(0.011, 0.102)
AUCmhF5_9yrs	-0.430	(-0.779, -0.080)	-0.021	(-0.133, 0.086)
AUCSodaOz5_9yrs	0.071	(0.027, 0.117)	-0.004	(-0.016, 0.008)
ToothBrushingFeq.Per_DayAvg	-0.568	(-0.793, -0.343)	-0.056	(-0.119, 0.008)
DentalVisitPast6monthAvg	0.256	(-0.286, 0.804)	0.026	(-0.150, 0.200)
FluorideTreatmentPast6monthAvg	0.304	(-0.058, 0.672)	0.021	(-0.090, 0.132)
HomeFluorideppmAvg	0.129	(-0.123, 0.382)	-0.127	(-0.205, -0.054)
v	N/A		0.424	(0.334, 0.517)

R-code

```
##############################################################################
# This r-code describes the MCMC samplings for the presence model for Chapter 4                    #
##############################################################################
library(MASS)
library(truncnorm)
library(matrixcalc)
library(mvtnorm)
library(MCMCpack)
library(coda)
# create another variable for random effects
caries <- read.csv("C:\\Users\\h0choo01\\Desktop\\caries.csv", header=T)
caries[,9] <- caries[,9]-9
molar <- c(3,14,1,2,15,16,"a","b","i","j",17:19,30:32,"k","l","s","t")
canine <- c(4:6,11:13,"r","m","c","h",20:22,27:29)
incisor <- c(7:10,"e","f",23:26,"p","o","d","g","q","n")
loc_t <- sapply(caries$Tooth, function(x) if(x %in% molar){res <- 1}else{if(x %in% canine){res <- 2}
                    else{res <-3}})
loc_t <- matrix(loc_t,nrow(caries),1)
caries_r <- cbind(caries, loc_t)
y <- as.matrix(caries$CariesCount)
umat <- model.matrix(~-1+as.factor(loc_t))
umat <- as.matrix(data.frame(molars=umat[,1], canine=umat[,2], incisor=umat[,3]))
xmat <- as.matrix(data.frame(umat,caries_r[, c(16, 9:15)] ))

all_data <- data.frame(id=caries$SID, xmat, y)
ids <- unique(all_data[,1])
pos_data <- subset(all_data, all_data$y>0)
y_pos <- pos_data$y
xmat_pos <- as.matrix(pos_data[,2:12])  # fixed factor
N <- length(unique(all_data[,1]))
sigma2b <- 10
cc <- ncol(umat)     # degree of freedom for IW distribution
psy <- diag(1/cc, cc, cc)    # positive-definite parameter for IW
sig_dg0 <- cc*psy  # var-cov matrix components for random effects :prior

b0 <- matrix(0, ncol(xmat),1)
d0 <- matrix(0,3,1)
z0 <- matrix(0,nrow(all_data),1)
for(i in 1:nrow(xmat)){
      z0[i,] <- abs(rnorm(1,xmat[i,]%*%b0+umat[i,]%*%d0, 1))*ifelse(y[i]==0,-1,1)
      }
di <- matrix(rnorm(N*ncol(umat),0,1),ncol(umat),N)
colnames(di) <-  ids
zaid <- matrix(0,3,1)
BB <- 150000
idx_thin <- seq(5001,BB,10)
all_sigmas <- array(0,c(3,3,length(idx_thin)))
all_tracks <- matrix(NA,length(idx_thin), length(b0)+2 )
bb <- 1
kk <- 1
watch <- Sys.time()

while(bb < BB+1){
          # Gibbs sampler for beta|Z, delta
```

```
mu1_b <- mu2_b <-0
for (i in ids){
    xmati <- as.matrix(all_data[which(all_data[,1]==i),2:12])
    umati <- as.matrix(all_data[which(all_data[,1]==i),2:4])
    yi <- as.matrix(all_data[which(all_data[,1]==i),13])
    dii <- di[,which(as.numeric(colnames(di))==i)]
    z0i <- as.matrix(z0[which(all_data[,1]==i),])
    mui1_b <- t(xmati)%*%xmati
    mui2_b <- t(xmati)%*%z0i-t(xmati)%*%umati%*%dii
    mu1_b <- mu1_b + mui1_b
    mu2_b <- mu2_b + mui2_b
     }
mu_b <- solve(mu1_b+diag(1/sigma2b,ncol(xmat)))%*%mu2_b
sigma2_bz <- solve(t(xmat)%*%xmat+diag(1/sigma2b,ncol(xmat)))
b1 <- matrix(rmvnorm(1,mu_b,  sigma2_bz), ncol(xmat),1)  # b|z,delta

# Gibbs sampler for Z|beta, delta
z1 <- NULL
for (zz in ids){
    xmati <- as.matrix(all_data[which(all_data[,1]==zz),2:12])
    umati <- as.matrix(all_data[which(all_data[,1]==zz),2:4])
    yi <- as.matrix(all_data[which(all_data[,1]==zz),13])
    dii <-   di[,which(as.numeric(colnames(di))==zz)]
    mu_zi <- xmati%*%b1+umati%*%dii
    z1i <- matrix(apply(cbind(mu_zi,yi),1,function(x) ifelse(x[2]==0,rtruncnorm(1,-Inf,0, x[1],1),
                    rtruncnorm(1,0,Inf, x[1],1))),nrow(yi),1)
    z1 <- rbind(z1,z1i)
    }
# Gibbs sampler for delta|beta, Z
sig_dd0 <- sig_dg0[1:3,1:3]
ii <- 1
for (i in ids){
   xmati <- as.matrix(all_data[which(all_data[,1]==i),2:12])
   umati <- as.matrix(all_data[which(all_data[,1]==i),2:4])
   zi<-  as.matrix(z1[which(all_data[,1]==i)])
   mu_di  <- t(solve(solve(sig_dd0)+t(umati)%*%umati))%*%(t(umati)%*%zi-
                    t(umati)%*%xmati%*%b1)
   var_di <- solve(solve(sig_dd0)+t(umati)%*%umati)
   di[,ii] <- as.matrix(mvrnorm(1,mu_di,var_di))
   ii <- ii+1
   }

# updating cov-var matrix   | delta
Lsum <- matrix(0,ncol(umat),ncol(umat))
for (i in 1:length(unique(all_data[,1]))){
     dii <- di[,i]
     Li <- matrix(dii,3,1)
     Lsum <- Lsum+Li%*%t(Li)}
sig_dg1 <- riwish(cc+N, psy+Lsum)

p1_all <- NULL
for (i in unique(all_data[,1])){
       xmati <- as.matrix(all_data[which(all_data[,1]==i),2:12])
       umati <- as.matrix(all_data[which(all_data[,1]==i),2:4])
       yi <- as.matrix(all_data[which(all_data[,1]==i),13])
```

```
            dii <- di[,which(as.numeric(colnames(di))==i)]
            p1 <- pnorm(xmati%*%b1+umati%*%dii)
            p1_all <- rbind(p1_all, p1)
            }
      logL_zero <- sum(apply(cbind(y,p1_all), 1, function(x) ifelse(x[1]==0, log(1-x[2]), log(x[2]))))
      if(bb %in% idx_thin){
            all_tracks[kk,] <- matrix(c(b1,log(det(sig_dg1)), logL_zero),1,length(b0)+2)
            all_sigmas[,,kk] <- sig_dg1
            kk <- kk+1
          }
       b0 <- b1
       z0 <- z1
       bb <- bb+1
       sig_dg0 <- sig_dg1
       }
Sys.time()-watch

# posterior mean and equal tail credible interval
round(apply(all_tracks,2,mean),3)
round(apply(all_tracks,2,function(x) quantile(x, c(0.025,0.975))),3)

library(coda)
effectiveSize(all_tracks)
post_sig <- apply(all_sigmas,c(1,2),mean)
post_corr <- diag(diag(post_sig)^(-0.5))%*%post_sig%*%diag(diag(post_sig)^(-0.5))
post_sig0 <- sapply(1:nrow(all_tracks), function(x) solve(all_sigmas[,,x]))
post_sig1 <- solve(matrix( apply(post_sig0,1,mean),3,3))

#############################################################################
# This r-code describes the MCMC samplings for the severity model for Chapter 4  #
#############################################################################
trunc_logZi<- function(xmat,b,v, maxi) { # xmat should be truncated xmat given y >0

    forans <- matrix(NA, nrow(xmat),maxi)
    for (i in 1:maxi){
      forans[,i]  <-  matrix(i*(xmat%*%b)-v*lgamma(i+1),nrow(xmat),1)
       }

    A <- apply(forans, 1,max)
    temp1 <- matrix(apply(forans, 2,function(x) x-A),nrow(xmat),maxi)
    ans <- log(rowSums(exp(temp1)))+A
  return(ans)
   }

trunc_logZ <- function(all_data,b,v, maxi) { # xmat should be truncated xmat given y >0
          ans <- 0
          for (i in unique(all_data[,1])){
                xmati <- as.matrix(all_data[which(all_data[,1]==i),2:12])

                yi <- as.matrix(all_data[which(all_data[,1]==i),13])

                forans <- matrix(NA, nrow(xmat),maxi)
                for (i in 1:maxi){
                  forans[,i]  <-  matrix(i*(xmat%*%b)-v*lgamma(i+1),nrow(xmat),1)
                   }
```

```
                A <- apply(forans, 1,max)
                temp1 <- matrix(apply(forans, 2,function(x) x-A),nrow(xmat),maxi)
                ansi <- log(rowSums(exp(temp1)))+A
                ans <- ans +ansi}
            return(ans)
    }
truncl_cmpi <- function(xmat,y,ahat,vhat) {
                    l_c <- y*(xmat%*%ahat)-vhat*lgamma(y+1)-trunc_logZi(xmat,ahat,vhat,100)
                    return(sum(l_c))
                     }
truncl_cmp <- function(all_data,ahat,vhat) {
            l_c <- 0
            for (i in unique(all_data[,1])){
                xmati <- as.matrix(all_data[which(all_data[,1]==i),2:12])
                yi <- as.matrix(all_data[which(all_data[,1]==i),13])
                li_c <- yi*(xmati%*%ahat)-vhat*lgamma(yi+1)-trunc_logZi(xmati,ahat,vhat,100)
                            l_c <- l_c+sum(li_c)
                   }
                return(sum(l_c))
                }
# create another variable for random effects
caries <- read.csv("C:\\Users\\h0choo01\\Desktop\\caries.csv", header=T)
caries[,9] <- caries[,9]-9
molar <- c(3,14,1,2,15,16,"a","b","i","j",17:19,30:32,"k","l","s","t")
canine <- c(4:6,11:13,"r","m","c","h",20:22,27:29)
incisor <- c(7:10,"e","f",23:26,"p","o","d","g","q","n")
loc_t <- sapply(caries$Tooth, function(x) if(x %in% molar){res <- 1}else{if(x %in% canine){res <- 2}
else{res <-3}})
loc_t <- matrix(loc_t,nrow(caries),1)
caries_r <- cbind(caries, loc_t)
y <- as.matrix(caries$CariesCount)
umat <- model.matrix(~-1+as.factor(loc_t))
umat <- as.matrix(data.frame(molars=umat[,1], canine=umat[,2], incisor=umat[,3]))
xmat <- as.matrix(data.frame(umat,caries_r[, c(16, 9:15)] ))

all_data <- data.frame(id=caries$SID, xmat, y)
ids <- unique(all_data[,1])
pos_data <- subset(all_data, all_data$y>0)
y_pos <- pos_data$y
xmat_pos <- as.matrix(pos_data[,2:12])  # fixed factor
N <- length(unique(all_data[,1]))

sigma2_acand <- round(diag(solve(t(xmat_pos)%*%(xmat_pos))),4)
sigma2_acand <- sigma2_acand/22
sigma2_ai <- sigma2_acand+c(0.001,0.17,0.4,0.008,0.005,0.005,0.00015,0.0015,0.003,0.005,0.003)
sigma_vcand <- 0.06
sigma2b <- sigma2a <- 10
v0 <- 1
alpha0 <- matrix(rep(0,11),ncol(xmat),1)
tcmpl_a0 <- truncl_cmp(pos_data,alpha0,v0)

BB <- 50000
# accecptance ratio for gamma over iterations
mha <- matrix(NA,BB,1+ncol(xmat))
mhv <- numeric(BB)
```

```
idx_thin <- seq(5001,BB,10)
all_tracks <- matrix(NA,length(idx_thin), length(alpha0)+2 )
bb <- 1
kk <- 1
while(bb < BB+1){
        a_cand <- matrix(mvrnorm(1,alpha0, diag(sigma2_acand,ncol(xmat))), ncol(xmat),1)
        v_cand <- rlnorm(1,log(v0),sigma_vcand)

        # truncated logL for a CMP distribution : y>0
        # for estimating alpha globally
        log_acand<- dmvnorm(c(a_cand), rep(0,ncol(xmat)),diag(sigma2a,ncol(xmat)), log=T)
        log_a0<- dmvnorm(c(alpha0), rep(0,ncol(xmat)),diag(sigma2a, ncol(xmat)), log=T)
        mh_a <- exp(truncl_cmp(pos_data,a_cand,v0)+ log_acand-tcmpl_a0-log_a0)
        u <- runif(1)
        if(mh_a>u){alpha0 <- a_cand
                tcmpl_a0 <- truncl_cmp(pos_data,alpha0,v0)
                mha[bb,1] <- 1          } else{ mha[bb,1] <- 0}
        # for estimating alpha individually
        tcmpl_a0i <- tcmpl_a0
        for(j in 1:length(a_cand)){
          a_cand <- alpha0
          a_cand[j] <- rnorm(1, alpha0[j],sqrt(sigma2_ai[j]))
          log_acandi <- dmvnorm(c(a_cand), rep(0,ncol(xmat)),diag(sigma2a,ncol(xmat)), log=T)
          log_ai0<- dmvnorm(c(alpha0), rep(0,ncol(xmat)),diag(sigma2a, ncol(xmat)), log=T)

          mh_aj <- exp(truncl_cmp(pos_data,a_cand,v0)+ log_acandi-tcmpl_a0i-log_ai0)
        # if mh_a=0, accept alpha0
          u <- runif(1)
          if(mh_aj>u){alpha0 <- a_cand
                tcmpl_a0i <- truncl_cmp(pos_data,alpha0,v0)
                mha[bb,j+1] <- 1          } else{ mha[bb,j+1] <- 0}
                }

        log_vcand <- dlnorm(v_cand, 0,0.5, log=T)
        log_v0 <- dlnorm(v0,0,0.5,log=T)
        logq_vcand <- dlnorm(v0,log(v_cand),sigma_vcand, log=T)
        logq_v0 <- dlnorm(v_cand,log(v0), sigma_vcand, log=T)
        mh_v <- min(1,exp(truncl_cmp(pos_data,alpha0,v_cand)+log_vcand+logq_vcand-tcmpl_a0i-
                log_v0-logq_v0))
        u <- runif(1)
       if(mh_v>u){
            v0 <- v_cand
            tcmpl_a0i <- truncl_cmp(pos_data,alpha0,v0)
            mhv[bb] <- 1      } else{mhv[bb] <- 0}

       logL_pos <- truncl_cmp(pos_data,alpha0,v0)
       tcmpl_a0 <- logL_pos

  if(bb %in% idx_thin){
       all_tracks[kk,] <- matrix(c(alpha0, v0,logL_pos),1,length(alpha0)+2)

     kk <- kk+1
      }
       bb <- bb+1
        }
```

```
apply(mha,2,mean)
mean(mhv)
round(apply(all_tracks,2,mean),3)
round(apply(all_tracks,2,function(x) quantile(x, c(0.025,0.975))),3)
library(coda)
effectiveSize(all_tracks)
```

CHAPTER 5

FUTURE PLAN

One of our future plans is to expand the usefulness of a CMP model by applying it to NGS (Next Generation Sequencing) data. Although we have used in for analyzing two genes in this dissertations, the plan is to undertake a full scale analysis of all genes in the essay. Since different types of dispersion truly exist on different genes, our suggested CMP model is an ideal choice in terms of accounting for each gene's dispersion and obtaining more accurate effects. In order to identify significant genes, it is important to correctly adjust the test results for multiple comparisons. For the final procedure, four performance measures can be considered: sensitivity, specificity, false discovery rate (FDR), and false nondiscovery rate (FNR). With these measures, we can also determine the effectiveness of a CMP model based analysis through simulation studies. We also plan to investigate the effectiveness of the CMP-seq analysis through biological validations of the findings.

For Chapter 4, we use the hurdle model for the Bayesian approach method, rather than the zero-inflated model framework. Another possible future plan would be to perform a Bayesian modeling based on a zero-inflated framework and to compare the results of both hurdle and zero-inflated Bayesian models from simulations and/or applications.

This dissertation has been providing the three different methodologies (Chapters 2-4) based on a CMP distribution. In the future we will consider extending our methodologies by using different types of count distributions.

REFERENCES

Barriga, G. D. C., and Louzada, F. (2014). The zero-inflated Conway-Maxwell-Poisson distribution: Bayesian inference, regression modeling and influence diagnostic. Statistical Methodology 21, 23-34.

Blocker, A. W., 2014. fastGHQuad: Fast Rcpp implementation of Gauss-Hermite quadrature. https://cran.rproject.org/web/packages/fastGHQuad/fastGHQuad.pdf

Böhning, D. (1998). Zero-inflated Poisson models and C.A.MAN: a tutorial collection of evidence. Biometrical Journal 40, 833-843.

Cheung, Y. B. (2002). Zero-inflated models for regression analysis of count data: a study of growth and development. Statistics in Medicine 21, 1461-1469.

Chib, S., and Greenberg, E. (1998). Analysis of multivariate probit models. Biometrika 85, 347-361.

Choo-Wosoba, H., Levy, S. M., and Datta, S., 2016. Marginal regression models for clustered count data based on zero-inflated Conway-Maxwell-Poisson distribution with applications. Biometrics. doi: 10.1111/biom.12436

Conway, R. W., and Maxwell, W. L. (1962). A queuing model with state dependent service rates, Journal of Industrial Engineering 12, 132-136.

Dalrymple, M. L., Hudson, I. L., and Ford, R. P. K. (2003). Finite Mixture, Zero-inflated Poisson and Hurdle models with application to SIDS. Computational statistics & Data analysis 41, 491-504.

Famoye, F., and Singh, K. P. (2006). Zero-Inflated Generalized Poisson Regression Model with an Application to Domestic Violence Data. Journal of Data Science 4, 117-130.

Field, C. A., and Welsh, A. H. (2007). Bootstrapping clustered data. Journal of Royal Statistical Society Ser B, 69, 369-390.

Fournier, D. A., Skaug, H. J., Ancheta, J., Ianelli, J., Magnusson, A., Maunder, M. N., Nielsen, A., and Sibert, J. (2012). AD Model Builder: using automatic differentiation for statistical inference of highly parameterized complex nonlinear models. Optim. Methods Softw. 27, 233-249.

Fulton, K. A., Liu D., Haynie, D. L., and Albert P. S. (2015). Mixed model and estimating equation approaches for zero inflation in clustered binary reponse data with application to a dating violence study. The Annals of Applied Statistics 9, 275-299.

Hall, D. B., (2000). Zero-inflated Poisson and binomial regression with random effects: a case study. Biometrics 56, 1030-1039.

Hall, D. B., and Zhang, Z. (2004). Marginal models for zero-inflated clustered data. Statistical Modelling 4, 161-180.

Hasan, M. T., Sneddon G., and Ma, R., 2009. Pattern-mixeture zero-inflated mixed models for longitudinal unbalanced count data with excessive zeros. Biometrics 51, 946-960.

Jackman, S. (2006). Package 'pscl'. Technical Report. Stanford, CA: Political Science Computational Laboratory, Stanford University.
http://cran.r-project.org/web/packages/pscl/pscl.pdf

Jansakul, N., and Hinde, J. P. (2002). Score tests for zero-inflated Poisson models. Computational Statistics & Data Analysis 40,75-96.

Kong, M., Xu, S., Levy, S. M., and Datta, S. (2015). GEE type inference for clustered zero-inflated negative binomial regression with application to dental caries. Computational Statistics and Data Analysis, 85, 54-66.

Kutner, M. H., Nachtsheim, C. J., and Neter, J. (2003). Applied Linear Regression Models, 4^{th} ed., McGraw-Hill, New York.

Lambert, D. (1992). Zero-inflated Poisson regression, with an application to defects in manufacturing. Technometrics, 1-14

Levy, S. M., Warren, J. J., Broffitt, B. A., Hillis, S. L., and Kanellis, M. J. (2003). Fluoride, beverages and dental caries in the primary dentition. Caries Research 37, 157-165.

Liang, K. Y., and Zeger, S. L. (1986). Longitudinal data analysis using generalized linear models. Biometrika 73, 13-22.

Lesaffre, E., and Spiessens, B. (2001). On the effect of the number of quadrature points in a logistic random-effects model: an example. Applied Statistics 50, 325-335.

Lord, D., Guikema, S. D., and Geedipally, S. R. (2008). Application of the Conway-Maxwell-Poisson generalized linear model for analyzing motor vehicle crashes. Accident Analysis & Prevention 40, 1123-1134.

McLachlan, G. J. (1997). On the EM algorithm for overdispersed count data. Statistical Methods in Medical Research 6, 76-98.

Paschold, A., Larson N. B., Marcon, C., Schnable, J. C., Yeh, C. T., Lanz C., Nettleton, D., Piepho, H. P., Schnable P. S., and Hochholdinger, F. (2014). Nonsyntenic genes drive highly dynamic complementation of gene expression in mazie hybrids. The Plant Cell 26, 3939-3948.

Roberts, G. O., Gelman, A., Gilks, W. R. (1997). Weak convergence and optimal scaling of random walk Metropolis algorithms. The Annals of Applied Probability 7, 110-120.

Rodrigue-Motta, M., Gianola, D., and Heringstad, B. (2010). A mixed effects model for overdispersed zero inflated poisson data with an application in animal breeding. Journal of Data Science 8, 379-396.

Rose, C. E., Martina, S. W., Wannemuehlerb, K. A., and Plikaytisb, B. D. (2006). On the Use of Zero-Inflated and Hurdle Models for Modeling Vaccine Adverse Event Count Data. Journal of Biopharmaceutical Statistics 16, 463-481.

Rosen, O., Jiang, W., and Tanner, M. A. (2000). Mixtures of marginal models. Biometrika 87, 391-404.

SAS (13.1). http://support.sas.com/documentation/cdl/en/etsug/66840/HTML/default/viewer.htm#etsug_countreg_details15.htm

SAS. https://support.sas.com/documentation/cdl/en/statugglmmix/61788/PDF/default/statugglmmix.pdf

Satten, G. A., Datta, S. (2000). The S-U algorithm for missing data problems. Computational Statistics 15, 243-277.

Sellers, K. F. and Lotze, T. (2010). http://cran.r-project.org/web/packages/COMPoissonReg/COMPoissonReg.pdf

Sellers, K. F., and Shmueli, G. (2010). A flexible regression model for count data, The Annals of Applied Statistics 4, 943-961.

Shmueli, G., Minka, T. P., Kadane, J. B., Borle, S., and Boatwright, P. (2005). A useful distribution for fitting discrete data: revival of the Conway–Maxwell–Poisson distribution. Applied Statistics 54, 127-142.

Xie, M., He, B., and Goh, T. N. (2001). Zero-inflated Poisson model in statistical process control. Computational statistics & Data analysis 38, 191-201.

Yang, R. and Berger, J. (1994). Estimation of the covariance matrix using the reference prior. The Annals of Statistics 22, 1195-1211.

Yau, K. K. W., Wang, K., and Lee, A. H. (2003). Zero-inflated negative binomial mixed regression modeling of over-dispersed count data with extra zeros. Biometrical Journal 45, 437-452.

APPENDIX

I would like to acknowledge that this dissertation research was supported by National Institutes of Health grants. In addition, I would like to thank Steven Levy and Dan Nettleton for sharing Iowa Fluoride Study data and the maize hybrids data, respectively.

CURRICULUM VITA

NAME: HYOYOUNG CHOO-WOSOBA

ADDRESS: DEPARTMENT of Bioinformatics and Biostatistics
485 E. Gray Street
University of Louisville, KY 40202

DOB: Seoul, Republic of Korea - January 23, 1984

EDUCATION: Ph.D. (2012-May, 2016), Bioinformatics and Biostatistics,

University of Louisville, Louisville

M.S. (2007-2009), Statistics, University of Nebraska, Lincoln

B.A. (2002-2006), Applied Statistics, KonKuk University, Seoul

PUBLICATIONS: Choo-Wosoba, H., Levy, S., and Datta, S. (2015). Marginal regression models for clustered count data based on zero-inflated Conway-Maxwell-Poisson distribution with applications. (Biometrics, in press)

Sikdar, S., Choo-Wosoba, H., Abdia,Y., Dutta, S., Gill, R., Datta, S., and Datta, S.(2014). An Integrative exploratory analysis of -omics data from the ICGC cancer genomes lung adenocarcinoma study. Systems Biomedicine 2, 56-64.

Choo-Wosoba, H., and Datta, S. Mixed effect models for clustered count data based on zero-inflated Conway-Maxwell-Poisson distribution - preprint

Choo-Wosoba, H., Datta, S., and Gaskins, J. Bayesian approach to inference for clustered count data based on a Zero-Inflated Conway-Maxwell-Poisson Distribution. - in preparation

Honors/Awards: Travel Award in 2015 Joint Statistical Meeting (JSM) in Seattle

Travel Award in 2014 Statistical and Applied Mathematical Sciences Institute (SAMSI) Workshop

Pass with Distinction in Ph.D. Comprehensive Exam at University of Louisville (2014)

Presentations: A contributed paper oral presentation in Joint Statistical Meeting (JSM) in Seattle (2015): Maximum Pseudo-Likelihood and GEE-Type Inference for Clustered Count Data Based on Zero-Inflated Conway-Maxwell Poisson Distribution with Application to the Iowa Fluoride Study

An oral presentation as the departmental colloquium in the department of Bioinformatics and Biostatistics at University of Louisville (2015): Inference for Clustered Count Data based on Zero-Inflated Conway-Maxwell-Poisson Distribution with Application to the Iowa Fluoride Study

Poster presentation in Statistical and Applied Mathematical Sciences Institute (SAMSI) Workshop (2014): A Comprehensive Omics Study for the ICGC Cancer Genomes Lung Adenocarcinoma Data