5-2014

A Reduced Bias Method of Estimating Variance Components in Generalized Linear Mixed Models

Elizabeth A. Claassen
University of Nebraska-Lincoln, eclaasse@gmail.com

A REDUCED BIAS METHOD OF ESTIMATING VARIANCE COMPONENTS IN GENERALIZED LINEAR MIXED MODELS

by

Elizabeth A. Claassen

A DISSERTATION

Presented to the Faculty of

The Graduate College at the University of Nebraska

In Partial Fulfillment of Requirements

For the Degree of Doctor of Philosophy

Major: Statistics

Under the Supervision of Professor Walter W. Stroup

Lincoln, Nebraska

May, 2014

A REDUCED BIAS METHOD OF ESTIMATING VARIANCE COMPONENTS IN GENERALIZED LINEAR MIXED MODELS

Elizabeth A. Claassen, Ph.D.

University of Nebraska, 2014

Adviser: Walter W. Stroup

In small samples it is well known that the standard methods for estimating variance components in a generalized linear mixed model (GLMM), pseudo-likelihood and maximum likelihood, yield estimates that are biased downward. An important consequence of this is that inferences on fixed effects will have inflated Type I error rates because their precision is overstated. We introduce a new method for estimating parameters in GLMMs that applies a Firth bias adjustment to the maximum likelihood-based GLMM estimating algorithm. We apply this technique to one- and two-treatment logistic regression models with a single random effect. We show simulation results that demonstrate that the Firth-adjusted variance component estimates are substantially less biased than maximum likelihood estimates and that inferences using the Firth estimates maintain their Type I error rates more closely than the standard methods.

ACKNOWLEDGEMENTS

Many thanks go to my advisers: Dr. Walt Stroup at the University of Nebraska and Dr. Chris Gotwalt at JMP. Without you I would be taking classes the rest of my life instead of holding a completed dissertation.

Thanks also go to the other members of my committee: Dr. Erin Blankenship, Dr. Steve Dunbar and Dr. Steve Kachman. Having also served as my masters committee, you have been with me since the beginning, and your support though the last five years has been invaluable.

To my parents, Alan and Janet. Thank you for the meals, cat sitting and general encouragement throughout this graduate school process.

To my brother Dave, MD PhD. If it were not for our sibling rivalry, this document would not exist. From one Dr. Claassen to another, thank you. Also, the book was crucial for laugh therapy.

To my sister Kat. You were too far away to experience directly all the highs and lows of this process, but you were always there just a Facebook or text message away when I needed you.

Special thanks also to Marie Gaudard and Sheila Loring for encouraging and improving my writing while interning at JMP in 2013. Also thanks to Nicole Jones for allowing me a big first project of documenting JMP features rather than the testing I was hired to do.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Generalized linear mixed models (GLMMs) are the most complex members of the linear models family. They combine the dual challenges of a non-Gaussian distribution with the nontrivial variance/covariance matrices of mixed models. These models have three main components that require estimation. They are the fixed effects, $\boldsymbol{\beta}$, the random effects, $\mathbf{b}$, and the variance components, $\boldsymbol{\sigma}$. This dissertation focuses on the estimation of the vector of variance components.

When it comes to estimating all of these components for GLMMs, there are two major non-Bayesian approaches currently in use: linearization and integral approximation. Linearization methods have an advantage that they can be made "REML-like" in their estimation process by replacing $\mathbf{y}$ in the linear mixed model equations with a pseudo-variable $\mathbf{y}^*$. Disadvantages of the linearization approach include no true fit statistics such as the Akaike information criterion (AICc) or the Bayesian information criterion (BIC) for model selection, and it is not well suited to many GLMMs. Integral approximation is in some ways preferable. Because it uses the true likelihood, fit statistics may be calculated. However, integral approximation is strictly a maximum likelihood (ML) method and, therefore, has the same

disadvantages ML has in Gaussian linear mixed models. Namely, the variance component estimates are downwardly biased, sometimes extremely biased such as in experiments with small numbers of replications.

While correcting the bias of the variance component estimate may be a goal in itself, the true impact of the bias correction is found in the hypothesis tests and confidence intervals, which depend upon the variance estimate. If the variance estimate is too small, test statistics are inflated, resulting in over-rejection of the hypothesis being tested. Similarly, confidence intervals are too narrow, with coverage probabilities lower than the stated value. The flawed inference resulting from these test statistics and confidence intervals could have devastating impacts depending on the situation.

Firth (1993) proposed a method of estimation that corrects for the bias of maximum likelihood. Gotwalt (2012) showed that for certain classes of linear mixed models restricted maximum likelihood (REML) is a Firth estimator. The goal of this dissertation is to apply the Firth correction method to GLMMs to obtain REML-like estimates via integral approximation. This method combines the advantages of REML and inference based on a true likelihood. The focus will be on a random intercept logit model for several reasons. The first reason is the relative simplicity of the model. Second, even though the model itself is simple, it exhibits many of the problematic characteristics of GLMMs, such as volatility with small sample sizes (Stroup 2013a). Finally, Heinze and Schemper (2002) showed that the Firth estimator is superior to the MLE for fixed effect logistic regression. For this reason we think the Firth estimator will work in a mixed effect logistic regression.

In chapter 2, we lay out a brief overview of model estimation methods beginning with the basic linear model and building to the GLMM. In chapter 3, we derive the

estimator for the random intercept logit model. In chapter 4, we investigate the estimator's behavior in several balanced case scenarios as well as for a series of missing data cases. Finally in chapter 5, we investigate a simple two-treatment scenario for the method's Type I error and power properties and confidence interval coverage under balanced data.

Chapter 2

A History of Linear Models and Their Estimation Methods

2.1 In the Beginning

It starts the first time sample variance is taught in the classroom. A hand goes up somewhere, "Why do you divide by n minus 1 and not n?" In a non-majors course, the answer might be explained as "it is the way it is to get an accurate estimate when we're also estimating the mean." In a calculus-based majors course, the question might be initially hand-waved away. However, during discussions of unbiasedness and maximum likelihood the sample variance is one of the first examples of a biased estimator. The bias issue is not limited to the sample variance but recurs in all linear models. From fixed effect only Gaussian linear models to Gaussian linear mixed models and GLMMs, the bias of variance component estimates is a common problem that must be addressed. This has been done successfully in many cases but not at all in others.

Eisenhart (1947) defined the types of effects in a linear model as Class I (fixed) and

Class II (random). This established the basis for modern linear model theory. Searle (1971) and Graybill (1976) wrote seminal textbooks establishing the language and notation for linear model theory. Searle, Casella, and McCulloch (1992) provided a comprehensive survey of variance component estimation for Gaussian linear mixed models.

Following these precedents with respect to the variance estimation problem, we divide linear models into categories.

1. The original General Linear Model. This model includes a linear predictor, $\boldsymbol{\eta} = \mathbf{X}\boldsymbol{\beta}$, an identity link function, $\boldsymbol{\mu} = \boldsymbol{\eta}$, and $\mathbf{y} \sim \mathrm{N}(\boldsymbol{\mu}, \mathbf{I}\sigma^2)$. The only variance to be estimated is σ^2.

2. The Gaussian Linear Mixed Model. This model is formulated as either a marginal or conditional model. The marginal model includes a linear predictor, $\boldsymbol{\eta} = \mathbf{X}\boldsymbol{\beta}$, an identity link function, $\boldsymbol{\mu} = \boldsymbol{\eta}$, and $\mathbf{y} \sim \mathrm{N}(\boldsymbol{\mu}, \mathbf{V})$. $\mathbf{V}$ is a function of $\boldsymbol{\sigma}$, a vector of at least two (co)variance components to be estimated. The conditional model modifies the linear predictor, $\boldsymbol{\eta} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\mathbf{b}$ with $\mathbf{b} \sim \mathrm{N}(\mathbf{0}, \mathbf{G})$, and $\mathbf{y}|\mathbf{b} \sim \mathrm{N}(\boldsymbol{\mu}, \mathbf{R})$, but maintains the identity link, $\boldsymbol{\mu} = \boldsymbol{\eta}$,. The variance component vector, $\boldsymbol{\sigma}$, contains the elements of $\mathbf{R}$ and $\mathbf{G}$ requiring estimation. Note that the conditional model may be equivalently expressed as a marginal model with $\mathbf{V} = \mathbf{Z}\mathbf{G}\mathbf{Z}' + \mathbf{R}$.

3. The Generalized Linear Mixed Model. In the GLMM, we have $\boldsymbol{\eta} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\mathbf{b}$ with $\mathbf{b} \sim \mathrm{N}(\mathbf{0}, \mathbf{G})$, $\boldsymbol{\eta} = g(\boldsymbol{\mu})$ and $\mathbf{y}|\mathbf{b} \sim G(\boldsymbol{\mu}, \mathbf{R})$. Variance estimation again focuses upon $\boldsymbol{\sigma}$, the vector of components in $\mathbf{G}$ and $\mathbf{R}$, as applicable. This dissertation addresses this variance estimation problem, specifically the components of $\mathbf{G}$.

For all members of the linear model family, we define models with the linear predictor, that is a function of the mean of the distribution, μ, equal to a linear combination of the parameters; a link function; and distributional assumptions of the observations (given the random effects) and the random effects (Stroup 2013). Approaches to estimation vary for the different models and include ordinary least squares (OLS), generalized least squares (GLS), and maximum likelihood (ML) based methods.

2.2 Linear Models

Students first learn the basic "general" linear model, which includes the linear predictor $\boldsymbol{\eta} = \mathbf{X}\boldsymbol{\beta}$ with the identity link function, $\boldsymbol{\mu} = \boldsymbol{\eta}$, and $\mathbf{y} \sim \mathrm{N}(\boldsymbol{\mu}, \mathbf{I}\sigma^2)$. The traditional method of solving this linear model is through ordinary least squares (OLS), that is minimizing the squared distance between the observed responses, $\mathbf{y}$, and the fitted responses, $\mathbf{X}\hat{\boldsymbol{\beta}}$:

$$\begin{aligned}
\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 &= (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})'(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) \\
&= \boldsymbol{\beta}'\mathbf{X}'\mathbf{X}\boldsymbol{\beta} - 2\boldsymbol{\beta}'\mathbf{X}'\mathbf{y} + \mathbf{y}'\mathbf{y} \\
&= (\boldsymbol{\beta} - (\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{y})'(\mathbf{X}'\mathbf{X})(\boldsymbol{\beta} - (\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{y}) + \mathbf{y}'\mathbf{y} - \mathbf{y}'\mathbf{X}(\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{y}
\end{aligned}$$

The distance is clearly minimized when $\boldsymbol{\beta} = (\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{y}$.

$\left(\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}\right)' \left(\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}\right)$ is the sum of squares for error (SSE). In ordinary least squares the degrees of freedom for SSE are $N - \mathrm{rank}(\mathbf{X})$, so the mean square for error (MSE) is $\frac{SSE}{N - \mathrm{rank}(\mathbf{X})}$. The expected value of $\frac{SSE}{N - \mathrm{rank}(\mathbf{X})}$ is σ^2, so it is a logical estimator for σ^2.

Another option for solving this model is through maximum likelihood estimation. The density, f, and likelihood, L, functions are

$$f\left(\mathbf{y}|\boldsymbol{\beta},\sigma^2\right) = L(\boldsymbol{\beta},\sigma^2|\mathbf{y}) = (2\pi)^{-N/2}\left|\mathbf{I}\sigma^2\right|^{-\frac{1}{2}} e^{\frac{(\mathbf{y}-\mathbf{X}\boldsymbol{\beta})'(\mathbf{y}-\mathbf{X}\boldsymbol{\beta})}{-2\sigma^2}};$$

and then the log-likelihood is

$$\ell\left(\boldsymbol{\beta},\sigma^2|\mathbf{y}\right) = -\frac{N}{2}\ln(2\pi) - \frac{N}{2}\ln\left(\sigma^2\right) - \frac{\mathbf{y}'\mathbf{y} - 2\boldsymbol{\beta}'\mathbf{X}'\mathbf{y} + \boldsymbol{\beta}'\mathbf{X}'\mathbf{X}\boldsymbol{\beta}}{2\sigma^2}.$$

To maximize the log-likelihood, we take the derivative with respect to each of the parameters, set the derivatives equal to 0 and solve:

$$\begin{aligned}
\tfrac{\partial \ell}{\partial \boldsymbol{\beta}} &= \tfrac{\mathbf{X}'\mathbf{y}-\mathbf{X}'\mathbf{X}\boldsymbol{\beta}}{\sigma^2} \\
\tfrac{\mathbf{X}'\mathbf{y}-\mathbf{X}'\mathbf{X}\hat{\boldsymbol{\beta}}}{\hat{\sigma}^2} &= 0 \\
\mathbf{X}'\mathbf{y} &= \mathbf{X}'\mathbf{X}\hat{\boldsymbol{\beta}} \\
(\mathbf{X}'\mathbf{X})^{-}\mathbf{X}'\mathbf{y} &= \hat{\boldsymbol{\beta}}
\end{aligned}$$

and

$$\begin{aligned}
\tfrac{\partial \ell}{\partial \sigma^2} &= -\tfrac{N}{2\sigma^2} + \tfrac{(\mathbf{y}-\mathbf{X}\boldsymbol{\beta})'(\mathbf{y}-\mathbf{X}\boldsymbol{\beta})}{2(\sigma^2)^2} \\
-\tfrac{N}{2\hat{\sigma}^2} + \tfrac{\left(\mathbf{y}-\mathbf{X}\hat{\boldsymbol{\beta}}\right)'\left(\mathbf{y}-\mathbf{X}\hat{\boldsymbol{\beta}}\right)}{2(\hat{\sigma}^2)^2} &= 0 \\
\hat{\sigma}^2 &= \tfrac{\left(\mathbf{y}-\mathbf{X}\hat{\boldsymbol{\beta}}\right)'\left(\mathbf{y}-\mathbf{X}\hat{\boldsymbol{\beta}}\right)}{N}.
\end{aligned}$$

Notice the estimator, $\hat{\boldsymbol{\beta}}$, for $\boldsymbol{\beta}$ is the same for both least squares and maximum likelihood estimation. The estimator, $\hat{\sigma}^2$, for σ^2 differs in the denominator. The least squares estimator is unbiased for σ^2, so the MLE is biased downward for σ^2. This downward bias means confidence intervals are too narrow, test statistics are too large and error rates for inference exceed the nominal rate.

Suppose that $\mathbf{y}$ is distributed more generally, $\mathbf{y} \sim \mathrm{N}(\boldsymbol{\mu}, \mathbf{V}\sigma^2)$, where $\mathbf{V}$ is a symmetric positive definite matrix. The least squares process may be completed to arrive at the generalized least squares (GLS) estimator, $\hat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{V}^{-1}\mathbf{X})^{-}\mathbf{X}'\mathbf{V}^{-1}\mathbf{y}$, provided $\mathbf{V}$ is known. As in OLS, the MSE is the obvious estimator for σ^2.

2.3 Linear Mixed Models

For the linear mixed model (LMM), we have $\boldsymbol{\eta} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\mathbf{b}$ with $\mathbf{b} \sim \mathrm{N}(\mathbf{0}, \mathbf{G})$, the identity link, $\boldsymbol{\mu} = \boldsymbol{\eta}$, and $\mathbf{y}|\mathbf{b} \sim \mathrm{N}(\boldsymbol{\mu}, \mathbf{R})$. Some of the first mixed models were variance component only models where $\mathbf{Zb}$ in the linear predictor can be written as $\sum \mathbf{Z}_i\mathbf{b}_i$ and

$$\mathbf{G} = \begin{bmatrix} \mathbf{I}\sigma_1^2 & & & \\ & \mathbf{I}\sigma_2^2 & & \\ & & \ddots & \\ & & & \mathbf{I}\sigma_r^2 \end{bmatrix}.$$

Henderson (1953) proposed three ANOVA-based methods roughly comparable to least squares for estimating the variance components in this model. Method 1 directly equates sums of squares to the expected mean squares to identify estimators for the variance components. However, this method may only be used with random models, that is $\boldsymbol{\eta} = \mathbf{Zb}$, not full mixed models. Method 2 is similar to the first method but allows for fixed effects in situations where there is no interaction between a fixed and a random effect. To accommodate the fixed effects, the method adjusts the data for the fixed effects and then estimates the variance components as in Method 1. This idea of accounting for the fixed effects recurs in REML estimation. Finally, Method 3 can be used with any variance component only mixed model including those with crossed fixed and random effects. Method 3 uses

reduction sums of squares to obtain expected values that are equated to the variance components. This formulation allows for the crossed effects. For many years, the first two estimation methods were primarily used for their relative simplicity and ease of calculation; the third was more computationally intensive due to large matrix inversions. One theoretically pleasing result of these ANOVA estimators is they are unbiased. A problem with them, however, is the possibility of negative variance estimates.

When $\mathbf{G}$ or $\mathbf{R}$ become more complex, Henderson's methods are no longer applicable. Given the linear mixed model mentioned at the beginning of this section, the marginal distribution of $\mathbf{y}$ (that is, the distribution after integrating out the random effects) is $\mathbf{y} \sim \mathrm{N}(\boldsymbol{\mu}, \mathbf{V} = \mathbf{ZGZ}' + \mathbf{R})$. Henderson et al. (1959) defined the mixed model equations

$$\begin{bmatrix} \mathbf{X}'\mathbf{R}^{-1}\mathbf{X} & \mathbf{X}'\mathbf{R}^{-1}\mathbf{Z} \\ \mathbf{Z}'\mathbf{R}^{-1}\mathbf{X} & \mathbf{Z}'\mathbf{R}^{-1}\mathbf{Z} + \mathbf{G}^{-1} \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{\beta}} \\ \hat{\mathbf{b}} \end{bmatrix} = \begin{bmatrix} \mathbf{X}'\mathbf{R}^{-1}\mathbf{y} \\ \mathbf{Z}'\mathbf{R}^{-1}\mathbf{y} \end{bmatrix} \tag{2.3.1}$$

which lead to solutions

$$\begin{aligned} \hat{\boldsymbol{\beta}} &= \left(\mathbf{X}'\mathbf{V}^{-1}\mathbf{X}\right)^{-}\mathbf{X}'\mathbf{V}^{-1}\mathbf{y} \\ \hat{\mathbf{b}} &= \mathbf{GZV}^{-1}\left(\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}\right), \end{aligned}$$

where $\mathbf{V}^{-1} = \mathbf{R}^{-1} - \mathbf{R}^{-1}\mathbf{Z}\left(\mathbf{Z}'\mathbf{R}^{-1}\mathbf{Z} + \mathbf{G}^{-1}\right)^{-1}\mathbf{Z}'\mathbf{R}^{-1}$. Notice the estimator $\hat{\boldsymbol{\beta}}$ is the same as the GLS estimator in section 2.2. In this case, $\mathbf{V}$ is the marginal variance of $\mathbf{y}$. Clearly, $\mathbf{G}$ and $\mathbf{R}$ need to be estimated in order to solve the mixed model equations (2.3.1). Two modern options are available. The first option, maximum likelihood, follows the same process as in the linear model. Instead of the single variance parameter, σ^2, all of the parameters in $\mathbf{G}$ and $\mathbf{R}$ require estimation. The

full likelihood over all parameters (fixed and random) reduces to a likelihood involving only the parameters of $\mathbf{G}$ and $\mathbf{R}$.

$$\ell(\mathbf{G}, \mathbf{R}) = -\frac{1}{2}\log|\mathbf{V}| - \frac{1}{2}\mathbf{r}'\mathbf{V}^{-1}\mathbf{r} - \frac{n}{2}\log(2\pi), \tag{2.3.2}$$

where $\mathbf{r} = \mathbf{y} - \mathbf{X}\left(\mathbf{X}'\mathbf{V}^{-1}\mathbf{X}\right)^{-}\mathbf{X}'\mathbf{V}^{-1}\mathbf{y}$. As with maximum likelihood for the fixed effect linear model, the variance component estimates are often biased downward. The downwardly biased variance estimators have an important consequence: the variance of the fixed effects is underestimated leading to inflated Type I errors rates. This problem led to the development of the second option, restricted or residual maximum likelihood (REML).

Patterson and Thompson (1971) derived REML for LMMs. The principle behind REML is to maximize the likelihood after taking the fixed effects into account. Instead of maximizing the likelihood for $\mathbf{y} \sim \mathrm{N}(\boldsymbol{\mu}, \mathbf{V})$, REML maximizes a likelihood for $\mathbf{K}'\mathbf{y} \sim \mathrm{N}(0, \mathbf{K}'\mathbf{V}\mathbf{K})$. $\mathbf{K}'$ may be any matrix such that $\mathrm{E}[\mathbf{K}'\mathbf{y}] = \mathbf{0}$ and $\mathrm{rank}\,[\mathbf{K}'] = n - \mathrm{rank}\,[\mathbf{X}]$. With the fixed effects removed, the restricted log-likelihood, denoted with the subscript R, is

$$\ell_R(\mathbf{G}, \mathbf{R}) = -\frac{1}{2}\log|\mathbf{V}| - \frac{1}{2}\log\left|\mathbf{X}'\mathbf{V}^{-1}\mathbf{X}\right| - \frac{1}{2}\mathbf{r}'\mathbf{V}^{-1}\mathbf{r} - \frac{n-p}{2}\log(2\pi), \tag{2.3.3}$$

where p is the rank of $\mathbf{X}$ and $\mathbf{r}$ is as defined above. REML estimates are in many cases unbiased, though for some complex covariance structures the bias is only reduced. Also, in particular problems which result in negative variance estimates being set to zero, the estimates are no longer unbiased.

Corbeil and Searle (1976) provided a Newton-Raphson algorithm for calculating the estimates due to the necessity of iterating between solving the mixed model

equations and maximizing the likelihood for $\mathbf{G}$ and $\mathbf{R}$. The algorithm is theoretically simple but potentially complex in implementation as we will see in section 3.3. The steps are

1. Initialize the covariance parameters.
2. Set up the $\mathbf{R}$, $\mathbf{G}$ and $\mathbf{V}$ matrices and compute their inverses.
3. Solve the mixed model equations.
4. Calculate the derivative of the (restricted) log-likelihood with respect to the covariance parameters, also known as the score vector or gradient vector.
5. Calculate the Hessian matrix, the matrix of second derivatives of the (restricted) log-likelihood with respect to the covariance parameters.
6. Update the covariance parameters using the formula $\boldsymbol{\sigma} = \boldsymbol{\sigma} + \mathbf{H}^{-1}\mathbf{s}$ where $\mathbf{H}$ is the Hessian and $\mathbf{s}$ is the score vector.
7. Check for convergence.
8. Iterate.

Harville (1977) derived the elements of the second derivative matrix for both maximum likelihood and restricted maximum likelihood for use in the Newton-Raphson process. He also derived the expected second derivative matrix for use in Fisher scoring, a similar iterative method to Newton-Raphson that uses the Fisher Information matrix instead of the Hessian matrix.

In addition, Corbeil and Searle (1976) commented upon the downward bias of the MLEs for the variance components compared to the REML estimates. This is

comparable to the bias seen in the simple linear model case with the same effects. Therefore, REML is the preferred estimation method for LMMs.

Kackar and Harville (1984) proposed a correction to the variance/covariance matrix of the fixed effects to adjust for the underestimation of the standard errors of fixed effects. This correction approximates the bias in the variance/covariance matrix with quantities approximate to the second-order Taylor series approximation to the bias term. The correction fixes the covariance matrix at the end of the estimation process. Kenward and Roger (1997) continued this work and derived an adjusted estimator for the covariance matrix that has reduced bias in small sample settings. In addition, they derived a degrees of freedom adjustment for the purposes of inference using the approximate F-distribution.

Outside of mixed models research, Firth (1993) derived a general technique for creating an estimator with less bias than the MLE. This method is an additive correction to the score equation that eliminates the highest order bias term. It is a preventative method rather than corrective. That is, it is used in the estimation process to prevent bias rather than a correcting adjustment at the end, such as the Kackar-Harville correction. The corrected bias is a direct result of the properties of the score equation, $S(\boldsymbol{\theta}) = \frac{\partial}{\partial \boldsymbol{\theta}} L(\boldsymbol{\theta}|\mathbf{y})$. At the parameters, $\boldsymbol{\theta}_0$, the expected value of the score is zero. Therefore the MLE is the inverse score evaluated at zero:

$$\begin{aligned} \mathrm{E}_{\boldsymbol{\theta}_0}[S(\boldsymbol{\theta}_0)|\mathbf{y}] &= 0 \\ \hat{\boldsymbol{\theta}}_{MLE} &= S^{-1}(0|\mathbf{y}). \end{aligned}$$

In general, if f is a nonlinear function, then $\mathrm{E}[f(\mathbf{y})] \neq f(\mathrm{E}[\mathbf{y}])$. As seen above, the score equation is unbiased, so when $S(\boldsymbol{\theta})$ is nonlinear in the parameters, the MLE is biased. The Firth correction induces bias into the score equation to eliminate bias

in the MLE. The adjusted score equation is $S^*(\boldsymbol{\theta}) = S(\boldsymbol{\theta}) + A(\boldsymbol{\theta})$ where there are a couple of variations upon the additive piece, $A(\boldsymbol{\theta})$.

Firth proposed two variations of the adjustment. The first, the expected Firth adjustment, uses the expected Hessian, or Fisher Information, matrix. The element of the adjustment associated with the j^{th} parameter is

$$\mathbf{A}_{\theta j} = -\frac{1}{2}\operatorname{tr}\left(\mathbf{F}^{-1}\mathrm{E}\left[S_{\theta j}(\mathbf{H} - \mathbf{S}\mathbf{S}^T)\right]\right). \tag{2.3.4}$$

$\mathbf{F}^{-1}$ is the inverse Fisher Information matrix, $\mathbf{H}$ is the Hessian matrix, $\mathbf{S}$ is the score vector and S_{θ_j} is the element of the score vector corresponding to the parameter θ_j.

Because the Hessian matrix has been computed already and the expectation can be computationally intensive, Firth also proposed the alternative observed Firth adjustment. This modifies the expected adjustment to use the observed Hessian matrix, $\mathbf{H}$, rather than the expected Hessian, $\mathbf{F}$:

$$\mathbf{A}_{\theta j} = -\frac{1}{2}\operatorname{tr}\left(\mathbf{H}^{-1}\mathrm{E}\left[S_{\theta j}(\mathbf{H} - \mathbf{S}\mathbf{S}^T)\right]\right). \tag{2.3.5}$$

Gotwalt (2012) showed that REML estimators for variance component only LMMs are Firth estimators. For LMMs with more complex covariance structures, such as first-order autoregressive or first-order antedependent, the Firth-REML equivalence does not hold.

2.4 Generalized Linear Model

Nelder and Wedderburn (1972) defined the Generalized Linear Model (GLM) as $\boldsymbol{\eta} = \mathbf{X}\boldsymbol{\beta}$; $\boldsymbol{\eta} = g(\boldsymbol{\mu})$ where common link functions of the mean, $g(\boldsymbol{\mu})$, are the logit

and probit for binomial responses and log for count data, and $\mathbf{y} \sim G(\boldsymbol{\mu}, \mathbf{R})$ where G is a general function of the mean and covariance. This model expanded the types of data that could be analyzed beyond those that were approximately normal (or could be transformed to be approximately normal) to any exponential family. A distribution is said to be a member of the exponential family if its pdf can be written as

$$f(\mathbf{y}|\boldsymbol{\theta}) = \exp\left[\frac{\mathbf{y}\boldsymbol{\theta} - b(\boldsymbol{\theta})}{a(\phi)} + c(\mathbf{y}, \phi)\right]$$

or in its log-likelihood form

$$\ell(\boldsymbol{\theta}|\mathbf{y}, \phi) = \frac{\mathbf{y}\boldsymbol{\theta} - b(\boldsymbol{\theta})}{a(\phi)} + c(\mathbf{y}, \phi), \tag{2.4.1}$$

where $\boldsymbol{\theta}$ is the canonical parameter, for example $\ln\left(\frac{p}{1-p}\right)$ for the binomial distribution, and ϕ is a scale parameter.

2.5 Generalized Linear Mixed Model

Breslow and Clayton (1993) combined all of the above ideas to form the Generalized Linear Mixed Model (GLMM). For the GLMM the linear predictor is $\boldsymbol{\eta} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\mathbf{b}$ with $\mathbf{b} \sim \mathrm{N}(\mathbf{0}, \mathbf{G})$, $\boldsymbol{\eta} = g(\boldsymbol{\mu})$ and $\mathbf{y}|\mathbf{b} \sim G(\boldsymbol{\mu}, \mathbf{R})$. $G(\boldsymbol{\mu}, \mathbf{R})$ is again some general function of the mean and covariance. They proposed two quasi-likelihood methods for estimation - penalized quasi-likelihood and marginal quasi-likelihood. At the same time Wolfinger and O'Connell (1993) presented a pseudo-likelihood (PL) approach. With both quasi-likelihood and pseudo-likelihood, REML-like or maximum likelihood-like estimating equations are used. The mixed model equations are applied to the quasi- or pseudo-variable and iteratively solved. PL directly follows Harville (1977), replacing $\mathbf{y}$ with $\mathbf{y}^* = \tilde{\boldsymbol{\eta}} + \tilde{\mathbf{D}}^{-1}\left(\mathbf{y} - \left(\tilde{\boldsymbol{\mu}}|\tilde{\mathbf{b}}\right)\right)$, the

pseudo-variable, and var($\mathbf{y}$) with var($\mathbf{y}^*$). This pseudo-variable comes from the Taylor series expansion of the inverse link function. For full details of the derivation see Stroup (2013) sections 4.3 and 4.5. Schall (1991) provided an expectation-maximization (EM) algorithm that is not as efficient as PL. One main detraction from these methods is that they do not use the true likelihood. It is unclear how these quasi-likelihoods should obtain parameter estimates near the true parameters of the full likelihood.

An alternative to the pseudo-likelihood methods is maximum likelihood. If we assume that the general function G is from the exponential family, call it f, then McCulloch (1997) defined the likelihood

$$L(\boldsymbol{\theta}|\mathbf{y}) = \int \prod_{i=1}^{n} f_{y_i|b}(y_i|\mathbf{b}, \boldsymbol{\theta}) f_b(\mathbf{b}|\mathbf{D}) d\mathbf{b} \tag{2.5.1}$$

where $\mathbf{D}$ are the parameters in the distribution of $\mathbf{b}$. He provided two Monte Carlo algorithms for estimating this likelihood, and simulations using them demonstrated the downward bias of MLEs. Unlike the LMM, where the product of two normal distributions integrates to a normal distribution, the GLMM marginal distribution is not analytically tractable in general. Therefore, a numerical method of integral approximation is necessary if MLEs are desired. The most common methods are the Laplace approximation or adaptive Gaussian quadrature. Pinheiro and Bates (1995) compared several approximation methods including Laplacian and quadrature. Noting that the Laplace approximation is equivalent to quadrature with one abscissa, they found that the two methods provide efficient and accurate solutions up to a "reasonable" number of abscissas for adaptive quadrature. Larger numbers of abscissas do not improve accuracy greatly and increase computation time. MLEs for variance components in GLMMs have similar downward bias as those for LMMs.

2.6 Prior studies demonstrating variance estimate bias in GLMMs

Analytic maximum likelihood estimators for the variance with Gaussian data have been proven to be downwardly biased (see Casella and Berger (2002) pg 331 for an example). Simulation evidence has shown clearly that the MLE is downwardly biased for non-Gaussian data. The extent to which MLE bias is present and affects inference in generalized linear mixed models is documented by several authors.

Breslow and Clayton (1993) studied the behavior of the estimates obtained from their penalized quasi-likelihood procedure. They found that with correlated binary data, the variance components were consistently underestimated, particularly when the number of binary observations per subject was small. Breslow and Lin (1995) calculated asymptotic results for penalized quasi-likelihood and first and second order Laplace approximation estimation methods confirming the observed bias discussed in Breslow and Clayton. While deriving efficient Laplacian and adaptive quadrature algorithms for estimating parameters in GLMMs, Pinheiro and Chao (2006) investigated the methods' behavior in a more complex binary response model with nested random effects. They found the maximum likelihood estimates obtained for the variance components were "severely biased" (page 74).

The previous studies provided evidence of variance component estimate bias, but they did not address the impact of that bias upon fixed effect inference. Stroup, (2013a) and Couton and Stroup (2013) gauged the impact of this bias on Type I error and power. Stroup investigated a beta-binomial model with a randomized complete block design (RCBD) and found that using quadrature resulted in inflated Type I error rates. Confidence interval coverage was well below the stated level,

particularly with large cluster sizes. He found similar results for a Poisson-Normal model. Couton and Stroup looked at the gamma and beta distributions in conjunction with an RCBD and a more complex split plot design. Quadrature was able to control Type I error with both distributions in conjunction with the RCBD. However, when the experiment design became more complicated, coverage probabilities for the confidence intervals failed to meet the stated level. In these two studies, the variance was not studied directly. The results suggest bias in the variance component estimates, and the inference issues noted are a consequence of this bias.

2.7 Conclusions

Linear modeling has grown substantially more complex in the last 20-30 years as advances in computing power enabled the fitting of models previously infeasible. The generalized linear mixed model provides the opportunity to model non-Gaussian responses without transforming them to be closer to Normally distributed. The methods to estimate the parameters in these models have competing pros and cons. The benefit of maximum likelihood is that it works with the true likelihood rather than a pseudo- or quasi-likelihood. However, the maximum likelihood estimator for the variance is known to be downwardly biased. This impacts fixed effect inference by inflating test statistics, leading to a loss of Type I error control and narrowing confidence intervals. REML addresses the bias for Gaussian linear mixed models. There is no consensus correction for the bias in GLMMs as REML is for LMMs. The research initiated in this dissertation is intended to begin the development of a single, general technique for parameter estimation of generalized linear mixed models. Although we develop the estimator

and demonstrate its superior inferential properties for two rather simple special cases, the one- and two-sample simple random effects logistic model, the technique is very general and can be adapted to a much wider variety of models.

Chapter 3

Random Intercept Model Simulations and Estimator Derivation

3.1 Introduction

In this chapter, we focus on a specific GLMM, the random intercept logit model. As mentioned in chapter 1, this model was chosen for its relative simplicity, yet it exhibits common GLMM behavior. We look first at the model's behavior under various common current estimation methods to better understand the problem. Then we derive the derivatives necessary to compute the Firth estimator for this model. Finally, we write a program to calculate the new estimator.

3.2 Results of variance simulations regarding random intercept model.

Estimation of the binomial rate parameter is often a research question of interest. Here, the parameter is denoted p to distinguish it from the mathematical constant

π. Suppose that the objective is to estimate the population proportion of individuals with a specific disease or condition. Researchers randomly choose r sites. At each site, they sample n individuals independently to determine whether a disease or condition is present. Alternatively, we could choose r subjects and n independent trials per subject. However, each site has an impact upon the rate at that site. This leads us to the random intercept model:

$$\text{logit}\,(p_i) = \eta + b_i. \tag{3.2.1}$$

The $\text{logit}(p_i) = \ln\left(\frac{p_i}{1-p_i}\right)$ is the canonical parameter for the binomial distribution for linear models, η is the intercept of the linear model, and b_i is the effect of the i^{th} subject, assumed to be Normally distributed with mean 0 and variance σ^2. For ease in notation later, we will rewrite equation 3.2.1 as:

$$\text{logit}\,(p_i) = \eta + \sigma z_i \tag{3.2.2}$$

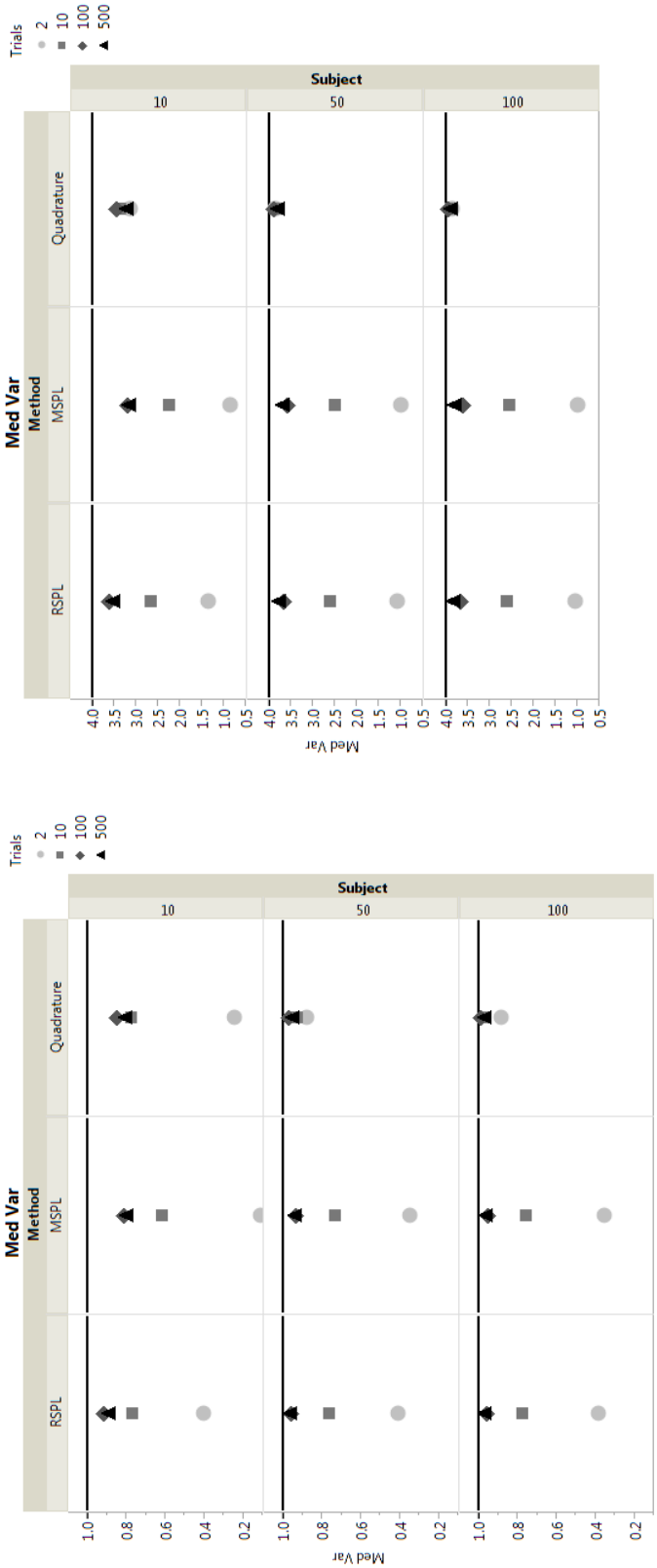
where the z_i denote standard normal random variables. Applying the inverse link function, we have:

$$p_i = \frac{e^{\eta+\sigma z_i}}{1+e^{\eta+\sigma z_i}} = \frac{1}{1+e^{-(\eta+\sigma z_i)}}.$$

While previous studies showed evidence of estimation bias in a variety of more complex GLMMs, we want to establish the baseline of the problem for this simple model with a small simulation study. For three common estimation methods, Residual Pseudo-likelihood (RSPL), Adaptive Quadrature and Maximum Pseudo-likelihood (MSPL), we examined performance in estimating η and σ^2 with various subject and sample size combinations. These methods were used with the default convergence criteria in SAS PROC GLIMMIX. Quadrature was the only

method where not all 1,000 simulated data sets converged. This non-convergence only occurred in the 10 subject, two observations per subject combination. When σ^2 was set to 1, 984 of 1,000 converged; when σ^2 was set to 4, 993 of 1,000 converged. As expected, all methods improve in their estimation of both the intercept, the fixed effect, and the variance, the random effect, as the number of subjects increases. Figures 3.2.1 and 3.2.2 illustrate the median variance and intercept estimates with the two subject distributions. Table 3.2.1 shows results when the subject variance is 1. Table 3.2.2 shows results when the subject variance is 4. The tables include both the mean and median of the estimates in 1,000 simulated experiments, due to the robustness of the median to outliers; some mean variance estimates are clearly extreme.

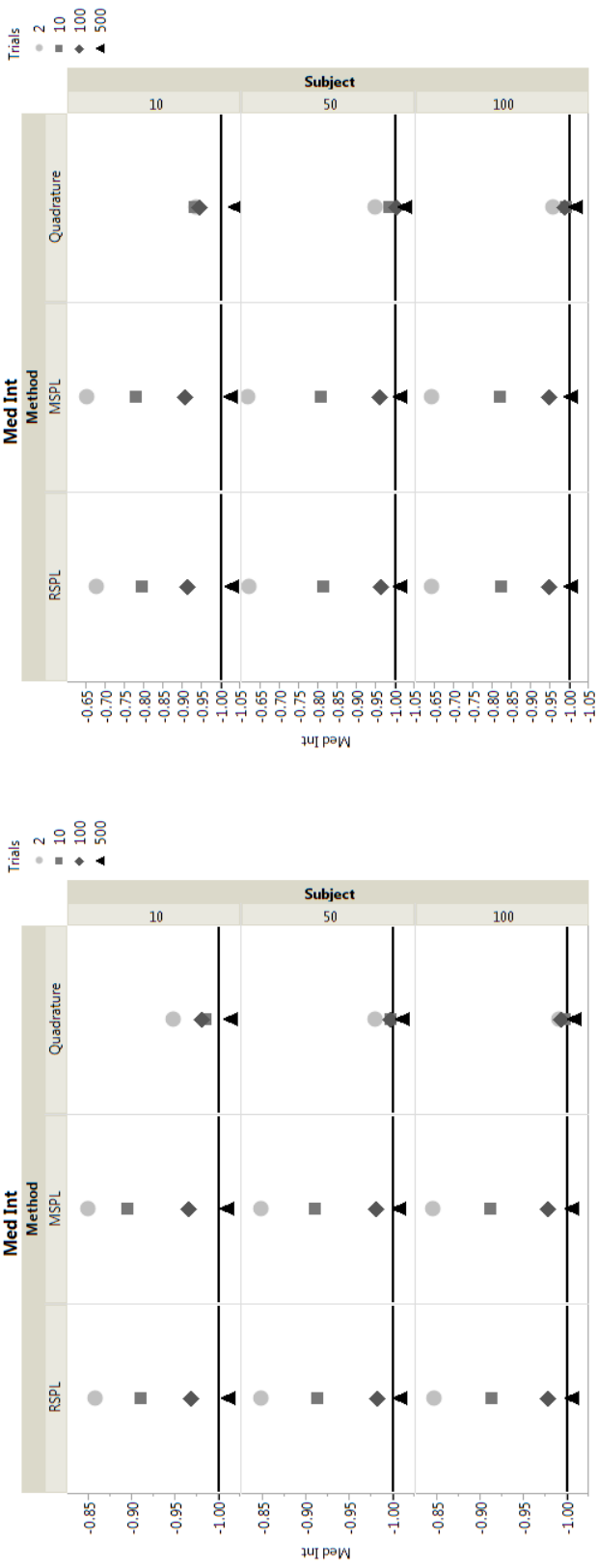
When both the number of random subjects and the number of trials per subject are large, all estimation methods perform comparably well. Conversely, when r and n are both small, all estimation methods tend to severely underestimate the variance. However, a preference is apparent in the cases where r is large and n is small and vice versa. For large r and small n, quadrature outperforms the pseudo-likelihood methods with variance estimates much closer to the true value; pseudo-likelihood still is underestimating the variance. In the reverse, RSPL obtains variance estimates closer to the true value while the maximum likelihood methods, MSPL and quadrature, exhibit the MLE downward bias. Table 3.2.3 summarizes the estimation method preferences.



(a) For $b \sim \mathrm{N}(0, 1)$

(b) For $b \sim \mathrm{N}(0, 4)$

Figure 3.2.1: Plots of median variance estimates



(a) For $b \sim \mathrm{N}(0, 1)$

(b) For $b \sim \mathrm{N}(0, 4)$

Figure 3.2.2: Plots of median intercept estimates

Table 3.2.1: Simulation Results for $b \sim N(0,1)$

		RSPL				MSPL				Quadrature			
subjects	trials	mean σ^2 est.	med. σ^2 est.	mean η est.	med. η est.	mean σ^2 est.	med. σ^2 est.	mean η est.	med. η est.	mean σ^2 est.	med. σ^2 est.	mean η est.	med. η est.
10	2	0.816	0.406	-0.94	-0.86	0.523	0.114	-0.93	-0.85	126.7	0.246	-1.71	-0.95
	10	0.894	0.768	-0.91	-0.91	0.726	0.618	-0.89	-0.89	0.981	0.776	-0.97	-0.98
	100	0.997	0.920	-0.98	-0.97	0.885	0.816	-0.98	-0.97	0.922	0.847	-0.99	-0.98
	500	0.977	0.892	-1.01	-1.01	0.877	0.800	-1.01	-1.01	0.886	0.807	-1.02	-1.01
50	2	0.455	0.411	-0.85	-0.85	0.399	0.350	-0.85	-0.85	1.327	0.882	-1.02	-0.98
	10	0.800	0.764	-0.91	-0.91	0.769	0.734	-0.91	-0.91	0.989	0.931	-1.00	-1.00
	100	0.977	0.960	-0.98	-0.98	0.955	0.938	-0.98	-0.98	0.993	0.975	-0.99	-1.00
	500	0.980	0.964	-1.01	-1.01	0.960	0.943	-1.01	-1.01	0.969	0.952	-1.01	-1.01
100	2	0.418	0.384	-0.84	-0.85	0.389	0.354	-0.84	-0.85	1.153	0.889	-1.01	-0.99
	10	0.792	0.775	-0.91	-0.91	0.777	0.760	-0.91	-0.91	0.996	0.967	-1.00	-1.00
	100	0.969	0.962	-0.98	-0.98	0.958	0.951	-0.98	-0.98	0.996	0.990	-0.99	-0.99
	500	0.983	0.975	-1.00	-1.01	0.973	0.965	-1.00	-1.01	0.982	0.973	-1.01	-1.01

Table 3.2.2: Simulation Results for $b \sim N(0, 4)$

		RSPL				MSPL				Quadrature			
Subjects	Trials	mean σ^2 est.	med. σ^2 est.	mean η est.	med. η est.	mean σ^2 est.	med. σ^2 est.	mean η est.	med. η est.	mean σ^2 est.	med. σ^2 est.	mean η est.	med. η est.
10	2	1.556	1.354	-0.71	-0.67	1.061	0.875	-0.69	-0.65	596.7	3.129	-1.67	-0.93
	10	2.982	2.680	-0.80	-0.79	2.499	2.256	-0.78	-0.78	4.458	3.368	-0.97	-0.93
	100	3.824	3.602	-0.95	-0.91	3.388	3.201	-0.94	-0.91	3.752	3.447	-0.99	-0.95
	500	3.882	3.516	-1.03	-1.03	3.477	3.157	-1.02	-1.03	3.591	3.234	-1.04	-1.04
50	2	1.105	1.085	-0.63	-0.62	1.014	0.996	-0.63	-0.62	4.713	3.865	-0.99	-0.95
	10	2.668	2.601	-0.81	-0.81	2.579	2.513	-0.81	-0.81	4.006	3.784	-0.99	-0.99
	100	3.718	3.655	-0.95	-0.96	3.633	3.570	-0.95	-0.96	3.967	3.896	-0.99	-1.00
	500	3.856	3.774	-1.02	-1.01	3.776	3.696	-1.02	-1.01	3.880	3.798	-1.03	-1.02
100	2	1.053	1.042	-0.64	-0.64	1.008	0.998	-0.64	-0.64	4.294	3.870	-0.99	-0.96
	10	2.635	2.604	-0.82	-0.82	2.591	2.560	-0.82	-0.82	3.984	3.887	-1.00	-0.99
	100	3.690	3.654	-0.95	-0.95	3.648	3.611	-0.95	-0.95	3.975	3.938	-0.99	-0.99
	500	3.867	3.838	-1.01	-1.00	3.826	3.799	-1.01	-1.00	3.929	3.887	-1.02	-1.02

Table 3.2.3: Estimation method of choice for varying r and n

		n	
		Small	Large
r	Small	??	PL
	Large	Quadrature	Either

3.3 Newton-Raphson and Broyden solvers for the random intercept binomial GLMM.

Before adding the Firth correction to the estimation process, we first develop a program to implement the quadrature algorithm used by both SAS PROC GLIMMIX and R's `lmer` function in the lme4 package to find the MLE. This insures a solid foundation on which to implement the Firth adjustment. To begin, we need the log-likelihood equation and the Gauss-Hermite quadrature equation. Gauss-Hermite quadrature is a method of numerically approximating integrals with respect to a Normal distribution. To obtain the likelihood to maximize, we need to integrate the Normally distributed random effects out of the joint likelihood. This integral in LMMs is analytically tractable; the marginal distributions of a multivariate Normal distribution are also multivariate Normal. In GLMMs an analytical solution is not tractable, making the numerical approximation using quadrature necessary.

For our model, we have the conditional likelihood and log-likelihood equations for one subject:

$$L(\boldsymbol{\theta}|y,z) = \binom{n}{y}\left(\frac{1}{1+e^{-(\eta+\sigma z)}}\right)^{y}\left(1-\frac{1}{1+e^{-(\eta+\sigma z)}}\right)^{n-y}\frac{1}{\sqrt{2\pi}}e^{\frac{-z^2}{2}} \tag{3.3.1}$$

$$\ell(\boldsymbol{\theta}|y,z) = \ln\binom{n}{y} + y(\eta+\sigma z) - n(\eta+\sigma z) - n\ln\left(1+e^{-(\eta+\sigma z)}\right) - \ln\sqrt{2\pi} - \frac{z^2}{2} \quad (3.3.2)$$

where $\boldsymbol{\theta}$ is our parameter vector of interest, in this case $(\eta\ \sigma^2)'$.

When we have r subjects, the full log-likelihood is:

$$\ell(\boldsymbol{\theta}|\mathbf{y},\mathbf{z}) = \sum_{i=1}^{r} \ln\binom{n_i}{y_i} + y_i\left(\eta+\sigma z_i\right) - n_i\left(\eta+\sigma z_i\right) - n_i\ln\left(1+e^{-(\eta+\sigma z_i)}\right) - \ln\sqrt{2\pi} - \frac{z_i^2}{2}, \quad (3.3.3)$$

where $\mathbf{y}$ is the vector of responses and $\mathbf{z}$ is the vector of random effects. Define $h(\mathbf{z},\boldsymbol{\theta})$ as the negative log-likelihood.

Ultimately, to solve for the fixed effects we need to integrate out the random effects. To do this, we will use Gauss-Hermite quadrature. That is, we want to solve

$$\int e^{-h(\mathbf{z},\boldsymbol{\theta})} d\mathbf{z}.$$

Though $\mathbf{z}$ is the vector of random effects, we may calculate the required likelihoods by subject and sum over the subjects' likelihoods at the end because the random subject effects are independent. In this light, the derivations will be in scalar notation for simplicity. Define the second-order Taylor series expansion of $h(z,\boldsymbol{\theta})$ with

$$q(z,\boldsymbol{\theta}) = h\left(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta}\right) + \frac{1}{2}h''\left(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta}\right)\left(z-\hat{z}(\boldsymbol{\theta})\right)^2$$

where $\hat{z}(\boldsymbol{\theta})$ maximizes the conditional likelihood at the current estimate of $\boldsymbol{\theta}$, and $h''\left(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta}\right)$ is the second derivative of $h(z,\boldsymbol{\theta})$ with respect to the random effect

evaluated at the current estimate of the random effect, $\hat{z}(\boldsymbol{\theta}) = \hat{\sigma}\hat{z}$. That is,

$$h''(z, \boldsymbol{\theta}) = \frac{1}{\sigma^2} + \frac{ne^{\eta+\sigma z}}{(1+e^{\eta+\sigma z})^2}.$$

Then

$$\begin{aligned}
\int e^{-h(z,\boldsymbol{\theta})}dz &= \int e^{-(h(z,\boldsymbol{\theta})-q(z,\boldsymbol{\theta}))}e^{-q(z,\boldsymbol{\theta})}dz \\
&= e^{-h(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta})}\int e^{-(h(z,\boldsymbol{\theta})-q(z,\boldsymbol{\theta}))}e^{\frac{h''(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta})}{2}(z-\hat{z}(\boldsymbol{\theta}))^2}dz \\
&= \frac{\sqrt{2\pi}e^{-h(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta})}}{\sqrt{h''\left(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta}\right)}}\int e^{-(h(z,\boldsymbol{\theta})-q(z,\boldsymbol{\theta}))}\frac{e^{\frac{h''(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta})(z-\hat{z}(\boldsymbol{\theta}))^2}{2}}}{\sqrt{2\pi}\left(\frac{1}{\sqrt{h''(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta})}}\right)}dz \\
&= \frac{\sqrt{2\pi}e^{-h(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta})}}{\sqrt{h''\left(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta}\right)}}\int e^{-\left(h\left(\frac{z}{\sqrt{h''(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta})}}+\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta}\right)-q\left(\frac{z}{\sqrt{h''(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta})}}+\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta}\right)\right)}\frac{e^{-\frac{z^2}{2}}}{\sqrt{2\pi}}dz.
\end{aligned}$$

Now let (x_i, w_i) be Gauss-Hermite quadrature abscissas and weights. Then

$$\begin{aligned}
\int e^{-h(z,\boldsymbol{\theta})}dz &\cong \frac{\sqrt{2\pi}e^{-h(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta})}}{\sqrt{h''\left(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta}\right)}}\sum_i e^{-\left(h\left(\frac{x_i}{\sqrt{h''(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta})}}+\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta}\right)-q\left(\frac{x_i}{\sqrt{h''(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta})}}+\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta}\right)\right)}w_i \\
&= \frac{\sqrt{2\pi}e^{-h(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta})}}{\sqrt{h''\left(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta}\right)}}\sum_i e^{-h\left(\frac{x_i}{\sqrt{h''(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta})}}+\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta}\right)}e^{h(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta})}e^{\frac{x_i^2}{2}}w_i. \qquad (3.3.4)
\end{aligned}$$

With these formulas established, we are ready to begin building the program. To begin the process, we require starting values for the intercept and variance. The simplest starting value for the intercept uses the usual, naive estimator for p, $\hat{p} = \sum_i \frac{y_i}{n_i}$. So the initial estimate is $\hat{\eta} = \text{logit}(\hat{p})$. For the variance, there are many options for obtaining an initial estimate, such as MIVQUE0 as implemented in PROC GLIMMIX (Goodnight 1978) or Henderson's ANOVA estimators. We will use the result of a single REML iteration so that the starting value is near the parameter. With these parameter estimates, we obtain initial estimates of the

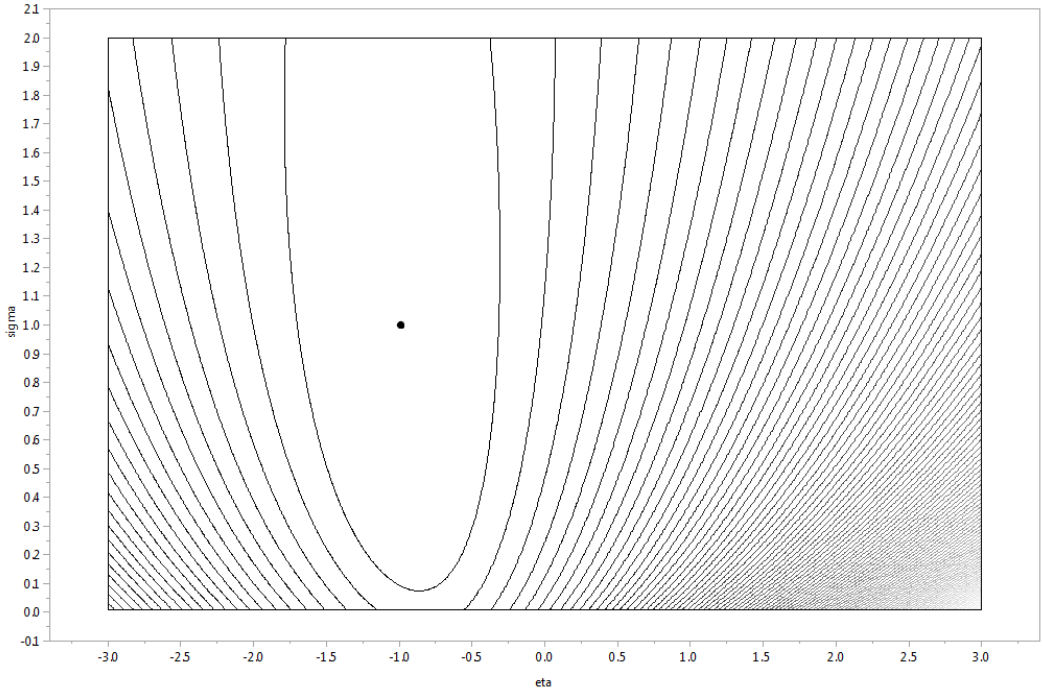
random effects (i.e. the BLUPs).

In the R software package, the `optim()` function minimizes a function over a specified set of parameters. Given our initial estimates of η and σ^2, we wish to minimize $h(\mathbf{z}, \boldsymbol{\theta})$ over the z_i random effects and return the BLUPs. To use `optim()`, we first write an R function that, given the z_i, η, σ^2 and the data, calculates $h(\mathbf{z}, \boldsymbol{\theta})$. This is now the function argument for `optim()`. We provide the starting values of η and σ^2 found before, call `optim()` and the BLUPs are returned. Now with initial estimates of all of the fixed and random effects, we proceed to the next step.

The objective function in maximum likelihood estimation is the -2 log-likelihood. This is the log-likelihood of the data with the random effects integrated out either through the Laplace approximation or Gauss-Hermite quadrature. We choose the number of quadrature points. With our initial estimates of all fixed and random effects, we calculate the -2 log-likelihood using formulas 3.3.3 and 3.3.4. This objective function is used to calculate both the gradients and the Hessian matrix required to solve for the fixed effects. Figure 3.3.1 shows a contour plot of a representative -2 log-likelihood surface for this model with $\eta = -1$ and $\sigma^2 = 1$. The point of true parameter values is indicated toward the center of the ridge. The objective function values increase as values of η or σ^2 move away from the true values.

The initial parameter estimates are unlikely to minimize the objective function. If they did, the gradients, the derivatives with respect to the parameters of the objective function, evaluated at the estimates would be equal to zero (or nearly so). If they do not equal zero, we use the second derivative matrix, the Hessian, to determine the step size for updating the estimates in order to decrease the objective

Figure 3.3.1: Contour Plot of -2 Log-Likelihood



function and move closer to the minimum. Analytic calculation of these first and second derivatives can be highly complex depending on the model. Even in our simple random intercept-only model, the calculation is not completely straightforward. Therefore we use the numerical derivative process to calculate the gradients and Hessian. With functions calculating the numerical gradients and Hessian, all of the pieces required to build a Newton-Raphson or Broyden method solver are in place.

Newton-Raphson and Broyden are methods for solving nonlinear systems of equations that work using the same underlying iteration principle. The Broyden method uses an approximate Hessian matrix in order to save the computation of the second derivatives. Both methods start with initial estimates we believe to be in the neighborhood of the solution, and check whether the gradients are zero, indicating a

minimum. If the gradients are not zero, step in the direction indicated by the gradient with a step-size calculated using the Hessian (or approximate Hessian in the case of Broyden) to update the estimates, then recalculate the gradients. The iteration process continues until either a solution is converged upon or a maximum number of iterations passes. Researchers know that sometimes the "step" to update the estimates overshoots the solution (Press et al., 1992). We verify that the step truly reduces the value of the objective function. If the step does not reduce the value, we include a line search function to "move back" along the direction of the step. This ensures that we move in the correct direction while improving our position. The line search function is called in the middle of the solve routine to halve the step size until the objective function has decreased.

This foundation program produces the MLEs equivalent to PROC GLIMMIX and `lmer`. We may now consider our adjustment to the estimation process.

3.4 Firth for GLMMs

Recall from chapter 2.3 that Firth proposed two variants of his adjustment to the score equation for calculating MLEs - the expected and observed Firth adjustment. Even the observed Firth adjustment, equation 2.3.5, contains a computationally intense expectation. We approached development of the adjustment in two stages. First, we reduced the computational burden by deriving a "doubly observed" adjustment to use in a proof of concept. Then we implemented the full expected Firth adjustment.

The "doubly observed" alternative reduces the expectation in the adjustment formula to a sum of observed quantities already calculated. Starting from equation

2.3.5

$$\mathbf{A}_{\theta j} = -\frac{1}{2}\,\mathrm{tr}\left(\mathbf{H}^{-1}\mathrm{E}\left[S_{\theta j}(\mathbf{H} - \mathbf{S}\mathbf{S}^T)\right]\right)$$

where $\mathbf{H}$ is the Hessian matrix, $\mathbf{S}$ is the score vector and S_{θ_j} is the element of the score vector corresponding to the parameter θ_j. Both the Hessian and score vector depend upon the subject data, y_i, which are assumed independent. To make this explicit, write

$$\mathbf{S} = \sum_{i=1}^{r} \mathbf{S}\left(y_i\right) \text{ and } \mathbf{H} = \sum_{i=1}^{r} \mathbf{H}\left(y_i\right).$$

Consider the expectation in the definition of the observed Firth adjustment,

$$\mathrm{E}_y\left[\sum_{i=1}^{r} \mathbf{S}_{\theta_j}\left(y_i\right)\left(\sum_{k=1}^{r} \mathbf{H}\left(y_k\right) - \sum_{k=1}^{r} \mathbf{S}\left(y_k\right)\sum_{l=1}^{r} \mathbf{S}^T\left(y_l\right)\right)\right].$$

Taking the first two summations outside the expectation gives

$$\sum_{i=1}^{r}\sum_{k=1}^{r} \mathrm{E}_y\left[\mathbf{S}_{\theta_{\mathrm{j}}}\left(y_i\right)\left(\mathbf{H}\left(y_k\right) - \mathbf{S}\left(y_k\right)\sum_{l=1}^{\mathrm{r}} \mathbf{S}^T\left(y_l\right)\right)\right].$$

Because the y_i are independent and because $\mathrm{E}_y\left[\mathbf{S}_{\theta_j}\left(y\right)\right] = 0$, all expectations in the expanded sum are zero when i, k and l are not all equal to one another. As a result, we may rewrite the expectation as

$$E_y\left[\sum_{i=1}^{r} \mathbf{S}_{\theta_j}\left(y_i\right)\left(\mathbf{H}\left(y_i\right) - \mathbf{S}\left(y_i\right)\mathbf{S}^T\left(y_i\right)\right)\right].$$

The "doubly observed" Firth adjustment skips the expectation as a matter of computational convenience. This is similar to using the Hessian (the observed Fisher Information matrix) in place of its expectation, the expected Fisher Information, when calculating the variance matrix of the MLE. So we have

$$\mathbf{A}_{\theta j} = -\frac{1}{2}\,\mathrm{tr}\left(\mathbf{H}^{-1}\sum_{i=1}^{r}\mathbf{S}_{\theta_j}\left(y_i\right)\left(\mathbf{H}\left(y_i\right) - \mathbf{S}\left(y_i\right)\mathbf{S}^T\left(y_i\right)\right)\right). \tag{3.4.1}$$

Once initial simulations using this "doubly observed" adjustment yielded promising results, we progressed to the full expected Firth adjustment.

Using this adjustment, the modified score equation was added to the existing Broyden solver routine to obtain Firth-adjusted MLEs. Because we now solve for adjusted MLEs, we may also use the MLEs from PROC GLIMMIX as the starting values for the Broyden routine. This provides a better starting place to allow for faster iteration to the final adjusted MLEs. While the numeric derivatives for the gradient and Hessian were sufficient for the foundation program finding the MLE, the approximation of so many pieces of the adjustment resulted in too many convergence failures. Therefore, we eliminated the numeric derivatives and derived the analytic gradient and Hessian.

3.4.1 Analytic Gradient

Recall the general expression for the marginal likelihood as computed using Gaussian quadrature in equation 3.3.4 on page 28. In the following derivations, $\hat{h}$ denotes h evaluated at the current estimate of the random effects, $\hat{z}(\boldsymbol{\theta})$. A prime indicates the derivative with respect to the random effects. To simplify the derivation of the analytic gradient and Hessian, we will compute them again for each independent subject individually and sum at the end. Rewrite equation 3.3.4 as

$$L(\boldsymbol{\theta}|y) = (2\pi)^{1/2}\sum_i w_i e^{-D_i(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta})}e^{\frac{x_i^2}{2}}$$

where

$$D_i\left(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta}\right)=h\left(\hat{z}(\boldsymbol{\theta})+\frac{x_i}{\sqrt{h''\left(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta}\right)}},\boldsymbol{\theta}\right)+\frac{1}{2}\ln h''\left(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta}\right).$$

Then

$$\frac{\partial L}{\partial\boldsymbol{\theta}}=-(2\pi)^{1/2}\sum_i w_i\left(\frac{\partial}{\partial\boldsymbol{\theta}}D_i\left(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta}\right)\right)e^{-D_i\left(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta}\right)}e^{\frac{x_i^2}{2}}.$$

Let

$$\hat{z}_{\boldsymbol{\theta}}=\frac{\partial}{\partial\boldsymbol{\theta}}\hat{z}(\boldsymbol{\theta})=-\frac{h'_{\boldsymbol{\theta}}\left(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta}\right)}{h''\left(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta}\right)}.$$

The subscript $\boldsymbol{\theta}$ indicates a derivative with respect to the parameters. Note that

$$\begin{aligned}\frac{\partial}{\partial\boldsymbol{\theta}}h''\left(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta}\right) &= \hat{z}_{\boldsymbol{\theta}}h'''\left(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta}\right)+h''_{\boldsymbol{\theta}}\left(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta}\right)\\ &= \hat{z}_{\boldsymbol{\theta}}\hat{h}'''+\hat{h}''_{\boldsymbol{\theta}}\end{aligned}$$

and

$$\frac{\partial}{\partial\boldsymbol{\theta}}\left(\hat{z}(\boldsymbol{\theta})+\frac{x_i}{\sqrt{h''\left(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta}\right)}}\right)=\hat{z}_{\boldsymbol{\theta}}-\frac{1}{2}\left(\hat{h}''\right)^{-\frac{3}{2}}\left(\hat{z}_{\boldsymbol{\theta}}\hat{h}'''+\hat{h}''_{\boldsymbol{\theta}}\right)x_i.$$

So,

$$\begin{aligned}\frac{\partial}{\partial\boldsymbol{\theta}}h\left(\hat{z}(\boldsymbol{\theta})+\frac{x_i}{\sqrt{h''(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta})}}\right) &= \left(\hat{z}_{\boldsymbol{\theta}}-\frac{1}{2}\left(\hat{h}''\right)^{-\frac{3}{2}}\left(\hat{z}_{\boldsymbol{\theta}}\hat{h}'''+\hat{h}''_{\boldsymbol{\theta}}\right)x_i\right)h'\left(\hat{z}(\boldsymbol{\theta})+\frac{x_i}{\sqrt{\hat{h}''}}\right)\\ &\quad +h_{\boldsymbol{\theta}}\left(\hat{z}(\boldsymbol{\theta})+\frac{x_i}{\sqrt{\hat{h}''}}\right)\end{aligned}$$

and

$$\frac{\partial}{\partial\boldsymbol{\theta}}D_i\left(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta}\right)=\frac{\partial}{\partial\boldsymbol{\theta}}h\left(\hat{z}(\boldsymbol{\theta})+\frac{x_i}{\sqrt{\hat{h}''}},\boldsymbol{\theta}\right)+\frac{1}{2}\frac{\frac{\partial}{\partial\boldsymbol{\theta}}h''\left(\hat{z}(\boldsymbol{\theta}),\boldsymbol{\theta}\right)}{\hat{h}''}.$$

This provides the derivatives necessary to calculate the gradient of the *likelihood.* It follows that the gradient of the *log-likelihood,* the term needed for the adjustment, is $\frac{\frac{\partial L}{\partial\boldsymbol{\theta}}}{L(\boldsymbol{\theta}|y)}$.

Table 3.4.1: Derivatives in the analytic gradient

$h'(\boldsymbol{\theta}\|y,b)$	$-y+n+\frac{b}{\sigma^2}-\frac{n}{1+\exp(\eta+b)}$
$h''(\boldsymbol{\theta}\|y,b)$	$\frac{1}{\sigma^2}+\frac{n\exp(\eta+b)}{(1+\exp(\eta+b))^2}$
$h'''(\boldsymbol{\theta}\|y,b)$	$\frac{n\exp(\eta+b)(1-\exp(\eta+b))}{(1+\exp(\eta+b))^3}$
$h_{\boldsymbol{\theta}}(\boldsymbol{\theta}\|y,b)$	$\left(-y+n-\frac{n}{1+\exp(\eta+b)},\ \frac{\sigma^2+b^2}{2\sigma^4}\right)$
$h'_{\boldsymbol{\theta}}(\boldsymbol{\theta}\|y,b)$	$\left(\frac{n\exp(\eta+b)}{(1+\exp(\eta+b))^2},\ \frac{-b}{\sigma^4}\right)$
$h''_{\boldsymbol{\theta}}(\boldsymbol{\theta}\|y,b)$	$\left(\frac{n\exp(\eta+b)(1-\exp(\eta+b))}{(1+\exp(\eta+b))^3},\ \frac{-1}{\sigma^4}\right)$

For the random intercept logit model, Table 3.4.1 provides the derivatives in these formulas. We return to the original random effect parameterization as in formula 3.2.1, $\text{logit}(p)=\eta+b$, for these derivatives as they are more straightforward. Therefore the negative log-likelihood is

$$h(\boldsymbol{\theta}|y,b)=-\ln\binom{n}{y}-y(\eta+b)+n(\eta+b)+n\ln\left(1+e^{-(\eta+b)}\right)+\ln\sqrt{2\pi}+\ln(\sigma)+\frac{b^2}{2\sigma^2}.$$

3.4.2 Analytic Hessian

The Hessian matrix of interest for this model is the 2×2 matrix of second derivatives defined by

$$-\frac{\partial^2}{\partial\theta_j\partial\theta_k}\ln L(\boldsymbol{\theta}|y)=-\frac{\partial}{\partial\theta_k}\frac{\frac{\partial L}{\partial\theta_j}}{L(\boldsymbol{\theta}|y)}=\frac{\frac{\partial L}{\partial\theta_j}\frac{\partial L}{\partial\theta_k}}{L(\boldsymbol{\theta}|y)^2}-\frac{\frac{\partial^2 L(\boldsymbol{\theta}|y)}{\partial\theta_j\partial\theta_k}}{L(\boldsymbol{\theta}|y)}. \tag{3.4.2}$$

Continuing from the gradient derivations, we have

$$\frac{\partial^2 L(\boldsymbol{\theta}|y)}{\partial\theta_j\partial\theta_k}=(2\pi)^{1/2}\sum_i w_i\left(\frac{\partial D_i}{\partial\theta_j}\frac{\partial D_i}{\partial\theta_k}-\frac{\partial^2 D_i}{\partial\theta_j\partial\theta_k}\right)e^{-D_i}e^{\frac{x_i^2}{2}}$$

and

$$\frac{\partial^2 D_i}{\partial\theta_j\partial\theta_k} = \frac{\partial^2}{\partial\theta_j\partial\theta_k} h\left(\hat{z}(\boldsymbol{\theta}) + \frac{x_i}{\sqrt{\hat{h}''}}\right) + \frac{1}{2}\frac{\frac{\partial^2}{\partial\theta_j\partial\theta_k}\hat{h}''}{\hat{h}''} - \frac{1}{2}\frac{\frac{\partial}{\partial\theta_j}\hat{h}''\frac{\partial}{\partial\theta_k}\hat{h}''}{\left(\hat{h}''\right)^2}.$$

These require

$$\hat{z}_{\theta_j\theta_k} = -\frac{\hat{z}_{\theta_j}\hat{h}''_{\theta_k} + \hat{h}'_{\theta_j\theta_k}}{\hat{h}''} + \frac{\hat{h}'_{\theta_k}\left(\hat{z}_{\theta_j}\hat{h}''' + \hat{h}''_{\theta_j}\right)}{\left(\hat{h}''\right)^2},$$

$$\begin{aligned}\frac{\partial^2}{\partial\theta_j\partial\theta_k}\left(\hat{z}(\boldsymbol{\theta}) + \frac{x_i}{\sqrt{\hat{h}''}}\right) &= \hat{z}_{\theta_j\theta_k} + \frac{3x_i}{2}\frac{\left(\hat{z}_{\theta_j}\hat{h}''' + \hat{h}''_{\theta_j}\right)\left(\hat{z}_{\theta_k}\hat{h}''' + \hat{h}''_{\theta_k}\right)}{\left(\hat{h}''\right)^{\frac{5}{2}}} \\ &\quad -\frac{x_i}{2}\frac{\hat{z}_{\theta_j\theta_k}\hat{h}''' + \hat{z}_{\theta_j}\hat{z}_{\theta_k}\hat{h}^{iv} + \hat{z}_{\theta_j}\hat{h}'''_{\theta_k} + \hat{z}_{\theta_k}\hat{h}'''_{\theta_j} + \hat{h}''_{\theta_j\theta_k}}{\left(\hat{h}''\right)^{\frac{3}{2}}},\end{aligned}$$

and

$$\begin{aligned}\frac{\partial^2}{\partial\theta_j\partial\theta_k} h\left(\hat{z}(\boldsymbol{\theta}) + \frac{x_i}{\sqrt{\hat{h}''}}\right) &= \frac{\partial^2}{\partial\theta_j\partial\theta_k}\left(\hat{z} + \frac{x_i}{\sqrt{\hat{h}''}}\right) h'\left(\hat{z} + \frac{x_i}{\sqrt{\hat{h}''}}\right) \\ &\quad + \left(\frac{\partial}{\partial\theta_j}\hat{z} + \frac{x_i}{\sqrt{\hat{h}''}}\right)\left(\frac{\partial}{\partial\theta_k}\hat{z} + \frac{x_i}{\sqrt{\hat{h}''}}\right)\hat{h}''\left(\hat{z} + \frac{x_i}{\sqrt{\hat{h}''}}\right) \\ &\quad + \left(\frac{\partial}{\partial\theta_j}\hat{z} + \frac{x_i}{\sqrt{\hat{h}''}}\right) h'_{\theta_k}\left(\hat{z} + \frac{x_i}{\sqrt{\hat{h}''}}\right) \\ &\quad + \left(\frac{\partial}{\partial\theta_k}\hat{z} + \frac{x_i}{\sqrt{\hat{h}''}}\right) h'_{\theta_j}\left(\hat{z} + \frac{x_i}{\sqrt{\hat{h}''}}\right) \\ &\quad + h_{\theta_j\theta_k}\left(\hat{z} + \frac{x_i}{\sqrt{\hat{h}''}}\right)\end{aligned}$$

with $\hat{z} = \hat{z}(\boldsymbol{\theta})$ where the "of theta" is implicit. Finally,

$$\begin{aligned}\frac{\partial^2}{\partial\theta_j\theta_k} h''\left(\hat{z}(\boldsymbol{\theta}), \boldsymbol{\theta}\right) &= \hat{z}_{\theta_j\theta_k}\hat{h}''' + \hat{z}_{\theta_j}\hat{z}_{\theta_k}\hat{h}^{iv} + \hat{z}_{\theta_j}\hat{h}'''_{\theta_k} + \hat{z}_{\theta_k}\hat{h}'''_{\theta_j} \\ &\quad + \hat{h}''_{\theta_j\theta_k}.\end{aligned}$$

Table 3.4.2: Derivatives in the analytic Hessian

$h'''(\boldsymbol{\theta}\|y,b)$	$\frac{n\exp(\eta+b)(1-4\exp(\eta+b)+\exp(2(\eta+b))}{(1+\exp(\eta+b))^4}$
$h'''_{\boldsymbol{\theta}}(\boldsymbol{\theta}\|y,b)$	$\left(\frac{n\exp(\eta+b)(1-4\exp(\eta+b)+\exp(2(\eta+b)))}{(1+\exp(\eta+b))^4},\ 0\right)$
$h_{\theta_j\theta_k}(\boldsymbol{\theta}\|y,b)$	$\begin{pmatrix} \frac{n\exp(\eta+b)}{(1+\exp(\eta+b))^2} & 0 \\ 0 & \frac{2b^2-\sigma^2}{2\sigma^6} \end{pmatrix}$
$h'_{\theta_j\theta_k}(\boldsymbol{\theta}\|y,b)$	$\begin{pmatrix} \frac{n\exp(\eta+b)(1-\exp(\eta+b))}{(1+\exp(\eta+b))^3} & 0 \\ 0 & \frac{2b}{\sigma^6} \end{pmatrix}$
$h''_{\theta_j\theta_k}(\boldsymbol{\theta}\|y,b)$	$\begin{pmatrix} \frac{n\exp(\eta+b)(1-4\exp(\eta+b)+\exp(2(\eta+b)))}{(1+\exp(\eta+b))^4} & 0 \\ 0 & \frac{2}{\sigma^6} \end{pmatrix}$

The additional derivatives required for the Hessian matrix are provided in Table 3.4.2.

3.4.3 Observed and Expected Firth

With the approximate "doubly observed" Firth adjustment showing promise, the remaining step is to calculate the expectations required in the observed and expected Firth adjustments. We utilize some shortcuts to reduce computation in these expectations. First, note that in these initial simulations, the number of trials per subject is equal. Therefore, we may calculate the expectation for one subject then multiply by the number of subjects for the full expectation. Second, we attempt to reduce the number of calculations of the likelihood by recognizing that a large percentage of the expectations come from values around the mode of the distribution.

We know that the expected value of the Hessian or the expected value of the second quantity in the Firth adjustment is $\mathrm{E}[g(y|\boldsymbol{\theta},z)] = \sum_{y=0}^{n} g(y|\boldsymbol{\theta},z)f(y|\boldsymbol{\theta},z)$. For binomial data, the mode of the distribution f will be near $y = \lfloor np \rfloor$ where p is the current estimate of the probability. When we calculate the expectations, we start

from this y and increment through the sum in both directions. When decreasing to $y = 0$ or increasing to $y = n$, we check to see if the relative change in the sum has decreased below some tolerance. If the relative change has decreased prior to reaching the sum's y bound, then we will truncate the sum at that point. Any further calculations will not add any significant amount. With small n this check likely will not decrease computation time, but with large n it could eliminate some calculations.

3.5 Summary

The issue of bias in the variance component estimation for the random intercept logit model is evident. There are combinations of random subject and number of observations per subject where quadrature is the preferred method of estimation. However, the bias of the MLE produced by quadrature makes the MLE a less appealing choice. The Firth adjustment to the MLE estimation procedure provides an opportunity to correct this bias. In the next chapter, we investigate whether the Firth adjustment to the MLE for the random intercept logit model performs as desired.

Chapter 4

Bias Simulation Study

4.1 Introduction

Variance component estimation in generalized linear mixed models has not received the same attention as for Gaussian linear mixed models. In LMMs, maximum likelihood estimators of variance components are known to be biased downward. This MLE bias induces Type I error inflation and inadequate confidence interval coverage. REML largely addresses the bias problem in LMMs. There is no REML for GLMMs. However, the Firth adjustment to the MLE, which is equivalent to REML for some LMMs, may serve as a REML analog for GLMMs. The downward bias of the variance component MLE holds for GLMMs as well as LMMs, but they are less well understood or appreciated. The derivations necessary to implement the adjustment for the random intercept logit model were completed in chapter 3.

In this chapter, we discuss simulations conducted to determine the behavior of the Firth-adjusted MLE with balanced data. These begin with the "doubly observed" adjustment that was developed to shortcut the expectations included in the observed and expected Firth adjustments. As a means to prove the concept, when the

"doubly observed" adjustment showed promise, we added the computational burden of the expectations in the other variations of the adjustment. The simulations with the observed and expected Firth adjustments show the reduction of the bias of the variance estimate as compared to the unadjusted MLE. Therefore, we conclude with a simulation using the expected Firth adjustment for various unbalanced data cases.

4.2 Firth Adjustment with Balanced Data

1,000 data sets were simulated using the random intercept logit model:

$$\text{logit}\left(p_i\right) = \eta + b_i$$

with ten subjects, 100 independent Bernoulli trials per subject and a subject variance of one. We analyzed these data sets using the quadrature option in SAS PROC GLIMMIX. The method was set at seventeen point quadrature to obtain the unadjusted MLE. The respective Firth adjustment procedure also with seventeen point quadrature was used to obtain the Firth-adjusted MLE. There were minor convergence issues using the Firth procedures. Convergence criteria were set for relative changes less than 10^{-6}. Causes of non-convergence are listed below in order of frequency.

1. An initial variance MLE starting value of 0.
2. A singular Hessian matrix during the estimation process.
3. An invalid argument to the exponential function, due to an estimated random effect being too large. The software is limited to less than e^{1000}.

4. Infeasible parameters when obtaining current random effect estimates using PROC GLIMMIX.

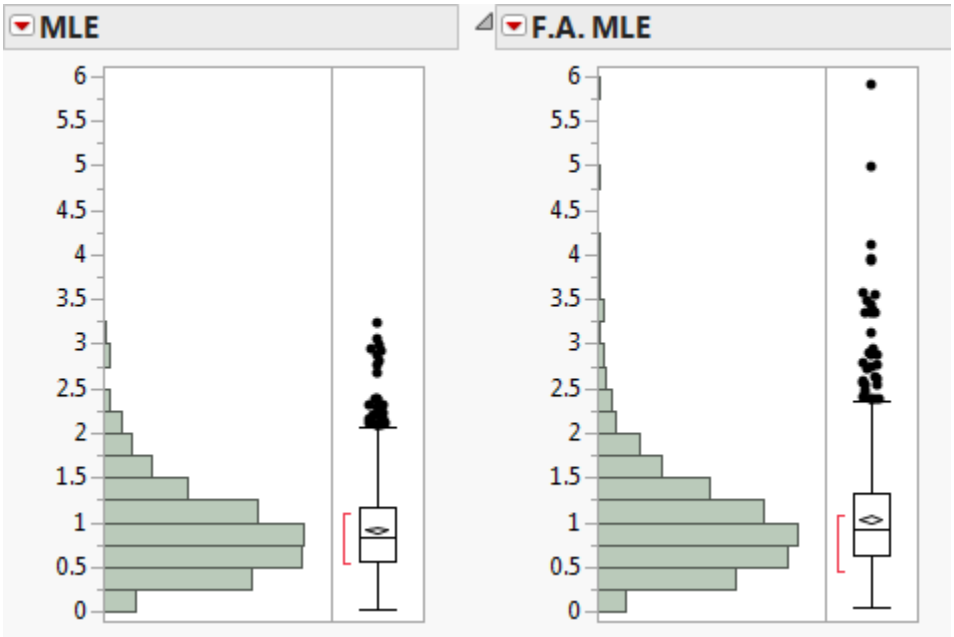
5. Exceeding 100 iterations.

In the next sections, these convergence issues limited some studies to fewer than 1,000 experiments as noted. However, the non-convergence rate did not exceed 1% in the preliminary simulations, and under the expected Firth adjustment there were no convergence failures.

4.2.1 "Doubly Observed" Firth Adjustment with Analytic Gradient

Figure 4.2.1 shows the distribution of the estimated variances for 998 of 1,000 simulated experiments. As expected the "doubly observed" Firth adjustment resulted in an improvement in the bias coupled with much greater variability in the estimates compared to the MLE due to the observed Hessian and approximation to the second expectation in the Firth adjustment used. This is borne out in the MLEs having a mean of 0.92 with standard deviation of 0.492 and the Firth-adjusted MLEs having a mean of 1.06 with standard deviation 0.693. Note, however, the extremely right skewed distribution of the estimates; a more accurate measure of center would be the median. The median MLE is 0.85 compared to the median Firth-adjusted MLE of 0.93. Because the "doubly observed" adjustment behaves as expected, we progressed to the observed and expected adjustments. These should improve the results, i.e. reduce the variability of the estimate.

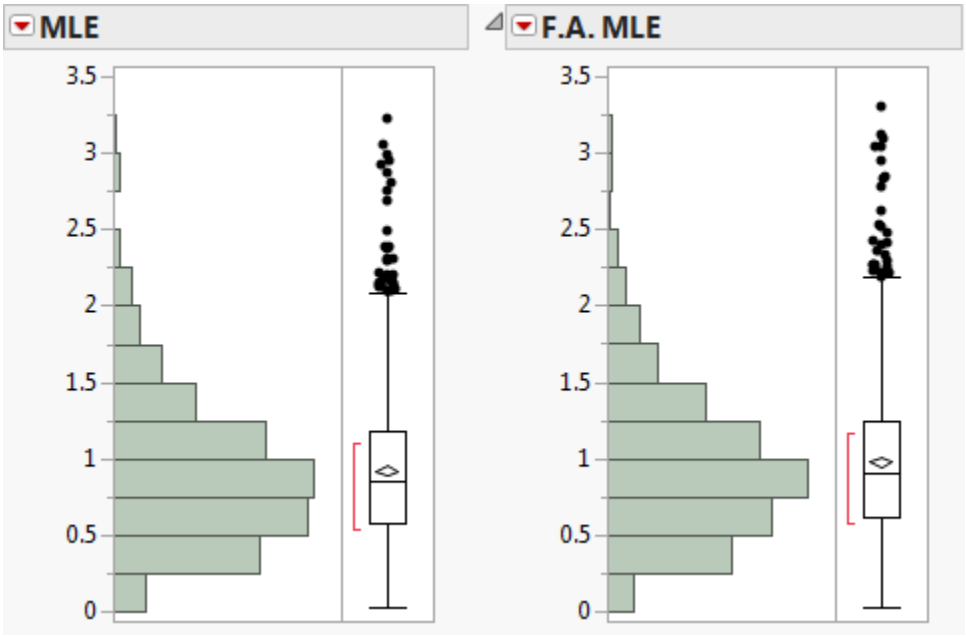
Figure 4.2.1: Distribution of MLE and D.O.F.A. MLE from 998 Experiments



4.2.2 Observed Firth Adjustment

Figure 4.2.2 shows the distribution of the estimated variances for 999 of 1,000 simulated experiments. The observed Firth-adjusted MLE mean was 0.98 with standard deviation 0.511. The mean and standard deviation for the MLE are the same as in the "doubly observed" simulation. We can see that the distributions of the two estimators are much more similar than when using the "doubly observed" adjustment. Because of this, we may compare these means instead of the outlier-robust medians. By taking the expectation instead of approximating it with the sum, the extreme estimates are eliminated lowering the mean and reducing the variability. We do not consider the observed Firth adjustment further as the computational effort is not significantly different between it and the expected Firth adjustment.

Figure 4.2.2: Distribution of MLE and O.F.A MLE from 999 Experiments

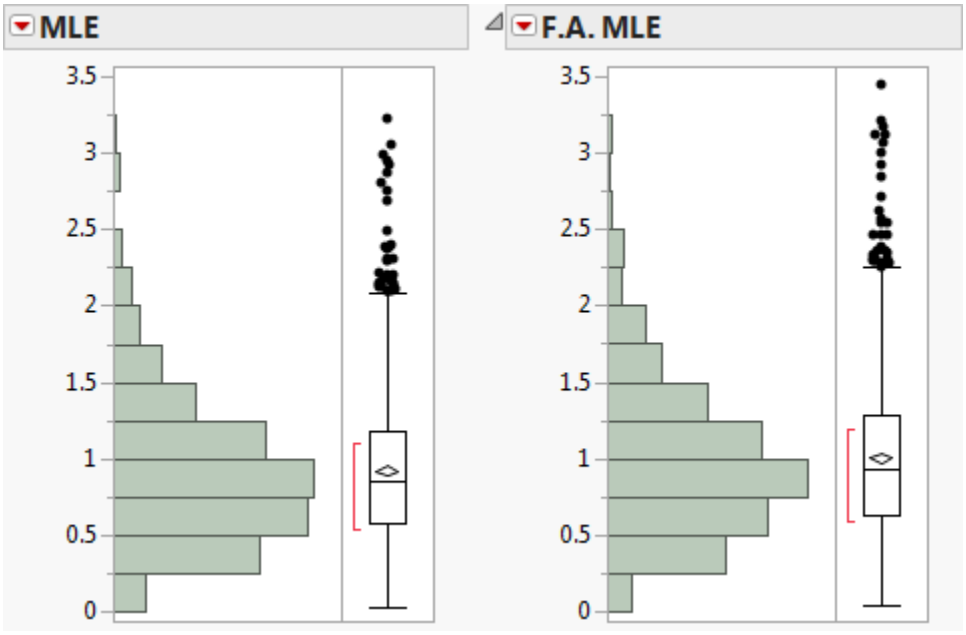


4.2.3 Expected Firth Adjustment

Figure 4.2.3 shows the distribution of the estimated variances for 1,000 simulated experiments. This adjustment results in a mean Firth MLE of 1.00, the true value used in the simulation, with a standard deviation of 0.526. These results again compare favorably to the MLE mean of 0.92 with standard deviation 0.493.

With these promising results, a full simulation study using the same combinations of subjects and observations per subject as the initial examination was conducted using the expected Firth adjustment. Figure 4.2.4 illustrates the results of this simulation study with Table 4.2.1 showing the data used in the figure. The mean estimate using quadrature for ten subjects and two observations per subject was omitted for clarity of the graph. That estimate was 2.67. The Firth estimates yield lower bias than the MLE for all scenarios.

Figure 4.2.3: Distribution of MLE and E.F.A MLE from 1,000 Experiments



4.3 Unequal sample size

While the adjustment appears to be the method of choice in the balanced data scenarios, real-world situations often produce unbalanced data. Therefore investigation of unbalanced or missing data scenarios is necessary. Using the framework of the balanced case simulations, we focused on cases representative of experiments likely to be used. We looked at scenarios where there are 10, 20 or 50 random subjects and 10 or 30 planned observations per subject. Then we designated some percentage of the random subjects to have missing observations. We looked at 20% missing, defined here as "minor missingness," and 50% missing. Then if the random subject had missing values, it had either 20% or 50% missing. This means

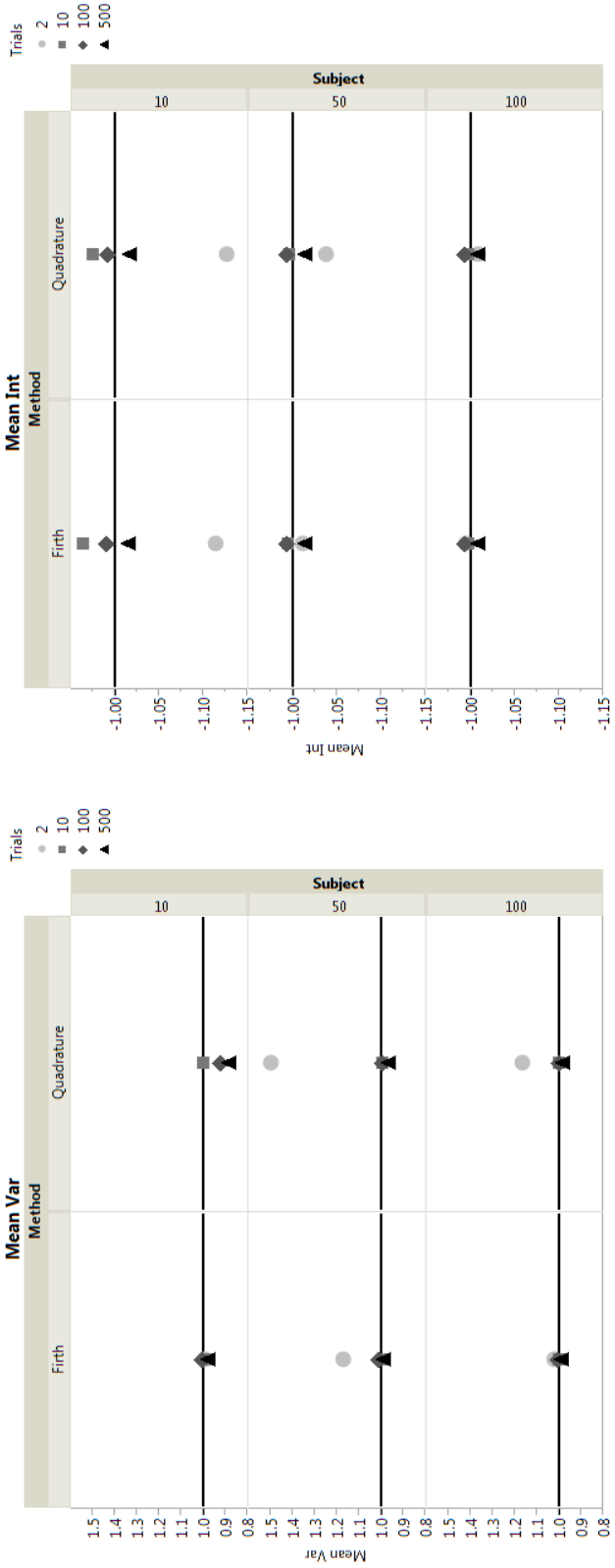


Figure 4.2.4: MLE vs. Expected Firth-adjusted MLE

(a) Mean Variance Estimates

(b) Mean Intercept Estimates

Table 4.2.1: Simulation Results using Firth for $b \sim N(0,1)$

		Firth				Quadrature				
Subjects	Trials	mean σ^2 est.	med. σ^2 est.	mean η est.	med. η est.	mean σ^2 est.	med. σ^2 est.	mean η est.	med. η est.	conv. rate
10	2	0.992	0.461	-1.11	-0.90	2.679	1.507	-1.13	-0.95	539
	10	0.990	0.805	-0.97	-0.98	0.998	0.785	-0.97	-0.99	987
	100	1.004	0.926	-0.99	-0.98	0.925	0.849	-0.99	-0.98	1000
	500	0.977	0.892	-1.02	-1.01	0.886	0.807	-1.02	-1.01	1000
50	2	1.17	0.764	-1.01	-0.98	1.501	1.002	-1.04	-1.00	898
	10	0.994	0.936	-0.99	-0.99	0.992	0.933	-1.00	-1.00	1000
	100	1.012	0.994	-0.99	-1.00	0.995	0.977	-0.99	-1.00	1000
	500	0.987	0.971	-1.01	-1.01	0.969	0.953	-1.01	-1.01	1000
100	2	1.024	0.771	-0.99	-0.98	1.165	0.890	-1.01	-0.99	991
	10	0.999	0.970	-0.99	-1.00	0.998	0.969	-1.00	-1.00	1000
	100	1.006	1.000	-0.99	-0.99	0.998	0.992	-0.99	-0.99	1000
	500	0.992	0.983	-1.01	-1.01	0.982	0.973	-1.01	-1.01	1000

Table 4.3.1: Overall percentage of missing data

		Subjects with missing data	
		20%	50%
Observations missing per subject	20%	4%	10%
	50%	10%	25%

total amounts of missing data are 4%, 10% and 25% of the planned number of observations. The levels of missing data are summarized in table 4.3.1. Combining the levels of missing data with the subject and trials per subject combinations, we have 24 total combinations of experiment sizes and levels of missing data.

A small modification must be made to the function used to calculate the required expectations. The original function utilized the fact that all subjects had an equal number of trials by calculating the expected values with respect to one subject and then multiplying by the number of subjects. Now with unequal trial numbers, this shortcut cannot be used. However, we still do not have to calculate the expectations for each subject (a very time consuming process) because some subjects have equal numbers of trials. If we tabulate the unique numbers of trials and how many subjects have them, we only need to go through the expectation calculation however many unique numbers of trials there are.

Table 4.3.2 on page 50 summarizes the results of these simulations. In nearly all cases, the Firth-adjusted MLE for the variance has a mean estimate closer to the true value of 1. This is most dramatically obvious when the number of subjects and observations per subject are smallest and there is a high level of missing data. The discrepancy is not as pronounced when there are more subjects, but this was also true with balanced data. The Firth-adjusted MLE helps when it is needed and does not hurt when it is less necessary.

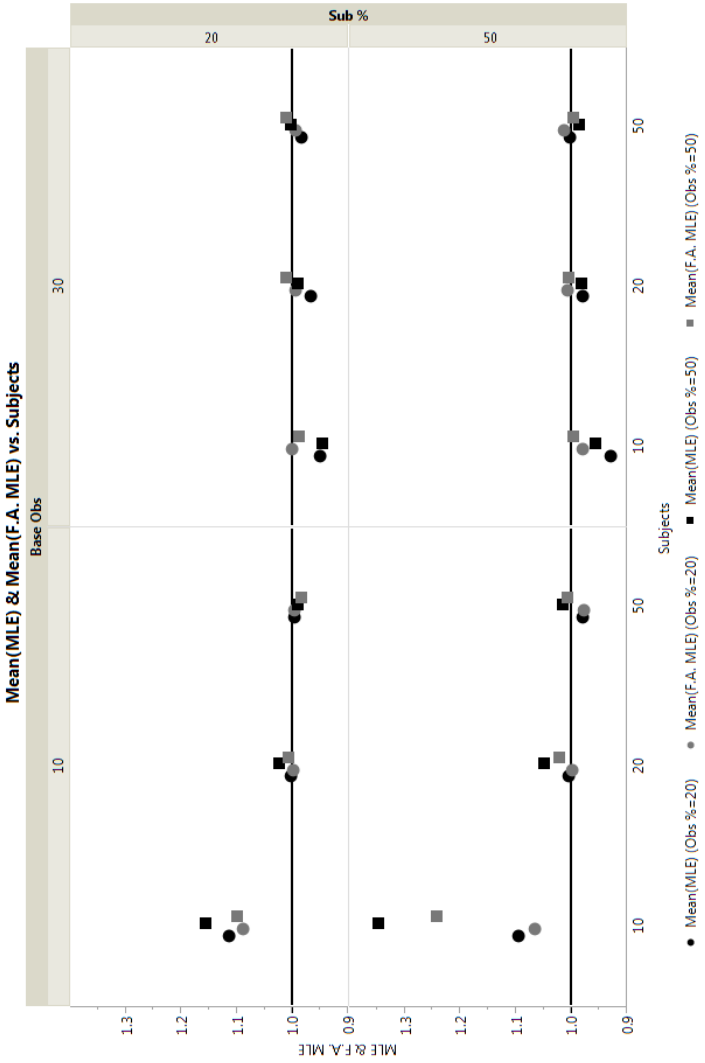


Figure 4.3.1: Unbalanced data MLE vs. F.A. MLE

4.4 Conclusions

Because variance estimate bias affects nearly all frequentist inference, we need to know first how the proposed Firth estimator performs in this regard. If the experiment is perfectly balanced with all subjects having the same number of observations, the expected Firth-adjusted MLE results in mean estimates closer to the true value than the unadjusted MLE. In experiments with various amounts of missing data, the Firth estimator more accurately estimates the variance than the MLE when the number of subjects is small. As the number of subjects increases, the discrepancy between the two estimates decreases. Therefore, the adjusted MLE is recommended for all cases; it decreases the bias in situations where the MLE bias is more pronounced and does not hurt otherwise. In the next chapter we will see whether this apparent improvement in the bias impacts some basic inference about the fixed effects.

Table 4.3.2: Unbalanced data MLE vs. F.A. MLE

				Firth				Quadrature				
Subjects	Base number of trials	Subject missing %	% missing obs	mean σ^2 est.	med. σ^2 est.	mean η est.	med. η est.	mean σ^2 est.	med. σ^2 est.	mean η est.	med. η est.	conv. rate
10	10	20	20	1.088	0.851	-0.99	-1.00	1.115	0.844	-1.01	-1.01	937
			50	1.010	0.824	-1.01	-0.98	1.157	0.834	-1.02	-1.00	915
		50	20	1.065	0.843	-1.01	-0.97	1.096	0.837	-1.02	-0.98	913
			50	1.242	0.948	-1.03	-0.99	1.346	0.977	-1.05	-1.01	873
	30	20	20	1.002	0.869	-1.00	-1.00	0.950	0.812	-1.00	-1.00	996
			50	0.988	0.870	-1.00	-1.00	0.947	0.826	-1.01	-1.01	996
		50	20	0.981	0.854	-1.00	-1.01	0.929	0.801	-1.01	-1.01	995
			50	0.997	0.851	-0.98	-0.97	0.956	0.801	-0.99	-0.98	992
20	10	20	20	0.998	0.894	-0.98	-0.95	1.003	0.891	-0.98	-0.96	992
			50	1.007	0.901	-0.99	-0.98	1.023	0.908	-0.99	-0.98	985
		50	20	0.999	0.885	-0.99	-0.97	1.005	0.882	-1.00	-0.98	988
			50	1.022	0.865	-1.00	-0.97	1.049	0.875	-1.00	-0.98	970
	30	20	20	0.994	0.947	-0.99	-0.99	0.967	0.920	-1.00	-1.00	1000
			50	1.012	0.925	-1.00	-0.99	0.990	0.901	-1.00	-1.00	1000
		50	20	1.008	0.954	-1.00	-1.00	0.980	0.923	-1.00	-1.00	1000
			50	1.005	0.935	-0.99	-0.98	0.983	0.913	-0.99	-0.99	1000
50	10	20	20	0.996	0.946	-1.00	-0.99	0.997	0.945	-1.00	-0.99	1000
			50	0.985	0.931	-1.00	-0.99	0.990	0.934	-1.00	-0.99	1000
		50	20	0.998	0.941	-0.99	-0.99	0.979	0.941	-0.99	-0.99	1000
			50	1.006	0.954	-1.00	-1.00	1.015	0.960	-1.00	-1.00	999
	30	20	20	0.995	0.969	-1.00	-1.00	0.983	0.958	-1.00	-1.00	1000
			50	1.012	0.989	-1.00	-1.01	1.002	0.979	-1.00	-1.01	1000
		50	20	1.014	0.983	-1.00	-1.00	1.003	0.972	-1.00	-1.00	1000
			50	0.996	0.964	-1.01	-1.01	0.986	0.954	-1.01	-1.01	1000

Chapter 5

Inference (Two Sample Test) Simulation Study

5.1 Introduction

Typically, researchers are not interested in a simple model such as the one used in chapter 3. The treatments that they can apply to the population or conditions in the survey areas also affect the observed response. They want to know, "Is the probability different between these conditions?" As Stroup (2013a) showed, the Type I error is greatly inflated when test statistics are computed using MLEs of variance components obtained using quadrature. The Firth-adjusted MLE reduces the downward bias of the MLE and, therefore, should improve Type I error control and confidence interval coverage. The goal of this chapter is to show that the adjusted MLE does control Type I error while also providing sufficient power.

5.2 Simulation Framework

For this study, we will look at the case of 10 random subjects with one hundred observations per subject. From the preliminary simulations in Chapter 3, we know this is a case where the downward bias of the quadrature-based MLE for the subject variance becomes evident. Also, in many disciplines, five subjects per treatment would be considered the upper limit for economically viable replications. To consider the Type I error control properties, we will set two treatment means equal to $\ln\left(\frac{p_i}{1-p_i}\right) = -1$ or $p_i \approx 0.27$, and the 10 random subjects will be equally divided between the two treatments. To consider power, the two treatment means will be unequal with $\ln\left(\frac{p_1}{1-p_1}\right) = -1$ or $p_1 \approx 0.27$ and $\ln\left(\frac{p_2}{1-p_2}\right) = -3.125$ or $p_2 \approx 0.04$. Using methods to calculate power described in Stroup (2013), these unequal treatment means with the five subjects per treatment should be detected about 80% of the time. Confidence intervals for the estimated treatment means will be calculated in all cases to compare coverage and width properties. One thousand simulated experiments will be run in each scenario.

Because the treatments are applied at the independent subject level, we utilize the existing functions to obtain the solutions. The only minor changes involve the gradient vector, now three elements instead of two, and the Hessian matrix, a three-by-three instead of two-by-two matrix. These modifications are obtained by combining the gradient vectors and Hessian matrices calculated from the two treatment groups separately.

Let $\mathbf{y}_1$ and $\mathbf{y}_2$ be the vectors of responses for the two treatments respectively. Then $L\left(\begin{pmatrix}\mathbf{y}_1\\ \mathbf{y}_2\end{pmatrix}|\eta_1,\eta_2,\sigma^2\right) = L_1\left(\mathbf{y}_1|\eta_1,\sigma^2\right) + L_2\left(\mathbf{y}_2|\eta_2,\sigma^2\right)$ due to the independence of

the subjects. The gradient vector is then

$$\begin{pmatrix} \frac{\partial L_1}{\partial \eta_1} \\ \frac{\partial L_2}{\partial \eta_2} \\ \frac{\partial L_1}{\partial \sigma^2} + \frac{\partial L_2}{\partial \sigma^2} \end{pmatrix}$$

and the Hessian matrix is

$$\begin{pmatrix} \frac{\partial^2 L_1}{\partial \eta_1 \partial \eta_1} & 0 & \frac{\partial^2 L_1}{\partial \eta_1 \partial \sigma^2} \\ 0 & \frac{\partial^2 L_2}{\partial \eta_2 \partial \eta_2} & \frac{\partial^2 L_2}{\partial \eta_2 \partial \sigma^2} \\ \frac{\partial^2 L_1}{\partial \sigma^2 \partial \eta_1} & \frac{\partial^2 L_2}{\partial \sigma^2 \partial \eta_2} & \frac{\partial^2 L_1}{\partial \sigma^2 \partial \sigma^2} + \frac{\partial^2 L_2}{\partial \sigma^2 \partial \sigma^2} \end{pmatrix}.$$

As all of these pieces are calculated as in the one sample case, estimation in this simulation is completed by separating the data into the two treatment groups at the point of calculating the gradient and Hessian, calculating the individual likelihoods' pieces and combining to solve the estimating equation.

5.3 Results

For Type I error control, we find that the Firth-adjusted MLE outperforms the MLE. For a test with a nominal $\alpha = 0.05$ rejection level and 1,000 Monte Carlo simulations, the expected margin of error is ± 0.02. Therefore, rejection rates that are < 0.03 or > 0.07 indicate problems with Type I error control. Confidence interval coverage should be between $(0.93, 0.97)$ for a stated 95% confidence level. The test using the MLE resulted in the null hypothesis being rejected 7.7% of the time; the test using the Firth-adjusted MLE rejected the null 5.8% for the 1,000 simulated data sets. The quadrature-based MLE does not sufficiently control Type I error due to the downward bias of the variance estimate; the Firth-adjusted MLE is

Table 5.3.1: Equal Treatment Effects

	$\hat{\eta}_{1,MLE}$	$\hat{\eta}_{2,MLE}$	$\hat{\eta}_{1,FAMLE}$	$\hat{\eta}_{2,FAMLE}$
Std. Dev. of Sampling Distribution	0.4629	0.4636	0.4639	0.4646
Average Standard Error	0.3990	0.3990	0.4391	0.4391
Estimated coverage probability	93.1%	91.5%	94.7%	94.5%

properly controlling error. Table 5.3.1 summarizes the confidence interval properties. The standard errors for the MLEs for the treatment effects are too small compared to the standard deviation of the sampling distribution. This is continued evidence of the downward bias of the variance estimate. The standard errors for the Firth-adjusted MLEs are much closer to the standard deviation of the sampling distribution. The lower standard errors of the MLEs result in too narrow confidence intervals and coverage probabilities lower than the nominal 95%. The more accurate standard errors of the Firth-adjusted MLEs result in wider confidence intervals than those for the MLEs, but the coverage probabilities are close to the nominal 95%. Therefore, having less biased variance component estimates leads to better estimates of the variability of the fixed effects estimates.

Because the MLE does not control Type I error, it is inappropriate to use it in practice. Therefore, comparing the power using the MLE against the power using the adjusted MLE is a moot point. Looking at just the results for the Firth-adjusted MLE, we find the rejection rate to be 78.4%. This is very close to the estimated 80% power used to choose the treatment effects for this simulation. Table 5.3.2 summarizes the confidence interval properties for these unequal treatments. Again we see the average standard error for the maximum likelihood fixed effects estimates is smaller than the standard deviation of the sampling distribution. The discrepancy is much smaller for the Firth-adjusted MLE. Confidence interval coverage is within nominal range for the Firth estimates. With the unadjusted

Table 5.3.2: Unequal Treatment Effects

	$\hat{\eta}_{1,MLE}$	$\hat{\eta}_{2,MLE}$	$\hat{\eta}_{1,FAMLE}$	$\hat{\eta}_{2,FAMLE}$
Std. Dev. of Sampling Distribution	0.5237	0.4637	0.5187	0.4633
Average Standard Error	0.4559	0.3905	0.4985	0.4356
Estimated coverage probability	94.7%	91.2%	96.9%	94.3%

MLE, there is not adequate coverage for both treatment effect estimates.

5.4 Conclusions

Inference about treatment effects is usually the goal of any experiment. When the variance is underestimated, any test statistics calculated will be inflated and confidence intervals will be too narrow. The simulations in this chapter show that any inference using quadrature is problematic. However, the Firth-adjusted MLE controls Type I error, achieves adequate power, and has confidence interval coverage equivalent to the nominal coverage probability. Therefore, we conclude that the Firth-adjusted MLE is preferable to the MLE for the purposes of fixed effect testing and estimation.

Chapter 6

Conclusions and Future Research

Excellent estimators (*e.g.,* REML) for variance components in the Gaussian linear mixed model have been known for a long time. REML provides estimates with reduced bias, resulting in accurate hypothesis tests and confidence intervals for fixed effects. To date, there is no REML analog for non-Gaussian mixed models. The estimation methods used for GLMMs have pros and cons that do not provide a comprehensive solution. Because REML in conjunction with certain LMMs is a Firth estimator, the Firth estimator is a likely candidate for a better estimator than the unadjusted MLE for GLMMs.

In the simple random intercept binary response model, simulations show the Firth-adjusted MLE improves the bias of the variance component estimate in both balanced and unbalanced data cases. This behavior is similar to the improvement of the REML estimator over the MLE in linear mixed models. While improved bias is interesting in its own right, an accurate variance component estimate impacts hypothesis tests and confidence intervals, both of which depend on the variance estimate. In the two treatment hypothesis test case, the Firth-adjusted MLE controls Type I error appropriately and achieves the nominal power rate. Similarly,

confidence intervals on the fixed effects have coverage probabilities near the nominal level. This contrasts with the MLE which has an inflated Type I error rate and too narrow confidence intervals.

The investigation of this estimator was limited to treatments applied to independent subjects. Often more than one treatment is applied within a given subject, a treatment and control drug within clinics for example. Instead of separating the treatment groups and combining their information to calculate the score vector and Hessian, each subject contains information on all of the effects in the model. This will require some recalculation of the derivatives necessary to complete the estimation. This design also further complicates the model by introducing the possibility of a subject by treatment interaction term. This second random effect also requires estimation. Based on results of this research, we anticipate the Firth estimation process will improve the estimates of this variance component as well.

Because the Firth estimator is superior to the MLE in fixed effect logistic regression, we limited this initial application to a binomial GLMM. Other non-Gaussian distributions are common for other response variables. The Poisson and negative binomial distributions are used for data arising from counts. Gamma distributions are logical choices for continuous time-to-event data. The behavior of the Firth-adjusted MLE is unknown in these cases and warrants investigation.

Given the results described in this dissertation, the Firth adjustment to the MLE is preferable to the unadjusted MLE for the model investigated. Future work will determine whether the Firth adjustment is a good choice for other models. If so, the Firth estimator should be considered in place of the maximum likelihood estimator.

Bibliography

[1] N. E. Breslow and D. G. Clayton. Approximate inference in generalized linear mixed models. *Journal of the American Statistical Association*, 88(421):9–25, 1993.

[2] N. E. Breslow and X. Lin. Bias correction in generalised linear mixed models with a single component of dispersion. *Biometrika*, 82(1):81–91, 1995.

[3] G. Casella and R.L. Berger. *Statistical Inference, Second Edition.* Duxbury Press, Pacific Grove, CA, 2002.

[4] R. R. Corbeil and S. R. Searle. Restricted maximum likelihood (REML) estimation of variance components in the mixed model. *Technometrics*, 18(1):31–38, 1976.

[5] J. Couton and W. W. Stroup. On the small sample behavior of generalized linear mixed models with complex experiments. *Proceedings of the 25th Conference on Applied Statistics in Agriculture*, 2013.

[6] C. Eisenhart. The assumptions underlying the analysis of variance. *Biometrics*, 3(1):1–21, 1947.

[7] D. Firth. Bias reduction of maximum likelihood estimates. *Biometrika*, 80(1):27–38, 1993.

[8] J. H. Goodnight. Computing MIVQUE0 estimates of variance components. SAS technical report R-105, SAS Institute Inc, 1978.

[9] C. Gotwalt. *What is REML?* JSM Proceedings, San Diego, CA, 2012.

[10] F.A. Graybill. *Theory and Application of the Linear Model.* Duxbury Press, Belmont, CA, 1976.

[11] D. A. Harville. Maximum likelihood approaches to variance component estimation and to related problems. *Journal of the American Statistical Association*, 72(358):320–338, 1977.

[12] G. Heinze and M. Schemper. A solution to the problem of separation in logistic regression. *Statistics in Medicine*, 21(16):2409–2419, 2002.

[13] C. R. Henderson. Estimation of variance and covariance components. *Biometrics*, 9(2):226–252, 1953.

[14] C. R. Henderson, O. Kempthorne, S.R. Searle, and C.M. von Krosigk. The estimation of environmental and genetic trends from records subject to culling. *Biometrics*, 15(2):192–218, 1959.

[15] R. N. Kackar and D. A. Harville. Approximations for standard errors of estimators of fixed and random effects in mixed linear models. *Journal of the American Statistical Association*, 79(388):853–862, 1984.

[16] C. T. Kelley. *Iterative Methods for Linear and Nonlinear Equations.* Society for Industrial and Applied Mathematics, Philadelphia, PA, 1995.

[17] M. G. Kenward and J. H. Roger. Small sample inference for fixed effects from restricted maximum likelihood. *Biometrics*, 53(3):983–997, 1997.

[18] C. E. McCulloch. Maximum likelihood algorithms for generalized linear mixed models. *Journal of the American Statistical Association*, 92(437):162–170, 1997.

[19] J. A. Nelder and R. W. M. Wedderburn. Generalized linear models. *Journal of the Royal Statistical Society Series A (General)*, 135(3):370–384, 1972.

[20] H. D. Patterson and R. Thompson. Recovery of inter-block information when block sizes are unequal. *Biometrika*, 58(3):545–554, 1971.

[21] J. C. Pinheiro and D. M. Bates. Approximations to the log-likelihood function in the nonlinear mixed-effects model. *Journal of Computational and Graphical Statistics*, 4(1):12–35, 1995.

[22] J. C. Pinheiro and E. C. Chao. Efficient Laplacian and adaptive Gaussian quadrature algorithms for multilevel generalized linear mixed models. *Journal of Computational and Graphical Statistics*, 15(1):58–81, 2006.

[23] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing (Second ed.)*. Cambridge University Press, New York, NY, 1992.

[24] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2013.

[25] SAS Institute. *JMP Version 11*. Cary, NC, 1989-2013.

[26] SAS Institute. *SAS System for Windows*. Cary, NC, 2002-2010.

[27] R. Schall. Estimation in gneralized linear models with random effects. *Biometrika*, 78(4):719–727, 1991.

[28] S.R. Searle. *Linear Models*. John Wiley and Sons, Inc., New York, NY, 1971.

[29] S.R. Searle, G. Casella, and C.E. McCulloch. *Variance Components*. John Wiley and Sons, Inc., New York, NY, 1992.

[30] W. W. Stroup. *Generalized Linear Mixed Models: Modern Concepts, Methods and Applications*. CRC Press, Boca Raton, FL, 2013.

[31] W. W. Stroup. Non-normal data in agricultural experiments. *Proceedings of the 25th Conference on Applied Statistics in Agriculture*, 2013.

[32] R. Wicklin. *Statistical programming with SAS/IML(R) software*. SAS Institute Inc, Cary, NC, 2010.

[33] R. Wolfinger and M. O'Connell. Generalized linear mixed models: A pseudo-likelihood approach. *Journal of Statistical Computation and Simulation*, 48(4):233–243, 1993.

Appendix A

Expected Firth Estimation Code

The code and data analysis for this dissertation was generated using SAS/STAT and SAS/IML software, Version 9.3 of the SAS System for Windows. Copyright ĺ 2002-2010 SAS Institute Inc. SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc., Cary, NC, USA. The Broyden method code and line search algorithm were adapted from Press et al. (1992).

This appendix contains all IML functions required to estimate the parameters in a random intercept logit model. It begins with the primary Broyden method function and proceeds through the subroutines required to calculate the estimating equation, gradients and Hessian matrix.

```
start ExpFirthBroy(dat, npoints, max);
z=dat[,1];
y=dat[,2];
nd=dat[,3];
zb=design(z);
n_site=ncol(zb);
zj_init=j(n_site,1,0);
BLUPdata=y||nd||zj_init;
parmvec=GlimmixStartingValues(y,nd,z,npoints);
if parmvec[2]<=0 then parmvec[2]=1;
STPMX=100;
tolf=1e-4;
tolx=1e-6;
```

```
maxiter=max;
tolmin=1e-6;
k=0;
n=nrow(parmvec);
offset=j(n_site,1,parmvec[1]);
BLUPdata=dat||offset;
fstuff=ExpectFirthfmin(parmvec,npoints,BLUPdata);
f=fstuff[1];
fvec=fstuff[2:n+1];
*test for initial guess being a root;
test=0;
do i=1 to n;
  if abs(fvec[i])>test then test=abs(fvec[i]);
end;
if test<(.01*tolf) then do;
  offset=j(n_site,1,parmvec[1]);
  BLUPdata=dat||offset;
  run GlimmixBLUPs(zj,parmvec[2],parmvec[1],BLUPdata,
     npoints);
  print 'N-R converged in ' k ' iterations.';
  print parmvec zj;
  return (parmvec//zj);
end;
else do;
  *calculate max stepsize for linesearch;
  x2=parmvec#parmvec;
  sumx=x2[+];
  t=sumx||n;
  stpmax=STPMX*max(t);
  k=k+1;
  x=parmvec;
  fold=f;
  restrt=1; *ensure initial Hessian computed;
  do while(k<=maxiter);
    if restrt=1 then do; *get initial Hessian;
      offset=j(n_site,1,x[1]);
      BLUPdata=dat||offset;
      run GlimmixBLUPs(hesszj,x[2],x[1],BLUPdata,npoints);
      hessqpoints=GetQPoints(npoints);
      hessdat=BLUPdata[,2:3]||hesszj;
      hessiHess=AllAnalyticHessians(hessdat,x,hessqpoints)
         ;
      hessmat=AnalyticFullHessian(hessiHess,hessdat,x);
    end;
    else do; *or do Broyden update;
      s=x-xold;
      if s[2]=0 then hessmat=hessmat;
      else do;
        w=fvec-fvecold-hessmat*s;
        s=s/(s#s);
        change=w#s;
```

```
    hessmat=hessmat+change;
  end;
end; *Broyden update finished;
*store x, F and f;
xold=x;
fvecold=fvec;
fold=f;
p=-fvec;
d=inv(hessmat)*p;
grad=hessmat*fvec;
offset=j(n_site,1,xold[1]);
BLUPdata=dat||offset;
lnsrchstuff=ExpFirthlnsrch(n,xold,fold,grad,d,stpmax,
   tolx,npoints,BLUPdata);
xnew=lnsrchstuff[1:n];
fnew=lnsrchstuff[n+1];
fvecnew=lnsrchstuff[n+2:2*n+1];
check=lnsrchstuff[2*n+2];
*convergence on function values?;
test=0;
do i=1 to n;
  if abs(fvecnew[i])>test then test=abs(fvecnew[i]);
end;
if test<tolf then do;
  offset=j(n_site,1,xnew[1]);
  BLUPdata=dat||offset;
  run GlimmixBLUPs(zj,xnew[2],xnew[1],BLUPdata,npoints
     );
  print 'N-R converged in ' k ' iterations.';
  print xnew zj;
  return (xnew//zj);
end;
else do;
  if check=1 then do;
    if restrt=1 then do;
      print 'Failed to converge at iteration ' k '.
         Already reinitialized.';
      return (xnew);
    end;
    else do;
      test=0;
      den=max(fnew||.5*n);
      do i=1 to n;
        temp=abs(grad[i])*max(abs(xnew[i])||1)/den;
        if temp>test then test=temp;
      end;
      if test<tolmin then return (xnew);
      else do;
        restrt=1;
        k=k+1;
      end;
```

```
        end;
      end;
      else do;
        *convergence on parms?;
        restrt=0;
        test=0;
        do i=1 to n;
          temp=abs(xnew[i]-xold[i])/max(xnew[1]||1);
          if temp>test then test=temp;
        end;
        if test<tolx then do;
          offset=j(n_site,1,xnew[1]);
          BLUPdata=dat||offset;
          run GlimmixBLUPs(zj,xnew[2],xnew[1],BLUPdata,
             npoints);
          print 'N-R converged in ' k ' iterations.';
          print xnew zj;
          return (xnew//zj);
        end;
        else do;
          x=xnew;
          f=fnew;
          fvec=fvecnew;
          k=k+1;
        end;
      end;
    end;
  end;
  print 'N-R failed to converge.';
  print 'Last iteration ' k ' resulted in ' xnew;
  result={., .};
  return (result);
end;
finish;

start GlimmixStartingValues(success, ntrials, subject,
   npoints);
create MyData var{"success" "ntrials" "subject"};
  append;
close MyData;
submit npoints;
  ods exclude all;
  proc glimmix method=quad(qpoints=&npoints) data=MyData
     hessian gradient itdetails;
  class subject;
  model success/ntrials=/solution;
  random intercept/subject=subject solution;
  ods output CovParms=variance ParameterEstimates=
     intercept Hessian=GHessian;
  run;
  ods select all;
```

```
endsubmit;
use intercept;
  read all var {"Estimate"};
close intercept;
IntEstimate=Estimate;
use variance;
  read all var {"Estimate"};
close variance;
VarEstimate=Estimate;
return(IntEstimate//VarEstimate);
finish;

start ExpectFirthfmin(x,npoints,BLUPdata);
points=GetQPoints(npoints);
n_sites=nrow(BLUPdata);
run GlimmixBLUPs(parmszj,x[2],x[1],BLUPdata,npoints);
newdata=BLUPdata[,1:3]||parmszj;
iGrad=AnalyticGradient2(BLUPdata[,2],BLUPdata[,3],parmszj,
   x[1],
x[2],points);
iGradT=iGrad`;
S=iGradT[,+];
A=ExpectedFirthAdjustment(points,newdata,x);
fvec=S+A;
sum=fvec#fvec;
sum=sum[+];
return (.5*sum//fvec);
finish;
```

The function `GetQPoints(npoints)` is a list of previously calculated quadrature weights and abscissas for various commonly used numbers of quadrature points. The function takes the number of points requested, `npoints`, and returns the weights and abscissas for that number. In the interest of brevity of this appendix, only the first two `npoints` are presented.

```
start GetQPoints(npoints);
if npoints=1 then do;
  absc={0};
  weig={1};
  qpoints=absc||weig;
end;
else if npoints=3 then do;
```

```
  absc={-1.73205080756888, 0, 1.73205080756888};
  weig={0.166666666666667, 0.666666666666667,
     0.166666666666667};
  qpoints=absc||weig;
end;
return(qpoints);
finish;

start GlimmixBLUPs(zj,sigma,init_eta,bdata,npoints);
n_site=nrow(bdata);
offset=j(n_site,1,init_eta);
success=bdata[,2];
ntrials=bdata[,3];
subject=bdata[,1];
create MyData var{"success" "ntrials" "subject" "offset"};
  append;
close MyData;
submit sigma npoints;
  ods exclude all;
  proc glimmix method=quad(qpoints=&npoints) data=MyData
     hessian gradient itdetails;
  class subject;
  model success/ntrials=/offset=offset noint solution;
  parms (&sigma)/hold=1;
  random intercept/subject=subject solution;
  ods output SolutionR=BLUPs;
  run;
  ods select all;
endsubmit;
use BLUPs;
  read all var {"Estimate"};
close BLUPs;
zj=Estimate;
finish;

start AnalyticGradient2(y,n,z,eta,sigma,qpoints);
num=j(nrow(z),2,.);
den=j(nrow(z),2,.);
do i = 1 to nrow(z);
  num[i,]=AnalyticGradient1(y[i],n[i],z[i],eta,sigma,
     qpoints);
  den[i,]=Like(y[i],n[i],z[i],eta,sigma,qpoints);
end;
grad=num/den;
grad=-grad;
return(grad);
finish;

start AnalyticGradient1(y,n,z,eta,sigma,qpoints);
npoints=nrow(qpoints);
pieces=j(nrow(y),2*npoints,.);
```

```
do i = 1 to npoints;
  pieces[,(i-1)*2+1:i*2]=GradWeight(y,n,z,eta,sigma,
     qpoints[i,]);
end;
sum=j(nrow(y),2,0);
do j = 1 to nrow(y);
  do k = 1 to npoints;
    sum[j,]=sum[j,]+pieces[j,(k-1)*2+1:k*2];
  end;
end;
pi=constant('PI');
const=-sqrt(2*pi);
grad=const*sum;
return(grad);
finish;

start GradWeight(y,n,z,eta,sigma,qpoints);
npoints=nrow(qpoints);
weigh=exp(-Di(y,n,z,eta,sigma,qpoints))#DSubTheta(y,n,z,
   eta,sigma,qpoints)#exp((qpoints[,1]##2)/2)#qpoints[,2];
sum=weigh;
return(sum);
finish;

start Di(y,n,z,eta,sigma,qpoints);
npoints=nrow(qpoints);
firstterm=NegLogLik2(y,n,qpoints[,1]/sqrt(
   SecondDerivative2(n,z,eta,sigma))+z,eta,sigma);
secondterm=.5*log(SecondDerivative2(n,z,eta,sigma));
full=firstterm+secondterm;
return(full);
finish;

start NegLogLik2(y,n,z,eta,sigma);
nchoosey=exp(lgamma(n+1)-lgamma(y+1)-lgamma(n-y+1));
pi=constant('PI');
h=-log(nchoosey)-y#(eta+z)+n#(eta+z)+n#log(1+exp(-(eta+z))
   )+log(sqrt(2*pi))+.5*log(sigma)+(z##2)/(2*sigma);
return(h);
finish;

start SecondDerivative2(n,z,eta,sigma);
hpp=1/sigma+(n#exp(eta+z))/((1+exp(eta+z))##2);
return(hpp);
finish;

start DSubTheta(y,n,z,eta,sigma,qpoints);
term1=HWeightSubTheta(y,n,z,eta,sigma,qpoints);
term2=.5*HPPSubThetaHat(n,z,eta,sigma)/SecondDerivative2(n
   ,z,eta,sigma);
full=term1+term2;
```

```
return(full);
finish;

start HWeightSubTheta(y,n,z,eta,sigma,qpoints);
npoints=nrow(qpoints);
term1half1=BlupSubTheta(n,z,eta,sigma)-.5*
   SecondDerivative2(n,z,eta,sigma)##(-1.5)#qpoints[,1]#(
   BlupSubTheta(n,z,eta,sigma)#ThirdDerivative(n,z,eta,
   sigma)+HPPSubTheta(n,z,eta,sigma));
term1half2=FirstDerivative(y,n,z+qpoints[,1]/sqrt(
   SecondDerivative2(n,z,eta,sigma)),eta,sigma);
term2=HSubTheta(y,n,z+qpoints[,1]/sqrt(SecondDerivative2(n
   ,z,eta,sigma)),eta,sigma);
full=term1half1#term1half2+term2;
return(full);
finish;

start BlupSubTheta(n,z,eta,sigma);
zhatsubtheta=-HPSubTheta(n,z,eta,sigma)/SecondDerivative2(
   n,z,eta,sigma);
return(zhatsubtheta);
finish;

start HPSubTheta(n,z,eta,sigma);
hpsubeta=(n#exp(eta+z))/((1+exp(eta+z))##2);
hpsubsig=-z/(sigma##2);
return(hpsubeta||hpsubsig);
finish;

start ThirdDerivative(n,z,eta,sigma);
hppp=(n#exp(eta+z)#(1-exp(eta+z)))/((1+exp(eta+z))##3);
return(hppp);
finish;

start HPPSubTheta(n,z,eta,sigma);
hppsubeta=(n#exp(eta+z)#(1-exp(eta+z)))/((1+exp(eta+z))
   ##3);
hppsubsig=-1/(sigma##2);
hppsubsig=j(nrow(n),1,hppsubsig);
return(hppsubeta||hppsubsig);
finish;

start FirstDerivative(y,n,z,eta,sigma);
hp=-y+n-n/(1+exp(eta+z))+z/sigma;
return(hp);
finish;

start HSubTheta(y,n,z,eta,sigma);
hsubeta=-y+n-n/(1+exp(eta+z));
hsubsig=(sigma-z##2)/(2*sigma##2);
return(hsubeta||hsubsig);
```

```
finish;

start HPPSubThetaHat(n,z,eta,sigma);
value=BlupSubTheta(n,z,eta,sigma)#ThirdDerivative(n,z,eta,
   sigma)+HPPSubTheta(n,z,eta,sigma);
return(value);
finish;

start Like(y,n,z,eta,sigma,qpoints);
n_site=nrow(y);
sum=j(n_site,1,.);
do i = 1 to n_site;
  sum[i]=LikeWeight(y[i],n[i],z[i],eta,sigma,qpoints);
end;
pi=constant('PI');
const=sqrt(2*pi);
like=const*sum;
return(like);
finish;

start LikeWeight(y,n,z,eta,sigma,qpoints);
npoints=nrow(qpoints);
weigh=exp(-(Di(y,n,z,eta,sigma,qpoints)))#exp((qpoints
   [,1]##2)/2)#qpoints[,2];
sum=sum(weigh);
return(sum);
finish;

start ExpectedFirthAdjustment(qpoints,BLUPdata,x);
n_parms=nrow(x);
expectations=Expect(BLUPdata,x,qpoints);
fHess=expectations[1:2,];
A=j(n_parms,1,.);
do j = 1 to n_parms;
  mat=expectations[(j-1)*2+3:j*2+2,];
  whole=inv(fHess)*mat;
  A[j]=-.5*trace(whole);
end;
return(A);
finish;

start Expect(dat,parms,points);
n=dat[1,3];
offset=parms[1];
n_site=nrow(dat);
npoints=nrow(points);
eta=parms[1];
sigma=parms[2];
p0=1/(1+exp(-eta));
mode=floor(n*p0);
*for expected hessian;
```

```
part1=j(nrow(parms),nrow(parms),0);
part2=j(nrow(parms),nrow(parms),0);
check1=0;
check2=0;
*for expectation;
part3a=j(nrow(parms),nrow(parms),0);
part3b=j(nrow(parms),nrow(parms),0);
part4a=j(nrow(parms),nrow(parms),0);
part4b=j(nrow(parms),nrow(parms),0);
do i = 0 to mode while (check1=0);
  y=mode-i;
  bdat=({1}||y||n||offset)//({2}||y||n||offset);
  run GlimmixBLUPs(ezj,sigma,eta,bdat,npoints);
  *expected Hessian pieces;
  hdat=(y||n||ezj[1]);
  h=AllAnalyticHessians(hdat,parms,points);
  prob=Like(y,n,ezj[1],eta,sigma,points);
  piece=h*prob;
  newsum=part1+piece;
  *expectation pieces;
  s=AnalyticGradient2(y,n,ezj[1],eta,sigma,points);
  sst=s`*s;
  diff=h-sst;
  piece2a=s[1]*(diff)*prob;
  piece2b=s[2]*(diff)*prob;
  newsum2a=part3a+piece2a;
  newsum2b=part3b+piece2b;
  if i > 0 then do;
    relsum=(newsum-part1)/part1;
    relsum2a=(newsum2a-part3a)/part3a;
    relsum2b=(newsum2b-part3b)/part3b;
    max=max(relsum||relsum2a||relsum2b);
    if max < 1e-8 then do;
      check1=1;
      part1=newsum;
      part3a=newsum2a;
      part3b=newsum2b;
    end;
  end;
  part1=newsum;
  part3a=newsum2a;
  part3b=newsum2b;
end;
do i = (mode+1) to n while (check2=0);
  y=i;
  bdat=({1}||y||n||offset)//({2}||y||n||offset);
  run GlimmixBLUPs(ezj,sigma,eta,bdat,npoints);
  *expected Hessian pieces;
  hdat=(y||n||ezj[1]);
  h=AllAnalyticHessians(hdat,parms,points);
  prob=Like(y,n,ezj[1],eta,sigma,points);
```

```
  piece=h*prob;
  newsum=part2+piece;
  *expectation pieces;
  s=AnalyticGradient2(y,n,ezj[1],eta,sigma,points);
  sst=s`*s;
  diff=h-sst;
  piece2a=s[1]*(diff)*prob;
  piece2b=s[2]*(diff)*prob;
  newsum2a=part4a+piece2a;
  newsum2b=part4b+piece2b;
  if i > (mode+1) then do;
    relsum=(newsum-part2)/part2;
    relsum2a=(newsum2a-part4a)/part4a;
    relsum2b=(newsum2b-part4b)/part4b;
    max=max(relsum||relsum2a||relsum2b);
    if max < 1e-8 then do;
      check2=1;
      part2=newsum;
      part4a=newsum2a;
      part4b=newsum2b;
    end;
  end;
  part2=newsum;
  part4a=newsum2a;
  part4b=newsum2b;
end;
sum=part1+part2;
sum2a=part3a+part4a;
sum2b=part3b+part4b;
expectedhessian=n_site*(sum);
expectedssta=n_site*(sum2a);
expectedsstb=n_site*(sum2b);
return(expectedhessian//expectedssta//expectedsstb);
finish;

start AllAnalyticHessians(datamat,parms,qpoints);
n_sites=nrow(datamat);
n_parms=nrow(parms);
y=datamat[,1];
n=datamat[,2];
z=datamat[,3];
eta=parms[1];
sigma=parms[2];
numhess=j(n_parms,n_parms*n_sites,.);
do i = 1 to n_sites;
  numhess[,(i-1)*n_parms+1:i*n_parms]=
  AnalyticHessian(y[i],n[i],z[i],eta,sigma,qpoints);
end;
return (numhess);
finish;
```

```
start AllAnalyticHessians(datamat,parms,qpoints);
n_sites=nrow(datamat);
n_parms=nrow(parms);
y=datamat[,1];
n=datamat[,2];
z=datamat[,3];
eta=parms[1];
sigma=parms[2];
numhess=j(n_parms,n_parms*n_sites,.);
do i = 1 to n_sites;
  numhess[,(i-1)*n_parms+1:i*n_parms]=
  AnalyticHessian(y[i],n[i],z[i],eta,sigma,qpoints);
end;
return (numhess);
finish;

start AnalyticHessian(y,n,z,eta,sigma,qpoints);
grad=AnalyticGradient1(y,n,z,eta,sigma,qpoints);
like=Like(y,n,z,eta,sigma,qpoints);
likehess=LikeSubJK(y,n,z,eta,sigma,qpoints);
subetaeta=(grad[1]*grad[1]/(like##2)-likehess[1,1]/like);
subsigsig=(grad[2]*grad[2]/(like##2)-likehess[2,2]/like);
subetasig=(grad[1]*grad[2]/(like##2)-likehess[1,2]/like);
subsigeta=(grad[2]*grad[1]/(like##2)-likehess[2,1]/like);
final=(subetaeta||subetasig)//(subsigeta||subsigsig);
return(final);
finish;

start LikeSubJK(y,n,z,eta,sigma,qpoints);
pi=constant('PI');
const=-sqrt(2*pi);
npoints=nrow(qpoints);
pieces=j(2,2*npoints,.);
do i = 1 to npoints;
  pieces[,(i-1)*2+1:i*2]=HessWeight(y,n,z,eta,sigma,
     qpoints[i,]);
end;
sum=j(2,2,0);
do j = 1 to 2;
  do k = 1 to npoints;
    sum[j,]=sum[j,]+pieces[j,(k-1)*2+1:k*2];
  end;
end;
LikeHess=const*sum;
return(LikeHess);
finish;

start HessWeight(y,n,z,eta,sigma,qpoints);
dst=DSubTheta(y,n,z,eta,sigma,qpoints);
dsjk=DSubJK(y,n,z,eta,sigma,qpoints);
same=exp(-Di(y,n,z,eta,sigma,qpoints))#exp((qpoints
```

```
    [,1]##2)/2)#qpoints[,2];
subetaeta=-(dst[1]*dst[1]-dsjk[1,1])*same;
subsigsig=-(dst[2]*dst[2]-dsjk[2,2])*same;
subetasig=-(dst[1]*dst[2]-dsjk[1,2])*same;
subsigeta=-(dst[2]*dst[1]-dsjk[2,1])*same;
final=(subetaeta||subetasig)//(subsigeta||subsigsig);
return(final);
finish;

start DSubJK(y,n,z,eta,sigma,qpoints);
hwsjk=HWeightSubJK(y,n,z,eta,sigma,qpoints);
hppsjkhat=HPPSubJKHat(y,n,z,eta,sigma);
second=SecondDerivative2(n,z,eta,sigma);
hppst=HPPSubThetaHat(n,z,eta,sigma);
subetaeta=hwsjk[1,1]+.5*hppsjkhat[1,1]/second-.5*hppst[1]*
    hppst[1]/(second##2);
subsigsig=hwsjk[2,2]+.5*hppsjkhat[2,2]/second-.5*hppst[2]*
    hppst[2]/(second##2);
subetasig=hwsjk[1,2]+.5*hppsjkhat[1,2]/second-.5*hppst[1]*
    hppst[2]/(second##2);
subsigeta=hwsjk[2,1]+.5*hppsjkhat[2,1]/second-.5*hppst[2]*
    hppst[1]/(second##2);
final=(subetaeta||subetasig)//(subsigeta||subsigsig);
return(final);
finish;

start HWeightSubJK(y,n,z,eta,sigma,qpoints);
wsjk=WeightSubJK(n,z,eta,sigma,qpoints);
first=FirstDerivative(y,n,z+qpoints[,1]/sqrt(
    SecondDerivative2(n,z,eta,sigma)),eta,sigma);
wst=WeightSubTheta(n,z,eta,sigma,qpoints);
second=SecondDerivative2(n,z+qpoints[,1]/sqrt(
    SecondDerivative2(n,z,eta,sigma)),eta,sigma);
hpst=HPSubTheta(n,z+qpoints[,1]/sqrt(SecondDerivative2(n,z
    ,eta,sigma)),eta,sigma);
hsjk=HSubJK(n,z+qpoints[,1]/sqrt(SecondDerivative2(n,z,eta
    ,sigma)),eta,sigma);
subetaeta=wsjk[1,1]*first+wst[1]*wst[1]*second+wst[1]*hpst
    [1]+wst[1]*hpst[1]+hsjk[1,1];
subsigsig=wsjk[2,2]*first+wst[2]*wst[2]*second+wst[2]*hpst
    [2]+wst[2]*hpst[2]+hsjk[2,2];
subetasig=wsjk[1,2]*first+wst[1]*wst[2]*second+wst[1]*hpst
    [2]+wst[2]*hpst[1]+hsjk[1,2];
subsigeta=wsjk[2,1]*first+wst[2]*wst[1]*second+wst[2]*hpst
    [1]+wst[1]*hpst[2]+hsjk[2,1];
final=(subetaeta||subetasig)//(subsigeta||subsigsig);
return(final);
finish;

start WeightSubJK(n,z,eta,sigma,qpoints);
bsjk=BlupSubJK(n,z,eta,sigma);
```

```
bst=BlupSubTheta(n,z,eta,sigma);
third=ThirdDerivative(n,z,eta,sigma);
hppst=HPPSubTheta(n,z,eta,sigma);
second=SecondDerivative2(n,z,eta,sigma);
fourth=FourthDerivative(n,z,eta,sigma);
hpppst=HPPPSubK(n,z,eta,sigma);
hppsjk=HPPSubJK(n,z,eta,sigma);
subetaeta=bsjk[1,1]+qpoints[,1]#.75*(bst[1]*third+hppst
   [1])*(bst[1]*third+hppst[1])/(second##(5/2))-qpoints
   [,1]#.5*(bsjk[1,1]*third+bst[1]*bst[1]*fourth+bst[1]*
   hpppst[1]+bst[1]*hpppst[1]+hppsjk[1,1])/(second##1.5);
subsigsig=bsjk[2,2]+qpoints[,1]#.75*(bst[2]*third+hppst
   [2])*(bst[2]*third+hppst[2])/(second##(5/2))-qpoints
   [,1]#.5*(bsjk[2,2]*third+bst[2]*bst[2]*fourth+bst[2]*
   hpppst[2]+bst[2]*hpppst[2]+hppsjk[2,2])/(second##1.5);
subetasig=bsjk[1,2]+qpoints[,1]#.75*(bst[1]*third+hppst
   [1])*(bst[2]*third+hppst[2])/(second##(5/2))-qpoints
   [,1]#.5*(bsjk[1,2]*third+bst[1]*bst[2]*fourth+bst[1]*
   hpppst[2]+bst[2]*hpppst[1]+hppsjk[1,2])/(second##1.5);
subsigeta=bsjk[2,1]+qpoints[,1]#.75*(bst[2]*third+hppst
   [2])*(bst[1]*third+hppst[1])/(second##(5/2))-qpoints
   [,1]#.5*(bsjk[2,1]*third+bst[2]*bst[1]*fourth+bst[2]*
   hpppst[1]+bst[1]*hpppst[2]+hppsjk[2,1])/(second##1.5);
final=(subetaeta||subetasig)//(subsigeta||subsigsig);
return(final);
finish;

start BlupSubJK(n,z,eta,sigma);
bst=BlupSubTheta(n,z,eta,sigma);
hppst=HPPSubTheta(n,z,eta,sigma);
hpsjk=HPSubJK(n,z,eta,sigma);
second=SecondDerivative2(n,z,eta,sigma);
hpst=HPSubTheta(n,z,eta,sigma);
third=ThirdDerivative(n,z,eta,sigma);
subetaeta=-(bst[1]*hppst[1]+hpsjk[1,1])/second+hpst[1]*(
   bst[1]*third+hppst[1])/(second##2);
subsigsig=-(bst[2]*hppst[2]+hpsjk[2,2])/second+hpst[2]*(
   bst[2]*third+hppst[2])/(second##2);
subetasig=-(bst[1]*hppst[2]+hpsjk[1,2])/second+hpst[2]*(
   bst[1]*third+hppst[1])/(second##2);
subsigeta=-(bst[2]*hppst[1]+hpsjk[2,1])/second+hpst[1]*(
   bst[2]*third+hppst[2])/(second##2);
final=(subetaeta||subetasig)//(subsigeta||subsigsig);
return(final);
finish;

start HPSubJK(n,z,eta,sigma);
subetaeta=(n#exp(eta+z)#(1-exp(eta+z)))/((1+exp(eta+z))
   ##3);
subsigsig=2*z/(sigma##3);
subetasig=0;
```

```
subsigeta=0;
final=(subetaeta||subetasig)//(subsigeta||subsigsig);
return(final);
finish;

start FourthDerivative(n,z,eta,sigma);
hpppp=(n#exp(eta+z)#(1-4*exp(eta+z)+exp(2*(eta+z))))/((1+
   exp(eta+z))##4);
return(hpppp);
finish;

start HPPPSubK(n,z,eta,sigma);
subeta=(n#exp(eta+z)#(1-4*exp(eta+z)+exp(2*(eta+z))))/((1+
   exp(eta+z))##4);
subsig=0;
final=subeta||subsig;
return(final);
finish;

start HPPSubJK(n,z,eta,sigma);
subetaeta=(n#exp(eta+z)#(1-4*exp(eta+z)+exp(2*(eta+z))))
   /((1+exp(eta+z))##4);
subsigsig=2/(sigma##3);
subetasig=0;
subsigeta=0;
final=(subetaeta||subetasig)//(subsigeta||subsigsig);
return(final);
finish;

start WeightSubTheta(n,z,eta,sigma,qpoints);
bst=BlupSubTheta(n,z,eta,sigma);
second=SecondDerivative2(n,z,eta,sigma);
third=ThirdDerivative(n,z,eta,sigma);
hppst=HPPSubTheta(n,z,eta,sigma);
wst=bst-.5*(second##(-3/2))*(bst*third+hppst)*qpoints[,1];
return(wst);
finish;

start HSubJK(n,z,eta,sigma);
subetaeta=(n#exp(eta+z))/((1+exp(eta+z))##2);
subsigsig=(2*z##2-sigma)/(2*sigma##3);
subetasig=0;
subsigeta=0;
final=(subetaeta||subetasig)//(subsigeta||subsigsig);
return(final);
finish;

start HPPSubJKHat(y,n,z,eta,sigma);
bsjk=BlupSubJK(n,z,eta,sigma);
third=ThirdDerivative(n,z,eta,sigma);
bst=BlupSubTheta(n,z,eta,sigma);
```

```
fourth=FourthDerivative(n,z,eta,sigma);
hpppst=HPPPSubK(n,z,eta,sigma);
hst=HSubTheta(y,n,z,eta,sigma);
hppsjk=HPPSubJK(n,z,eta,sigma);
subetaeta=bsjk[1,1]*third+bst[1]*bst[1]*fourth+bst[1]*
   hpppst[1]+bst[1]*hpppst[1]+hppsjk[1,1];
subsigsig=bsjk[2,2]*third+bst[2]*bst[2]*fourth+bst[2]*
   hpppst[2]+bst[2]*hpppst[2]+hppsjk[2,2];
subetasig=bsjk[1,2]*third+bst[1]*bst[2]*fourth+bst[1]*
   hpppst[2]+bst[2]*hpppst[1]+hppsjk[1,2];
subsigeta=bsjk[2,1]*third+bst[2]*bst[1]*fourth+bst[2]*
   hpppst[1]+bst[1]*hpppst[2]+hppsjk[2,1];
final=(subetaeta||subetasig)//(subsigeta||subsigsig);
return(final);
finish;

start AnalyticFullHessian(iHess,datamat,parms);
n_parms=nrow(parms);
n_sites=nrow(datamat);
fHess=j(n_parms, n_parms, 0);
do k = 1 to (n_sites);
  ind=iHess[,(k-1)*n_parms+1:k*n_parms];
  fHess=fHess+ind;
end;
return(fHess);
finish;

start ExpFirthlnsrch(n,xold,fold,grad,dir,stpmax,tolx,
   npoints,BLUPdata);
check=0;
alf=1E-4;
p2=dir#dir;
sum=p2[+];
sum=sqrt(sum);
if sum>stpmax then dir=dir*stpmax*sum;
slope=grad#dir;
slope=slope[+];
test=0.0;
test2=xold//1.0;
temp=abs(dir)/max(abs(test2));
test=max(temp);
alamin=tolx/test;
alam=1.0;
do while (1=1);
  x=xold+alam*dir;
  if x[2]<=0 then x[2]=.01;
  if x[2]>100 then x[2]=5;
  offset=j(nrow(BLUPdata),1,x[1]);
  BLUPdata=BLUPdata[,1:3]||offset;
  fstuff=ExpectFirthfmin(x,npoints,BLUPdata);
  f=fstuff[1];
```

```
  if alam < alamin then do;
    x=xold;
    check=1;
    return (x//fstuff//check);
  end;
  else if f<=fold+alf*alam*slope then return (x//fstuff//
     check);
  else do;
    if alam=1 then tmplam=-slope/(2*(f-fold-slope));
    else do;
      rhs1=f-fold-alam*slope;
      rhs2=f2-fold-alam2*slope;
      a=(rhs1/(alam*alam)-rhs2/(alam2*alam2))/(alam-alam2)
         ;
      b=(-alam2*rhs1/(alam*alam)+alam*rhs2/(alam2*alam2))
         /(alam-alam2);
      if a=0 then tmplam=-slope/(2*b);
      else do;
        disc=b*b-3*a*slope;
        if disc<0 then tmplam=0.5*alam;
        else if b<=0 then tmplam=(-b+sqrt(disc))/(3*a);
        else tmplam=-slope/(b+sqrt(disc));
      end;
      if tmplam>.5*alam then tmplam=.5*alam;
    end;
  end;
  alam2=alam;
  f2=f;
  lamopt=tmplam||(.1*alam);
  alam=max(lamopt);
end;
finish;
```

Appendix B

Code Modifications for Unbalanced Data

This appendix contains the modification to the Expect function required for the unbalanced data simulations.

```
start UnbExpect(dat,parms,points);
categories=unique(char(dat[,3]));
count=j(ncol(categories),1,0);
do i = 1 to ncol(categories);
  idx=loc(char(dat[,3])=categories[i]);
  count[i]=ncol(idx);
end;
offset=parms[1];
npoints=nrow(points);
eta=parms[1];
sigma=parms[2];
p0=1/(1+exp(-eta));
*for expected hessian;
part1=j(nrow(parms),nrow(parms),0);
part2=j(nrow(parms),nrow(parms),0);
check1=0;
check2=0;
*for expectation;
part3a=j(nrow(parms),nrow(parms),0);
part3b=j(nrow(parms),nrow(parms),0);
part4a=j(nrow(parms),nrow(parms),0);
part4b=j(nrow(parms),nrow(parms),0);
*for final things;
expectedhessian=0;
expectedssta=0;
expectedsstb=0;
```

```
do j = 1 to nrow(count);
  n=num(categories[j]);
  mode=floor(n*p0);
  n_site=count[j];
  do i = 0 to mode while (check1=0);
    y=mode-i;
    bdat=({1}||y||n||offset)//({2}||y||n||offset);
    run GlimmixBLUPs(ezj,sigma,eta,bdat,npoints);
    *expected Hessian pieces;
    hdat=(y||n||ezj[1]);
    h=AllAnalyticHessians(hdat,parms,points);
    prob=Like(y,n,ezj[1],eta,sigma,points);
    piece=h*prob;
    newsum=part1+piece;
    *expectation pieces;
    s=AnalyticGradient2(y,n,ezj[1],eta,sigma,points);
    sst=s`*s;
    diff=h-sst;
    piece2a=s[1]*(diff)*prob;
    piece2b=s[2]*(diff)*prob;
    newsum2a=part3a+piece2a;
    newsum2b=part3b+piece2b;
    if i > 0 then do;
      relsum=(newsum-part1)/part1;
      relsum2a=(newsum2a-part3a)/part3a;
      relsum2b=(newsum2b-part3b)/part3b;
      max=max(relsum||relsum2a||relsum2b);
      if max < 1e-8 then do;
        check1=1;
        part1=newsum;
        part3a=newsum2a;
        part3b=newsum2b;
      end;
    end;
    part1=newsum;
    part3a=newsum2a;
    part3b=newsum2b;
  end;
  do i = (mode+1) to n while (check2=0);
    y=i;
    bdat=({1}||y||n||offset)//({2}||y||n||offset);
    run GlimmixBLUPs(ezj,sigma,eta,bdat,npoints);
    *expected Hessian pieces;
    hdat=(y||n||ezj[1]);
    h=AllAnalyticHessians(hdat,parms,points);
    prob=Like(y,n,ezj[1],eta,sigma,points);
    piece=h*prob;
    newsum=part2+piece;
    *expectation pieces;
    s=AnalyticGradient2(y,n,ezj[1],eta,sigma,points);
    sst=s`*s;
```

```
    diff=h-sst;
    piece2a=s[1]*(diff)*prob;
    piece2b=s[2]*(diff)*prob;
    newsum2a=part4a+piece2a;
    newsum2b=part4b+piece2b;
    if i > (mode+1) then do;
      relsum=(newsum-part2)/part2;
      relsum2a=(newsum2a-part4a)/part4a;
      relsum2b=(newsum2b-part4b)/part4b;
      max=max(relsum||relsum2a||relsum2b);
      if max < 1e-8 then do;
        check2=1;
        part2=newsum;
        part4a=newsum2a;
        part4b=newsum2b;
      end;
    end;
    part2=newsum;
    part4a=newsum2a;
    part4b=newsum2b;
  end;
  sum=part1+part2;
  sum2a=part3a+part4a;
  sum2b=part3b+part4b;
  expectedhessian=expectedhessian+n_site*(sum);
  expectedssta=expectedssta+n_site*(sum2a);
  expectedsstb=expectedsstb+n_site*(sum2b);
end;
return(expectedhessian//expectedssta//expectedsstb);
finish;
```

Appendix C

Code Modifications for Two-Treatment Simulations

This appendix contains the modifications to the gradient vector and Hessian matrix necessary for the two-treatment scenarios.

```
start TwoExpFirthBroy(dat, npoints, max);
z=dat[,1];
y=dat[,2];
nd=dat[,3];
trt=dat[,4];
xb=design(trt);
n_trt=ncol(xb);
zb=design(z);
n_site=ncol(zb);
zj_init=j(n_site,1,0);
BLUPdata=y||nd||zj_init||trt;
parmvec=TwoGlimmixStartingValues(y,nd,z,npoints,trt);
if parmvec[n_trt+1]<=0 then parmvec[n_trt+1]=1;
STPMX=100;
tolf=1e-4;
tolx=1e-6;
maxiter=max;
tolmin=1e-6;
k=0;
n_parm=nrow(parmvec);
n=n_parm;
offset=j(n_site,1,.);
do i = 1 to n_site;
  if trt[i]=1 then offset[i]=parmvec[1];
  else offset[i]=parmvec[2];
end;
```

```
BLUPdata=dat||offset;
fstuff=TwoExpectFirthfmin(parmvec,npoints,BLUPdata);
f=fstuff[1];
fvec=fstuff[2:n+1];
*test for initial guess being a root;
test=0;
do i=1 to n;
  if abs(fvec[i])>test then test=abs(fvec[i]);
end;
if test<(.01*tolf) then do;
  run TwoGlimmixBLUPs(zj,parmvec[n_parm],BLUPdata,npoints)
     ;
  print 'N-R converged in ' k ' iterations.';
  print parmvec zj;
  return (parmvec//zj);
end;
else do;
  *calculate max stepsize for linesearch;
  x2=parmvec#parmvec;
  sumx=x2[+];
  t=sumx||n;
  stpmax=STPMX*max(t);
  k=k+1;
  x=parmvec;
  fold=f;
  restrt=1; *ensure initial Hessian computed;
  do while(k<=maxiter);
    if restrt=1 then do; *get initial Hessian;
      offset=j(n_site,1,.);
      do i = 1 to n_site;
        if trt[i]=1 then offset[i]=x[1];
        else offset[i]=x[2];
      end;
      BLUPdata=dat||offset;
      run TwoGlimmixBLUPs(hesszj,x[n_parm],BLUPdata,
         npoints);
      hessqpoints=GetQPoints(npoints);
      hessdat=BLUPdata[,2:3]||hesszj;
      *carefully build hessian...;
      hessmat=j(n_parm,n_parm,0);
      do i = 1 to n_trt;
        parms=x[i]//x[n_parm];
        hessiHess=AllAnalyticHessians(hessdat[(i-1)*(
           n_site/2)+1:i*(n_site/2),],parms,hessqpoints);
        hessmattemp=AnalyticFullHessian(hessiHess,hessdat
           [(i-1)*(n_site/2)+1:i*(n_site/2),],parms);
        hessmat[i,i]=hessmat[i,i]+hessmattemp[1,1];
        hessmat[i,n_parm]=hessmat[i,n_parm]+hessmattemp
           [1,2];
        hessmat[n_parm,i]=hessmat[i,n_parm];
```

```
    hessmat[n_parm,n_parm]=hessmat[n_parm,n_parm]+
       hessmattemp[2,2];
  end;
end;
else do; *or do Broyden update;
  s=x-xold;
  if s[n_parm]=0 then hessmat=hessmat;
  else do;
    w=fvec-fvecold-hessmat*s;
    s=s/(s#s);
    change=w#s;
    hessmat=hessmat+change;
  end;
end; *Broyden update finished;
*store x, F and f;
xold=x;
fvecold=fvec;
fold=f;
p=-fvec;
d=inv(hessmat)*p;
grad=hessmat*fvec;
offset=j(n_site,1,.);
do i = 1 to n_site;
  if trt[i]=1 then offset[i]=xold[1];
  else offset[i]=xold[2];
end;
BLUPdata=dat||offset;
lnsrchstuff=TwoExpFirthlnsrch(n,xold,fold,grad,d,
   stpmax,tolx,npoints,BLUPdata);
xnew=lnsrchstuff[1:n];
fnew=lnsrchstuff[n+1];
fvecnew=lnsrchstuff[n+2:2*n+1];
check=lnsrchstuff[2*n+2];
*convergence on function values?;
test=0;
do i=1 to n;
  if abs(fvecnew[i])>test then test=abs(fvecnew[i]);
end;
if test<tolf then do;
  offset=j(n_site,1,.);
  do i = 1 to n_site;
    if trt[i]=1 then offset[i]=xnew[1];
    else offset[i]=xnew[2];
  end;
  BLUPdata=dat||offset;
  run TwoGlimmixBLUPs(zj,xnew[n_parm],BLUPdata,npoints
     );
  print 'N-R converged in ' k ' iterations.';
  print xnew zj;
  return (xnew//zj);
end;
```

```
  else do;
    if check=1 then do;
      if restrt=1 then do;
        print 'Failed to converge at iteration ' k '.
           Already reinitialized.';
        return (xnew);
      end;
      else do;
        test=0;
        den=max(fnew||.5*n);
        do i=1 to n;
          temp=abs(grad[i])*max(abs(xnew[i])||1)/den;
          if temp>test then test=temp;
        end;
        if test<tolmin then return (xnew);
        else do;
          restrt=1;
          k=k+1;
        end;
      end;
    end;
    else do;
      *convergence on parms?;
      restrt=0;
      test=0;
      do i=1 to n;
        temp=abs(xnew[i]-xold[i])/max(xnew[1]||1);
        if temp>test then test=temp;
      end;
      if test<tolx then do;
        offset=j(n_site,1,.);
        do i = 1 to n_site;
          if trt[i]=1 then offset[i]=xnew[1];
          else offset[i]=xnew[2];
        end;
        BLUPdata=dat||offset;
        run TwoGlimmixBLUPs(zj,xnew[n_parm],BLUPdata,
           npoints);
        print 'N-R converged in ' k ' iterations.';
        print xnew zj;
        return (xnew//zj);
      end;
      else do;
        x=xnew;
        f=fnew;
        fvec=fvecnew;
        k=k+1;
      end;
    end;
  end;
end;
```

```
  print 'N-R failed to converge.';
  print 'Last iteration ' k ' resulted in ' xnew;
  result={., .};
  return (result);
end;
finish;

start TwoGlimmixStartingValues(success, ntrials, subject,
   npoints , trt);
create MyData var{"success" "ntrials" "subject" "trt"};
  append;
close MyData;
submit npoints;
  ods exclude all;
  proc glimmix method=quad(qpoints=&npoints) data=MyData
     hessian gradient itdetails;
  class subject trt;
  model success/ntrials=trt/noint solution;
  random intercept/subject=subject(trt) solution;
  ods output CovParms=variance ParameterEstimates=
     intercept Hessian=GHessian;
  run;
  ods select all;
endsubmit;
use intercept;
  read all var {"Estimate"};
close intercept;
IntEstimate=Estimate;
use variance;
  read all var {"Estimate"};
close variance;
VarEstimate=Estimate;
return(IntEstimate//VarEstimate);
finish;

start TwoGlimmixBLUPs(zj, sigma, bdata, npoints);
n_site=nrow(bdata);
offset=bdata[,5];
success=bdata[,2];
ntrials=bdata[,3];
subject=bdata[,1];
trt=bdata[,4];
create MyData var{"success" "ntrials" "subject" "offset" "
   trt"};
  append;
close MyData;
submit sigma npoints;
  ods exclude all;
  proc glimmix method=quad(qpoints=&npoints) data=MyData
     hessian gradient itdetails;
  class subject trt;
```

```
  model success/ntrials=/offset=offset noint solution;
  parms (&sigma)/hold=1;
  random intercept/subject=subject(trt) solution;
  ods output SolutionR=BLUPs;
  run;
  ods select all;
endsubmit;
use BLUPs;
  read all var {"Estimate"};
close BLUPs;
zj=Estimate;
finish;

start TwoExpectFirthfmin(parmvec,npoints,BLUPdata);
points=GetQPoints(npoints);
n_sites=nrow(BLUPdata);
n_parm=nrow(parmvec);
n_trt=n_parm-1;
run TwoGlimmixBLUPs(parmszj,parmvec[n_parm],BLUPdata,
   npoints);
newdata=BLUPdata[,1:4]||parmszj;
S=j(n_parm,1,0);
do i = 1 to n_trt;
  iGrad=AnalyticGradient2(BLUPdata[(i-1)*(n_sites/2)+1:i*(
     n_sites/2),2],BLUPdata[(i-1)*(n_sites/2)+1:i*(n_sites
     /2),3],parmszj[(i-1)*(n_sites/2)+1:i*(n_sites/2)],
     parmvec[i],parmvec[n_parm],points);
  iGradT=iGrad`;
  Stemp=iGradT[,+];
  S[i]=S[i]+Stemp[1];
  S[n_parm]=S[n_parm]+Stemp[2];
end;
A=AltTwoExpectedFirthAdjustment(points,newdata,parmvec);
fvec=S+A;
sum=fvec#fvec;
sum=sum[+];
return (.5*sum//fvec);
finish;

start AltTwoExpectedFirthAdjustment(qpoints,BLUPdata,x);
n_parms=nrow(x);
n_trt=n_parms-1;
n_obs=nrow(BLUPdata)/n_trt;
fHess=j(n_parms,n_parms,0);
expectations=j(n_parms*n_parms,n_parms,0);
do i = 1 to n_trt;
  parms=x[i]//x[n_parms];
  expect=Expect(BLUPdata[(i-1)*n_obs+1:i*n_obs,],parms,
     qpoints);
  fHess[i,i]=fHess[i,i]+expect[1,1];
  fHess[i,n_parms]=fHess[i,n_parms]+expect[1,2];
```

```
  fHess[n_parms,i]=fHess[i,n_parms];
  fHess[n_parms,n_parms]=fHess[n_parms,n_parms]+expect
     [2,2];
  do k = 1 to n_parms;
    if i=k then do;
      expectations[(k-1)*n_parms+1*k,k]=expect[3,1];
      expectations[(k-1)*n_parms+1*k,n_parms]=expect[3,2];
      expectations[(k-1)*n_parms+n_parms,k]=expect[4,1];
      expectations[(k-1)*n_parms+n_parms,n_parms]=expect
         [4,2];
    end;
    if k=n_parms then do;
      expectations[n_parms*n_trt+i,i]=expectations[n_parms
         *n_trt+i,i]+expect[5,1];
      expectations[n_parms*n_trt+i,n_parms]=expectations[
         n_parms*n_trt+i,n_parms]=expect[5,2];
      expectations[n_parms*n_parms,i]=expectations[n_parms
         *n_parms,i]+expect[6,1];
      expectations[n_parms*n_parms,n_parms]=expectations[
         n_parms*n_parms,n_parms]+expect[6,2];
    end;
  end;
end;
A=j(n_parms,1,.);
do j = 1 to n_parms;
  mat=expectations[(j-1)*n_parms+1:j*n_parms,];
  whole=inv(fHess)*mat;
  A[j]=-.5*trace(whole);
end;
return(A);
finish;
```

Appendix D

Sample Data Generation and Analysis

This appendix contains examples of how the data were generated for the simulations in this dissertation. It also includes how the analyses were performed and results saved. Subsequent analysis of results to obtain summary statistics and graphs was performed in JMP.

D.1 Balanced Data Generation and Analysis

```
*create data;
submit;
  data binary;
  do expt=1 to 1000;
    do subject = 1 to 10;      /* ==> s = # subj =10 */
      ranint = rannor(42);         /* ==> variance =1 */
      do i = 1 to 100;       /* ==> n = # bern trials  /
         subj = 100 */
        linp = -1 + ranint;       /* ==> eta is -1 */
        pi = 1/(1 + exp(-linp));
        y = ranbin(0,1,pi);
        output;
      end;
    end;
  end;
  drop i;
  run;
  proc means data=binary noprint sum n;
  by expt subject;
  var y;
```

```
  output out=binmulti n=ntrials sum=success;
  run;
endsubmit;
use binmulti;
  read all;
close binmulti;
fulldat=subject||success||ntrials;
z=design(subject);
n_sites=ncol(z);
n_expt=nrow(fulldat)/n_sites;
n_parms=2;
*set up results vectors;
Eglimmixresults=j(n_parms,n_expt,.);
Efirthresults=j(n_parms,n_expt,.);
Eglimhess=j(n_parms,n_expt*n_parms,.);
Efirthhess=j(n_parms,n_expt*n_parms,.);
*Analyze;
do i = 1 to n_expt;
  results=ExpFirthBroy(fulldat[(i-1)*n_sites+1:i*n_sites
     ,],17,100);
  Efirthresults[,i]=results[1:2];
  if results[1] ^= . then do;
    points=GetQPoints(17);
    offset=j(n_sites,1,results[1]);
    BLUPdata=fulldat[(i-1)*n_sites+1:i*n_sites,]||offset;
    run GlimmixBLUPs(hesszj,results[2],results[1],BLUPdata
       ,17);
    dat=fulldat[(i-1)*n_sites+1:i*n_sites,2:3]||hesszj;
    FiHess=AllAnalyticHessians(dat,results[1:2],points);
    Fhessmat=AnalyticFullHessian(FiHess,dat,results[1:2]);
    FInvHess=inv(Fhessmat);
    Efirthhess[,(i-1)*n_parms+1:i*n_parms]=FInvHess;
  end;
  use intercept;
    read all var {"Estimate"};
  close intercept;
  IntEstimate=Estimate;
  use variance;
    read all var {"Estimate"};
  close variance;
  VarEstimate=Estimate;
  use GHessian;
    read all var {"Col1" "Col2"};
  close GHessian;
  GlHess=Col1||Col2;
  GlInvHess=Inv(GlHess);
  Eglimmixresults[,i]=IntEstimate//VarEstimate;
  Eglimhess[,(i-1)*n_parms+1:i*n_parms]=GlInvHess;
end;
egt=Eglimmixresults`;
eft=Efirthresults`;
```

```
print egt eft;
ghess=j(n_expt,3,.); fhess=j(n_expt,3,.);
do i = 1 to n_expt;
  ghess[i,1]=Eglimhess[1,(i-1)*n_parms+1];
  ghess[i,2]=Eglimhess[2,(i-1)*n_parms+1];
  ghess[i,3]=Eglimhess[2,(i-1)*n_parms+2];
  fhess[i,1]=Efirthhess[1,(i-1)*n_parms+1];
  fhess[i,2]=Efirthhess[2,(i-1)*n_parms+1];
  fhess[i,3]=Efirthhess[2,(i-1)*n_parms+2];
end;
print ghess fhess;
```

D.2 Unbalanced Data Generation

The analysis of the unbalanced data simulations is identical to the balanced data case.

```
*create data;
submit;
  data binary;
  do expt=1 to 1000;
    do subject = 1 to 25;     /* ==> s = # subj =50 with
       50% of subjects having missing data*/
      ranint = rannor(2317);        /* ==> variance =1 */
      do i = 1 to 10;      /* ==> n = # bern trials  /
         subj = 10 */
        linp = -1 + ranint;      /* ==> eta is -1 */
        pi = 1/(1 + exp(-linp));
        y = ranbin(0,1,pi);
        output;
      end;
    end;
    do subject = 26 to 50;     /* ==> s = # subj =50 with
       50% of subjects having missing data*/
      ranint = rannor(2317);        /* ==> variance =1 */
      do i = 1 to 8;      /* ==> n = # bern trials  / subj
          = 10 with 20% missing */
        linp = -1 + ranint;      /* ==> eta is -1 */
        pi = 1/(1 + exp(-linp));
        y = ranbin(0,1,pi);
        output;
      end;
```

```
    end;
  end;
  drop i;
  run;
  proc means data=binary noprint sum n;
  by expt subject;
  var y;
  output out=binmulti n=ntrials sum=success;
  run;
endsubmit;
```

D.3 Two-Treatment Data Generation and Analysis

The F-tests for treatment effect were calculated in IML. Summary statistics and graphs were generated using JMP.

```
*create data;
submit;
  data twotreat;
  do expt = 1 to 1000;
    do trt = 1 to 2;
      eta = -1*(trt=1)-1*(trt=2);
      do subject = 1 to 5;     /* ==> s = # subj =10 */
        ranint = rannor(28927456);        /* ==> variance
            =1 */
        do i = 1 to 100;      /* ==> n = # bern trials  /
            subj = 100 */
          linp = eta + ranint;      /* ==> eta is -1 */
          pi = 1/(1 + exp(-linp));
          y = ranbin(0,1,pi);
          output;
        end;
      end;
    end;
  end;
  drop i;
  run;
  data twotreat2;
  set twotreat;
  if trt=2 then subject=subject+5;
  proc means data=twotreat2 noprint sum n;
  by expt subject;
```

```
  var y;
  output out=binsmall n=ntrials sum=success;
  run;
  data twotreatfinal2;
  set binsmall;
  trt=2;
  if subject<6 then trt=1;
  run;
endsubmit;
use twotreatfinal2;
  read all;
close twotreatfinal2;
fulldat=subject||success||ntrials||trt;
z=design(subject);
n_sites=ncol(z);
n_expt=nrow(fulldat)/n_sites;
x=design(trt);
n_trt=ncol(x);
n_parms=n_trt+1;
*set up results vectors;
Eglimmixresults=j(n_parms,n_expt,.);
Efirthresults=j(n_parms,n_expt,.);
Eglimhess=j(n_parms,n_expt*n_parms,.);
Efirthhess=j(n_parms,n_expt*n_parms,.);
*analyze;
do i = 1 to n_expt;
  results=TwoExpFirthBroy(fulldat[(i-1)*n_sites+1:i*
     n_sites,],17,100);
  Efirthresults[,i]=results[1:n_parms];
  if results[1] ^= . then do;
    dat=fulldat[(i-1)*n_sites+1:i*n_sites,];
    offset=j(n_sites,1,.);
    do j = 1 to n_sites;
      if dat[j,4]=1 then offset[j]=results[1];
      else offset[j]=results[2];
    end;
    BLUPdata=dat||offset;
    run TwoGlimmixBLUPs(hesszj,results[n_parms],BLUPdata
       ,17);
    hessqpoints=GetQPoints(17);
    hessdat=BLUPdata[,2:3]||hesszj;
    *carefully build hessian...;
    hessmat=j(n_parms,n_parms,0);
    do j = 1 to n_trt;
      parms=results[j]//results[n_parms];
      hessiHess=AllAnalyticHessians(hessdat[(j-1)*(n_sites
         /2)+1:j*(n_sites/2),],parms,hessqpoints);
      hessmattemp=AnalyticFullHessian(hessiHess,hessdat[(j
         -1)*(n_sites/2)+1:j*(n_sites/2),],parms);
      hessmat[j,j]=hessmat[j,j]+hessmattemp[1,1];
```

```
      hessmat[j,n_parms]=hessmat[j,n_parms]+hessmattemp
         [1,2];
      hessmat[n_parms,j]=hessmat[j,n_parms];
      hessmat[n_parms,n_parms]=hessmat[n_parms,n_parms]+
         hessmattemp[2,2];
    end;
    FInvHess=inv(hessmat);
    Efirthhess[,(i-1)*n_parms+1:i*n_parms]=FInvHess;
  end;
  use intercept;
    read all var {"Estimate"};
  close intercept;
  IntEstimate=Estimate;
  use variance;
    read all var {"Estimate"};
  close variance;
  VarEstimate=Estimate;
  use GHessian;
    read all var {"Col1" "Col2" "Col3"};
  close GHessian;
  GlHess=Col1||Col2||Col3;
  GlInvHess=Inv(GlHess);
  Eglimmixresults[,i]=IntEstimate//VarEstimate;
  Eglimhess[,(i-1)*n_parms+1:i*n_parms]=GlInvHess;
end;
egt=Eglimmixresults`;
eft=Efirthresults`;
print egt eft;
print Eglimhess Efirthhess;
ghess=j(n_expt,6,.);
fhess=j(n_expt,6,.);
do i = 1 to n_expt;
  ghess[i,1]=Eglimhess[1,(i-1)*n_parms+1]; *eta1;
  ghess[i,2]=Eglimhess[2,(i-1)*n_parms+1]; *etaetacross;
  ghess[i,3]=Eglimhess[2,(i-1)*n_parms+2]; *eta2;
  ghess[i,4]=Eglimhess[3,(i-1)*n_parms+1]; *eta1sigcross;
  ghess[i,5]=Eglimhess[3,(i-1)*n_parms+2]; *eta2sigcross;
  ghess[i,6]=Eglimhess[3,(i-1)*n_parms+3]; *sigma;
  fhess[i,1]=Efirthhess[1,(i-1)*n_parms+1]; *eta1;
  fhess[i,2]=Efirthhess[2,(i-1)*n_parms+1]; *etaetacross;
  fhess[i,3]=Efirthhess[2,(i-1)*n_parms+2]; *eta2;
  fhess[i,4]=Efirthhess[3,(i-1)*n_parms+1]; *eta1sigcross;
  fhess[i,5]=Efirthhess[3,(i-1)*n_parms+2]; *eta2sigcross;
  fhess[i,6]=Efirthhess[3,(i-1)*n_parms+3]; *sigma;
end;
print ghess fhess;
*ftests;
dat=fulldat[1:n_sites,];
fres=j(3,n_expt,.);
gres=j(3,n_expt,.);
do i = 1 to n_expt;
```

```
  if Efirthresults[1,i] <> . then do;
    fres[,i]=filikelihoodftest(fulldat[(i-1)*n_sites+1:i*
       n_sites,],Efirthresults[,i],Efirthhess[,(i-1)*
       n_parms+1:i*n_parms]);
    if Eglimmixresults[3,i] <> 0 then do;
      gres[,i]=gllikelihoodftest(fulldat[(i-1)*n_sites+1:i
         *n_sites,],Eglimmixresults[,i],Eglimhess[,(i-1)*
         n_parms+1:i*n_parms]);
    end;
  end;
end;
firthtest=fres';
glimtest=gres';
print firthtest glimtest;
```