HÖGSKOLAN
Dalarna

Evaluation of a New Variance Component Estimation Method - Hierarchical GLM Approach with Application in QTL Analysis

Author: Xia Shen

Supervisor: Lars Rönnegård

D-level Essay in Statistics, Spring 2008.

Department of Economics and Society, Dalarna University College.

Evaluation of a New Variance Component Estimation Method - Hierarchical GLM Approach with Application in QTL Analysis

by

Xia Shen

Submitted to the Department of Economics and Society
on 23 June 2008, in partial fulfillment of the
requirements for the degree of
Master of Science

Abstract

Background: Variance component (VC) models have been widely used in different areas, especially in genetics. To find statistical evidence in Quantitative Trait Loci (QTL) analysis, variance component estimation for random effects is a powerful tool. However, approaches are desired to estimate models with various distribution families. We consider hierarchical generalized linear model (HGLM) for variance component estimation.

Analysis: We implement an HGLM algorithm for normal linear mixed models. The algorithm is available for VC models with only one random effect term. We apply this algorithm to a simple example and a QTL analysis problem. Developing the algorithm, we upgrade it to be available for two or even more random effect terms and apply it to the QTL analysis. The results from HGLM algorithm is good but when comparing to Fisher scoring algorithm, we find that the convergence of HGLM algorithm is much slower. Thus, proposals for accelerating convergence are discussed and included in the algorithm as complement.

Conclusion: HGLM approach is a good method in estimating variance components, since the algorithm is able to be easily extended to non-normal cases, and the gamma GLM fitting in the algorithm overcomes the problem of the sign of variance components. Convergence efficiency is a problem for HGLM algorithm, and some prediction methods may be considered.

Thesis Supervisor: Lars Rönnegård
Title: Lecturer, Ph.D.

Contents

Notation List

$b(\boldsymbol{\theta})$	Cumulant function in generalized linear models.
$E(\mathbf{y})$ / $E(y)$	Expectation vector of $\mathbf{y}$ / Expectation of y.
$g(\boldsymbol{\mu})$	Link function in generalized linear models.
$H(\boldsymbol{\theta}, \mathbf{v}; \mathbf{y}, \mathbf{v})$	H-likelihood of $(\boldsymbol{\theta}, \mathbf{v})$ based on data $(\mathbf{y}, \mathbf{v})$ (omitted if obvious).
$h(\boldsymbol{\theta}, \mathbf{v}; \mathbf{y}, \mathbf{v})$	H-loglihood (h-log-likelihood) of $(\boldsymbol{\theta}, \mathbf{v})$ based on data $(\mathbf{y}, \mathbf{v})$.
$I(\boldsymbol{\theta})$	Fisher information.
$I(\widehat{\boldsymbol{\theta}})$	Observed Fisher information.
$\mathcal{I}(\boldsymbol{\theta})$	Expected Fisher information.
$\mathcal{J}(\boldsymbol{\theta})$	Observed information matrix at $\boldsymbol{\theta}$.
$L(\boldsymbol{\theta}; \mathbf{y})$	Likelihood function of $\boldsymbol{\theta}$ based on data $\mathbf{y}$.
$\ell(\boldsymbol{\theta}; \mathbf{y})$	Loglihood (log-likelihood) function of $\boldsymbol{\theta}$ based on data $\mathbf{y}$.
$N, \phi, \cdots$	Normal English or Greek letters refer to numbers, variables or parameters.
$p_{\mathbf{v}}(\ell)$	Adjusted profile of loglihood ℓ with nuisance parameter $\mathbf{v}$ eliminated.
S_λ	Likelihood ratio test (LRT) statistic with form $-2(\ell_0 - \ell_1)$.
$S(\boldsymbol{\theta})$	Score function of parameter vector $\boldsymbol{\theta}$.
$Var(\mathbf{y})$ / $Var(y)$	Covariance matrix of $\mathbf{y}$ / Variance of y.
$V(\boldsymbol{\mu})$	Variance function in generalized linear models.
$\mathbf{X}, \boldsymbol{\Sigma}, \cdots$	Capital English or Greek letters in bold refer to matrices.
$\mathbf{y}, \boldsymbol{\beta}, \cdots$	Lowercase English or Greek letters in bold refer to vectors.

1 Introduction

Variance component (VC) estimation is widely used in statistical modeling, where different types of effects can be involved in the model. Using likelihood theory, such kind of models are able to be estimated using maximum likelihood (ML). Harville, D.A. (1977) introduced the application of the ML method in variance component estimation of mixed linear models. From the research related to this work, we already have some iterative algorithms for estimating the fixed and random effects as well as the variance components.

In genetic analysis, variance component estimation is a useful technique. One aspect that Robinson G.K. (1991) illustrated is the theory of the best linear unbiased predictor (BLUP) for the estimation of random effects, which is feasible in estimating genetic merits. Iterative algorithms which are used for estimating variance components are available in some genetic problems, for instance, restricted maximum likelihood (REML) estimation with Fisher Scoring approach that we shall mention later in this article. Most variance component estimation algorithms are difficult to extend to various distribution families for random effects. For example, binary data are often analyzed in genetics, and in other statistical applications, exponential family is so important that many essential distributions belong to such a family. Lee, Y. and Nelder, J.A. (1996) proposed the hierarchical generalized linear models (HGLMs) which emancipate the distribution of random effects in the models. HGLM is an epoch in the estimation of variance components, which is rather potential in statistical applications. In genetics, squaring up to fit various distribution of random effects, we have the desirability of applying HGLMs. If HGLMs are able to be controlled and conveniently used, more achievements will be obtained in different occasions.

The aim of the article is to implement and evaluate an HGLM algorithm for mixed linear models which we apply in QTL analysis. In Section 2, we simply introduce the concept of variance component (VC) models, and in Section 3, some background knowledge of hierarchical GLMs is stated. The algorithm for HGLMs is summarized and modified in Section 4 in accordance with Lee, Y., Nelder, J.A. and Pawitan, Y. (2006). An application of the method in QTL analysis is illustrated in Section 5. In Section 6, we evaluate HGLMs by comparing with Fisher Scoring method. And for the purpose of widely application in genetics, an additional part will upgrade the original algorithm by Lee, Y., Nelder, J.A. and Pawitan, Y. (2006) in Section 7, where more random effects can be involved. We also discuss about the convergence and extension of iteration algorithm.

2 VC Models

In general or generalized linear models, we usually have a parameter vector β which refers to some fixed effects. All the random factors are included in the error term or the response vector. However, these models may not be exactly what we desire, since sometimes effects can be interpreted as random and should not be simply included in the error term. Starting with the general form, we have a model as

$$\mathbf{y} = \mathbf{X}\beta + \mathbf{Z}\mathbf{v} + \mathbf{e} \tag{2.1}$$

where $\mathbf{y}$ is a response vector containing N elements, $\mathbf{X}(N \times p)$ and $\mathbf{Z}(N \times q)$ are model matrices, β is the fixed-effect vector, $\mathbf{v}$ is the random-effect vector with multivariate normal distribution $MVN(\mathbf{0}, \mathbf{D})$, and $\mathbf{e} \sim MVN(\mathbf{0}, \mathbf{\Sigma})$ which is independent of $\mathbf{v}$. This is the so-called random effect model. Let $\boldsymbol{\tau}$ be a parameter vector containing all the parameters in $\mathbf{\Sigma}$ and $\mathbf{D}$, then we can express that $\boldsymbol{\tau}$ is formed by the variance components. Thus, the variance component (VC) model, the linear model with random effects and the mixed linear model are the same thing. If we have $\mathbf{\Sigma} = \sigma^2\mathbf{I}_N$ and $\mathbf{D} = \sigma_v^2\mathbf{I}_q$, where $\mathbf{I}_n$ denotes the $n \times n$ identity matrix, $\boldsymbol{\tau}$ should be a vector as $\boldsymbol{\tau} = (\sigma^2, \sigma_v^2)$.

In our analysis, we are interested in estimating the variance-component vector $\boldsymbol{\tau}$. This could be done by applying some likelihood methods in normal linear mixed models, where the REML adjustment (developed by Patterson, H.D. and Thompson, R., 1971) would be useful. However, classical methods often give very slow procedure. The algorithm introduced by Lee, Y., Nelder, J.A. and Pawitan, Y. (2006) will be implemented and discussed later via the hierarchical generalized linear models, and we are going to apply it to some data sets.

3 Hierarchical GLMs

Hierarchical generalized linear models (HGLMs), an extension of generalized linear mixed models (GLMMs), was first presented by Lee, Y. and Nelder, J.A. (1996). HGLM is a reasonable tool for estimating VC models. A general case of the VC model was introduced in the previous section, now we start from the definition of HGLMs.

3.1 HGLMs

The original definition of HGLMs by Lee, Y. and Nelder, J.A. (1996) can be summarized as:

- Conditional distribution of the response vector $\mathbf{y}$ on random effect $\mathbf{u}$ follows a GLM family and

$$E(\mathbf{y}|\mathbf{u}) = \boldsymbol{\mu} \tag{3.1}$$

$$Var(\mathbf{y}|\mathbf{u}) = \phi V(\boldsymbol{\mu}) \tag{3.2}$$

where ϕ is the dispersion parameter and $V(\boldsymbol{\mu})$ is the variance function of GLMs. The kernel of loglihood is given by

$$\sum \left(\frac{y\boldsymbol{\theta} - b(\boldsymbol{\theta})}{\phi} \right) \tag{3.3}$$

where $\boldsymbol{\theta}$ is a function of $\boldsymbol{\mu}$ and known as the canonical parameter. The linear predictor have the form of

$$\eta = g(\boldsymbol{\mu}) = \mathbf{X}\beta + \mathbf{Z}\mathbf{v} \tag{3.4}$$

where $\mathbf{v}$, the random-effect vector, is a monotone function of $\mathbf{u}$, and β is the fixed-effect vector.

- The distribution of random vector $\mathbf{u}$ is conjugate to a GLM family with parameter λ.

HGLMs are flexible in the sense that the distribution of random-effect vector can be specified as needed. The original definition by Lee, Y. and Nelder, J.A. (1996) emphasized conjugacy although sometimes we may not need to constrain us to such a situation. The concept of conjugate distribution as defined by Cox, D.R. and Hinkley, D.V. (1974), leads to the conjugate HGLMs. If the random-effect vector follows a conjugate distribution to the GLM family in the model, we will have a conjugate HGLM. The reason why noting this is that for weak canonical scale (see Definition 3.2) of $\mathbf{v}$, conjugacy has a nice characteristic. The scale of random effects is not important in conjugate distributions, since they will be integrated out. We put some combinations of the distributions in HGLMs as follows, which are specified in Lee, Y., Nelder, J.A. and Pawitan, Y. (2006). In this article, the conjugate normal-normal HGLM will be implemented and applied.

Table 3.1 Examples of HGLMs from Lee, Y., Nelder, J.A. and Pawitan, Y. (2006)

$\mathbf{y}|\mathbf{u}$ distribution	$g(\boldsymbol{\mu})^{\dagger}$	$\mathbf{u}$ distribution	$v(\mathbf{u})$	Model
Normal	**id**	**Normal**	**id**	**Conjugate HGLM (Linear mixed model)**
Binomial	logit	Beta	logit	Conjugate HGLM (beta-binomial model)
Binomial	logit	Normal	id	Binomial GLMM
Binomial	comp	Gamma	log	HGLM
Gamma	recip	Inverse-gamma	recip	Conjugate HGLM
Gamma	log	Inverse-gamma	recip	Conjugate HGLM (non-canonical link)
Gamma	log	Gamma	log	HGLM
Poisson	log	Normal	id	Poisson GLMM
Poisson	log	Gamma	log	Conjugate HGLM

† id=identity, recip=reciprocal, comp=complementary-log-log

3.2 Normal Linear Mixed Models

Conjugate normal-normal HGLM is the normal linear mixed model as we usually know. According to the definition by Lee, Y. and Nelder, J.A. (1996), if in conjugate distribution of random vector $\mathbf{u}$, we have

$$E(\mathbf{u}) = \boldsymbol{\psi}_M \tag{3.5}$$

$$Var(\mathbf{u}) = \rho V_M(\boldsymbol{\psi}_M) \tag{3.6}$$

and the kernel of loglihood of random effects has the form

$$\sum \left(\frac{\boldsymbol{\psi}_M \boldsymbol{\theta}_M - b(\boldsymbol{\theta}_M)}{\lambda} \right) \tag{3.7}$$

the model can be specified with terms in Table 3.2 below.

Table 3.2 Normal response and conjugate normal random effects

Distribution of Response $\mathbf{y}|\mathbf{v}$	$V(\boldsymbol{\mu})$	$\boldsymbol{\theta} = \theta(\boldsymbol{\mu})$	$b(\boldsymbol{\theta})$		
Normal	1	μ	$\frac{\theta^2}{2}$		
Distribution of random effects $\mathbf{u}$	$V_M(\boldsymbol{\psi}_M)$	$\boldsymbol{\theta}_M = \theta_M(\boldsymbol{\psi}_M)$	$b_M(\boldsymbol{\theta}_M)$	$\boldsymbol{\psi}_M$	ρ
Normal	1	ψ_M	$\frac{\theta_M^2}{2}$	0	λ

Estimation of the mean parameters and variance components can be achieved through different kinds of methods, in which, as mentioned in Section 2, classical ones such as ordinary REML adjustment are available. However, we are interested in another method which may step up the estimation procedure. To approach such a method, we introduce the h-likelihood in the next subsection.

3.3 Extended Likelihood and H-likelihood

For details of the concepts introduced below see Lee, Y., Nelder, J.A. and Pawitan, Y. (2006) or Pawitan, Y. (2001). Generally speaking, there are two kinds of likelihood. One is the classical Fisher likelihood, which typically has the form

$$L(\boldsymbol{\theta};\mathbf{y}) = f_{\boldsymbol{\theta}}(\mathbf{y}) \tag{3.8}$$

where the right-hand side is used for generating data according to the distribution of $\mathbf{y}$ with some fixed $\boldsymbol{\theta}$, and the left-hand side is applied to make inference about $\boldsymbol{\theta}$ based on the sample $\mathbf{y}$. This is the likelihood principle we usually use, however what we only see is its advantage on estimating or inferring fixed parameters. Considering the VC model in Equation (2.1), if we are interested in estimating both parameters β and $\mathbf{v}$, simply maximizing Fisher likelihood gives no information about the random-effect parameter $\mathbf{v}$. Thus, we need to generate the likelihood principle, which can deal with not only fixed parameter $\boldsymbol{\theta}$, observed variable $\mathbf{y}$, but the unobservable random quantity $\mathbf{v}$ as well. For the generating data part, we have

$$f_{\boldsymbol{\theta}}(\mathbf{v})f_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{v}) = f_{\boldsymbol{\theta}}(\mathbf{y},\mathbf{v}) \tag{3.9}$$

where a fixed $\mathbf{v}$ can be first generated from $f_{\boldsymbol{\theta}}(\mathbf{v})$ and then $\mathbf{y}$ will be generated from the conditional density $f_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{v})$. For parameter inference, the following extended likelihood is defined.

Definition 3.1 *The extended likelihood for fixed parameter $\boldsymbol{\theta}$ and random quantity $\mathbf{v}$ is*

$$L(\boldsymbol{\theta},\mathbf{v};\mathbf{y},\mathbf{v}) = L(\boldsymbol{\theta};\mathbf{y})L(\boldsymbol{\theta},\mathbf{v};\mathbf{v}|\mathbf{y}) \tag{3.10}$$

where $L(\boldsymbol{\theta};\mathbf{y})$ is the marginal likelihood and $L(\boldsymbol{\theta},\mathbf{v};\mathbf{v}|\mathbf{y}) \equiv f_{\boldsymbol{\theta}}(\mathbf{v}|\mathbf{y})$ is the conditional likelihood.

In the right-hand side of Equation (3.10), $\boldsymbol{\theta}$ can be first inferred from the marginal likelihood $L(\boldsymbol{\theta};\mathbf{y})$, then with $\boldsymbol{\theta}$ known, $\mathbf{v}$ can be inferred from the conditional likelihood $L(\boldsymbol{\theta},\mathbf{v};\mathbf{v}|\mathbf{y})$. Such a definition was already used by Butler, R.W. (1986), Berger, J.O. and Wolpert, R. (1984) and Bjørnstad, J.F. (1996), and it is suitable for working with models like VC models we mentioned in Section 2. Detailed inference procedure for both fixed and random parameters can be found in Lee, Y., Nelder, J.A. and Pawitan, Y. (2006), and computationally such an inference making from extended likelihood is much less complicated and faster than some popular simulation methods such as the EM (expectation-maximization) algorithm (Dempster, A.P., Laird, N.M. and Rubin, D.B., 1977), Monte-Carlo EM (Vaida, F. and Meng, X.L., 2004) and Gibbs sampling (Gelfand, A.E. and Smith, A.F.M., 1990). However, to make the inference for HGLMs more straightforward, a special extended likelihood approach should be introduced. The estimation from the extended likelihood may be wrong when the scale of $\mathbf{v}$ varies. But some scales always lead to the correct estimation. Assume there exists a scale $\mathbf{v}$ such that we have the following likelihood ratio

$$\frac{L(\boldsymbol{\theta}_1,\widehat{\mathbf{v}}_{\boldsymbol{\theta}_1};\mathbf{y},\mathbf{v})}{L(\boldsymbol{\theta}_2,\widehat{\mathbf{v}}_{\boldsymbol{\theta}_2};\mathbf{y},\mathbf{v})} = \frac{L(\boldsymbol{\theta}_1;\mathbf{y})}{L(\boldsymbol{\theta}_2;\mathbf{y})} \Leftrightarrow \frac{L(\boldsymbol{\theta}_1,\widehat{\mathbf{v}}_{\boldsymbol{\theta}_1};\mathbf{v}|\mathbf{y})}{L(\boldsymbol{\theta}_2,\widehat{\mathbf{v}}_{\boldsymbol{\theta}_2};\mathbf{v}|\mathbf{y})} = 1 \tag{3.11}$$

where $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$ are two arbitrary values of fixed parameter $\boldsymbol{\theta}$, and $\widehat{\mathbf{v}}_{\boldsymbol{\theta}_k}(k=1,2)$ is the MLE of $\mathbf{v}$ for $\boldsymbol{\theta}$ at $\boldsymbol{\theta}_k$. In such a situation, there's no information about $\boldsymbol{\theta}$ in neither $\widehat{\mathbf{v}}_{\boldsymbol{\theta}}$ nor conditional profile likelihood $L(\boldsymbol{\theta}_1,\widehat{\mathbf{v}}_{\boldsymbol{\theta}_1};\mathbf{v}|\mathbf{y})$, which satisfies the classical likelihood principle. Then we call the scale of $\mathbf{v}$ a canonical scale, and we have the following definition.

Definition 3.2 $L(\boldsymbol{\theta},\mathbf{v};\mathbf{y},\mathbf{v})$ *is called h-likelihood if the parameter* $\mathbf{v}$ *is canonical.*

The h-likelihood is denoted by $H(\boldsymbol{\theta},\mathbf{v})$ and h-loglihood $h(\boldsymbol{\theta},\mathbf{v})$. The h-likelihood has a nice feature for further inference in HGLMs. As we desire, the MLE of $\boldsymbol{\theta}$ from $H(\boldsymbol{\theta},\mathbf{v})$ will be exactly the same as that from the marginal likelihood $L(\boldsymbol{\theta};\mathbf{y})$.

Now come back to the normal-normal HGLMs we show interest in the previous subsections. According to Definition 3.1 and Equation (3.9), we have the extended loglihood function as

$$\begin{aligned} \ell_e(\beta,\boldsymbol{\tau},\mathbf{v}) &= \log f(\mathbf{y},\mathbf{v}) \qquad (3.12)\\ &= \log f(\mathbf{y}|\mathbf{v}) + \log f(\mathbf{v})\\ &= -\frac{1}{2}\log|2\pi\boldsymbol{\Sigma}|\\ &\quad -\frac{1}{2}(\mathbf{y}-\mathbf{X}\beta-\mathbf{Z}\mathbf{v})'\boldsymbol{\Sigma}^{-1}(\mathbf{y}-\mathbf{X}\beta-\mathbf{Z}\mathbf{v})\\ &\quad -\frac{1}{2}\log|2\pi\mathbf{D}| - \frac{1}{2}\mathbf{v}'\mathbf{D}^{-1}\mathbf{v} \end{aligned}$$

Given the fixed parameters and maximizing the extended loglihood, we have

$$\widehat{\mathbf{v}} = \left(\mathbf{Z}'\boldsymbol{\Sigma}^{-1}\mathbf{Z}+\mathbf{D}^{-1}\right)^{-1}\mathbf{Z}'\boldsymbol{\Sigma}^{-1}(\mathbf{y}-\mathbf{X}\beta) \tag{3.13}$$

and the observed Fisher information

$$I(\widehat{\mathbf{v}}) = -\frac{\partial^2}{\partial \mathbf{v} \partial \mathbf{v}'} \ell_e(\boldsymbol{\beta}, \boldsymbol{\tau}, \mathbf{v})\bigg|_{\mathbf{v}=\widehat{\mathbf{v}}} = \mathbf{Z}'\boldsymbol{\Sigma}^{-1}\mathbf{Z} + \mathbf{D}^{-1} \tag{3.14}$$

Finding a canonical scale is often difficult but there are some ways to check whether or not a scale is canonical. Since only $\boldsymbol{\tau}$ is contained in the observed Fisher information of $\mathbf{v}$, $\mathbf{v}$ can be canonical for $\boldsymbol{\beta}$ and actually it is. Therefore, inference about $\boldsymbol{\beta}$ and $\mathbf{v}$ can be made jointly because the extended likelihood we have is actually an h-likelihood. Detailed theoretical work for estimating $\boldsymbol{\beta}$ and $\mathbf{v}$ can be found in Lee, Y., Nelder, J.A. and Pawitan, Y. (2006). What's more, the marginal loglihood we need for estimating $\boldsymbol{\tau}$ can be derived as an adjusted profile likelihood as

$$\ell(\boldsymbol{\beta}, \boldsymbol{\tau}) = p_{\mathbf{v}}(h|\boldsymbol{\beta}, \boldsymbol{\tau}) = h(\boldsymbol{\beta}, \boldsymbol{\tau}, \widehat{\mathbf{v}}_{\boldsymbol{\beta},\boldsymbol{\tau}}) - \frac{1}{2}\log\left|\frac{I(\widehat{\mathbf{v}})}{2\pi}\right| \tag{3.15}$$

In terms of h-likelihood, the profile likelihood of variance components is

$$\ell_p(\boldsymbol{\tau}) = \ell(\widehat{\boldsymbol{\beta}}_{\boldsymbol{\tau}}, \boldsymbol{\tau}) = h(\widehat{\boldsymbol{\beta}}_{\boldsymbol{\tau}}, \boldsymbol{\tau}, \widehat{\mathbf{v}}_{\boldsymbol{\tau}}) - \frac{1}{2}\log\left|\frac{I(\widehat{\mathbf{v}})}{2\pi}\right| \tag{3.16}$$

If REML adjustment for the estimation of fixed effects $\boldsymbol{\beta}$ is included, we have

$$p_{\boldsymbol{\beta},\mathbf{v}}(\ell|\boldsymbol{\tau}) = \ell\left(\widehat{\boldsymbol{\beta}}_{\boldsymbol{\tau}}, \boldsymbol{\tau}\right) - \frac{1}{2}\log\left|\frac{\mathbf{X}'\mathbf{V}^{-1}\mathbf{X}}{2\pi}\right| \tag{3.17}$$

which can be seen further in Fitting Algorithm.

We intend to carry out an algorithm from the h-likelihood approach for the normal case HGLMs. Lee, Y., Nelder, J.A. and Pawitan, Y. (2006) say about this approach that these results about random effects are only approximately true (Laplace's approximation) for most non-normal cases.

4 Fitting Algorithm

The h-likelihood approach in Subsection 3.3 actually suggests a fitting algorithm. In this section, the fitting algorithm will be first summarized for an augmented linear model, and then we shall implement it in a R[i] function, which can be flexibly applied to data sets of interest.

4.1 Algorithm Summary

The fitting algorithm (summarized from Lee, Nelder and Pawitan, 2006) that we shall implement can be made out from the h-likelihood approach in Subsection 3.3. The VC model (2.1) can be written as

$$\mathbf{y}_a = \mathbf{T}\boldsymbol{\delta} + \mathbf{e}_a \tag{4.1}$$

where $\mathbf{e}_a \sim MVN(\mathbf{0}, \boldsymbol{\Sigma}_a)$, and

$$\begin{aligned} \mathbf{y}_a &= \begin{pmatrix} \mathbf{y} \\ \psi_M \end{pmatrix}, \mathbf{T} = \begin{pmatrix} \mathbf{X} & \mathbf{Z} \\ \mathbf{0} & \mathbf{I} \end{pmatrix}, \boldsymbol{\delta} = \begin{pmatrix} \boldsymbol{\beta} \\ \mathbf{v} \end{pmatrix}, \\ \mathbf{e}_a &= \begin{pmatrix} \mathbf{e} \\ \mathbf{e}_M \end{pmatrix}, \boldsymbol{\Sigma}_a = \begin{pmatrix} \boldsymbol{\Sigma} & \mathbf{0} \\ \mathbf{0} & \mathbf{D} \end{pmatrix}. \end{aligned}$$

Here, $\boldsymbol{\Sigma} = \sigma^2\mathbf{I}_N$, $\mathbf{D} = \sigma_v^2\mathbf{I}_q$ and $\boldsymbol{\tau} = (\sigma^2, \sigma_v^2)$. The subscript M of ψ_M is to indicate the random effects appear in the linear predictor for the mean. Therefore ψ_M is set to be $\mathbf{0}$ as response for estimating the variance component in matrix $\mathbf{D}$. The following algorithm is constructed through IWLS for the augmented model to estimate $(\boldsymbol{\beta}, \boldsymbol{\tau}, \mathbf{v})$. Mixed models equations (MME) and REML estimation are combined in the algorithm procedure.

Algorithm 4.1 *IWLS for the augmented linear mixed model (modified from Lee, Nelder and Pawitan, 2006):*

- *Set a starting value of variance-component parameter $\boldsymbol{\tau}$.*

- *Given the current estimate of $\boldsymbol{\tau}$, solve the augmented mixed models equation $\mathbf{T}'\boldsymbol{\Sigma}_a^{-1}\mathbf{T}\widehat{\boldsymbol{\delta}} = \mathbf{T}'\boldsymbol{\Sigma}_a^{-1}\mathbf{y}_a$ (derived from Equation (4.1)) to get a new estimate of $\boldsymbol{\delta}$:*

$$\widehat{\boldsymbol{\delta}} = \left(\mathbf{T}'\boldsymbol{\Sigma}_a^{-1}\mathbf{T}\right)^{-1}\mathbf{T}'\boldsymbol{\Sigma}_a^{-1}\mathbf{y}_a \tag{4.2}$$

- *Given the current estimate of $\boldsymbol{\delta}$, calculate the deviance components corresponding to $\mathbf{e}_a$ as the squared residuals:*

$$d_i = (y_i - \mathbf{X}_i\widehat{\beta} - \mathbf{Z}_i\widehat{\mathbf{v}})^2 \tag{4.3}$$

$$d_{Mi} = (\psi_M - \widehat{v}_i)^2 = \widehat{v}_i^2 \tag{4.4}$$

where $\mathbf{X}_i$ and $\mathbf{Z}_i$ refer to the i'th row vector in $\mathbf{X}$ and $\mathbf{Z}$.

- *Calculate the hat-matrix:*

$$\mathbf{H}_a = \mathbf{T}\left(\mathbf{T}'\boldsymbol{\Sigma}_a^{-1}\mathbf{T}\right)^{-1}\mathbf{T}'\boldsymbol{\Sigma}_a^{-1} \tag{4.5}$$

and set the leverages:

$$\begin{pmatrix} \mathbf{q} \\ \mathbf{q}_M \end{pmatrix} = \begin{pmatrix} (h_{11}, \cdots, h_{NN})' \\ (h_{N+1,N+1}, \cdots, h_{N+q,N+q})' \end{pmatrix} \tag{4.6}$$

where h_{ii} refers to the i'th diagonal element of $\mathbf{H}_a$.

[i] R codes in detail are presented in Appendix.

- *Let*

$$d_i^* = \frac{d_i}{1 - q_i} \tag{4.7}$$

$$d_{Mi}^* = \frac{d_{Mi}}{1 - q_{Mi}} \tag{4.8}$$

$$\varpi = \frac{\mathbf{1 - q}}{2} \tag{4.9}$$

$$\varpi_M = \frac{\mathbf{1 - q}_M}{2} \tag{4.10}$$

and model a GLM with response $\mathbf{d}^*$*, gamma family*[ii]*, identity link, intercept only linear predictor and prior weight* ϖ *to get an updated estimate for* σ^2*; model a GLM with response* $\mathbf{d}_M^*$*, gamma family, identity link, intercept only linear predictor and prior weight* ϖ_M *to get an updated estimate for* σ_v^2*. Note that* $E(d_i^*) = \sigma^2$*,* $Var(d_i^*) = 2\sigma^2/(1 - q_i)$ *and* $E(d_{Mi}^*) = \sigma_v^2$*,* $Var(d_{Mi}^*) = 2\sigma_v^2/(1 - q_{Mi})$*.*

- *Iterate from the second step until convergence. At convergence, calculate standard errors of* $\widehat{\boldsymbol{\beta}}$ *and* $\widehat{\mathbf{v}} - \mathbf{v}$ *from* $\mathbf{H}^{-1}$*, where*

$$\mathbf{H} = \mathbf{T}'\mathbf{\Sigma}_a^{-1}\mathbf{T} \tag{4.11}$$

After checking singularity, calculate the standard errors of $\widehat{\boldsymbol{\tau}}$ *from the negative inverse of the Hessian*[iii] *of*

$$p_{\beta,\mathbf{v}}(h|\boldsymbol{\tau}) = \ell(\widehat{\boldsymbol{\beta}}_{\boldsymbol{\tau}}, \boldsymbol{\tau}) - \frac{1}{2}\log\left|\frac{\mathbf{X}'\mathbf{V}^{-1}\mathbf{X}}{2\pi}\right| \tag{4.12}$$

where $\mathbf{V} = \mathbf{ZDZ}' + \mathbf{\Sigma}$ *and*

$$\ell(\boldsymbol{\beta}, \boldsymbol{\tau}) = -\frac{1}{2}\log|2\pi\mathbf{V}| - \frac{1}{2}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})'\mathbf{V}^{-1}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) \tag{4.13}$$

- *Calculate the likelihood ratio test (LRT) statistic of test*

$$\begin{aligned} H_0 &: \sigma_v^2 = 0 \\ H_1 &: \sigma_v^2 > 0 \end{aligned}$$

in accordance with the profile likelihood in (4.12), we have

$$S_\lambda = -2\left(\ell_0 - p_{\beta,\mathbf{v}}(h|\widehat{\boldsymbol{\tau}})\right) \tag{4.14}$$

where ℓ_0 *is the likelihood under the null hypothesis.*

The above algorithm is an extension of ordinary REML methods. In the following subsection, this algorithm will be implemented in R and we shall see how it works for a simple data set.

4.2 Programming Implementation

Assume that we have the one-way random effects model as

$$y_{ij} = \mu + v_i + e_{ij} \tag{4.15}$$

for $i = 1, 2, \cdots, q$ and $j = 1, 2, \cdots, n$. Writing it with the form of VC models in Equation (2.1), if the total data size is $N = qn$, we have

$$\begin{aligned} \mathbf{X} &= \mathbf{1}_N \\ \boldsymbol{\beta} &= \mu \\ \mathbf{Z} &= \left(z_{ij} = \begin{cases} 1, \textit{ if } y_{ij} \textit{ comes from the } i\textit{'th group} \\ 0, \textit{ otherwise} \end{cases} \right)_{N \times q} \\ \mathbf{v} &= (v_1, v_2, \cdots, v_q)' \\ \mathbf{\Sigma} &= \sigma^2 \mathbf{I}_N \\ \mathbf{D} &= \sigma_v^2 \mathbf{I}_q \end{aligned}$$

where $\mathbf{1}_N$ is the column vector of N ones and the variance component parameter is $\boldsymbol{\tau} = (\sigma^2, \sigma_v^2)$.

This model will be carried out using our algorithm on the data in Table 4.1. This data set (from Fears, T.R., Benichou, J. and Gail, M.H., 1996) shows the estrone measurement results from five menopausal women, in which 16 measurements were taken from each person. The application on these data tries to detect whether the variability among the data comes mainly from the difference among persons, i.e. whether the person effects are significant large. For this purpose, naturally, we model the persons as random effects and apply the model in (4.15). Notice that after the transformation of $y_{ij} = 10\log_{10}(\textit{estrone measurements})$, $q = 5$ and $n = 16$, the model will be available for the data.

Table 4.1 Estrone measurements from five menopausal women from Fears et al. (1996)

$i = 1$	2	3	4	5	$i = 1$	2	3	4	5
23	25	38	14	46	22	26	35	17	32
23	33	38	16	36	22	30	40	18	31
22	27	41	15	30	23	30	41	20	30
20	27	38	19	29	23	29	37	18	32
25	30	38	20	36	27	29	28	12	25
22	28	32	22	31	19	37	36	17	29
27	24	38	16	30	23	24	30	15	31
25	22	42	19	32	18	28	37	13	32

[ii] We can obtain REML estimators from fitting gamma-GLMs.

[iii] The Hessian matrix of a function at a certain value of an argument can be calculated using the Richardson approximation. Package {`numDeriv`} in R can do this approximation (for help see http://rss.acs.unt.edu/Rdoc/library/numDeriv/html/hessian.html).

The summary of estimation results from R is shown in Table 4.2 and Figure 4.1, in which estimates of the four parameters are obtained and the iterations converge in just five times. The convergence condition is set up to be $\max|\boldsymbol{\tau}_k - \boldsymbol{\tau}_{k-1}| \leqslant 1 \times 10^{-5}$ (k is the current number of iteration). In fact as shown in Figure 4.1, estimates of variance components get quite close to the final convergence values after just two iterations.

Table 4.2 Estimation results for estrone measurements data

	$\widehat{\mu}$	$\widehat{\mathbf{v}} = (v_1, v_2, v_3, v_4, v_5)'$
Estimates	14.1751	$(-.6229, .2691, 1.4431, -1.9196, .8303)'$
	$\widehat{\sigma}^2$	$\widehat{\sigma}_v^2$
Estimates	.3254	1.7494
Total Iteration Number	5	

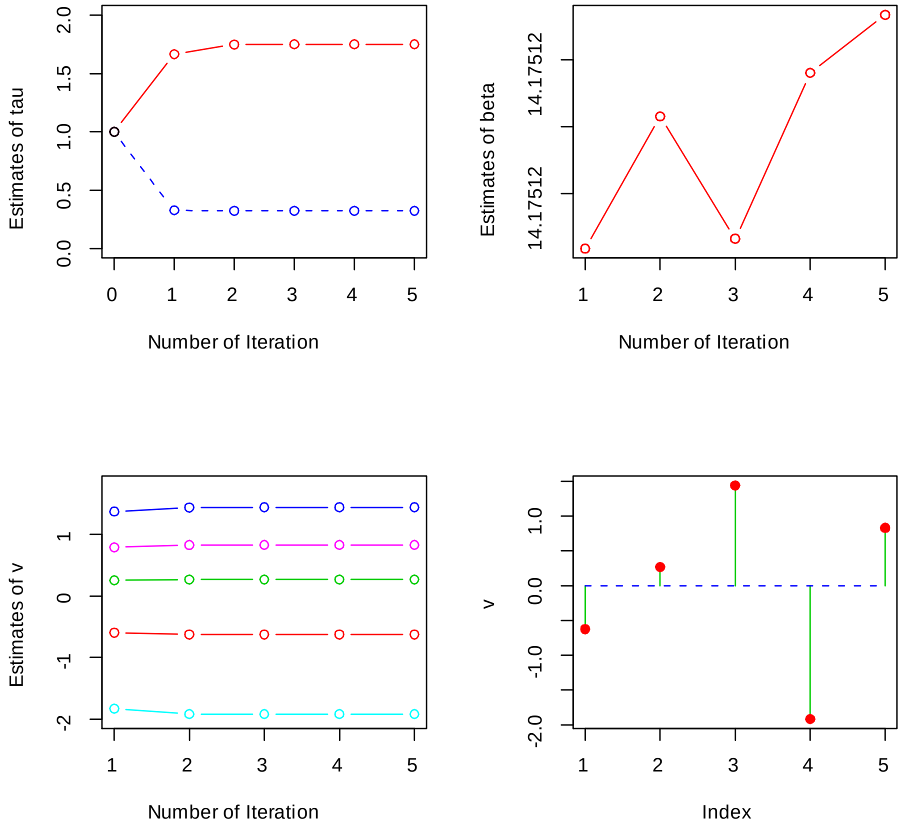

Figure 4.1 Estimation and iteration for estrone measurements data. In the first subfigure, starting values are set to be 1. Blue dashed line and red single line refer to $\widehat{\sigma}^2$ and $\widehat{\sigma}_v^2$, respectively.

Actually, $\overline{y} = 14.1751$ which is exactly the same as $\widehat{\mu}$. We also have the sample means $\overline{y}_i$'s as

13.5450 14.4473 15.6350 12.2331 15.0151

and the variance of these five sample means is 1.7698. And that according to our estimation results, the estimates of the individual means $\widehat{\mu}_i$'s should be

13.5522 14.4442 15.6182 12.2555 15.0054

with the estimate $\widehat{\sigma}_v^2 = 1.7494$. The estimates of individual means are quite close to the sample means and the estimate of the variance component σ_v^2 is close to the variance of the five sample means as well. The original response data set $\mathbf{y}$ and the estimated $\widehat{\mathbf{y}}$ are both shown in Figure 4.2.

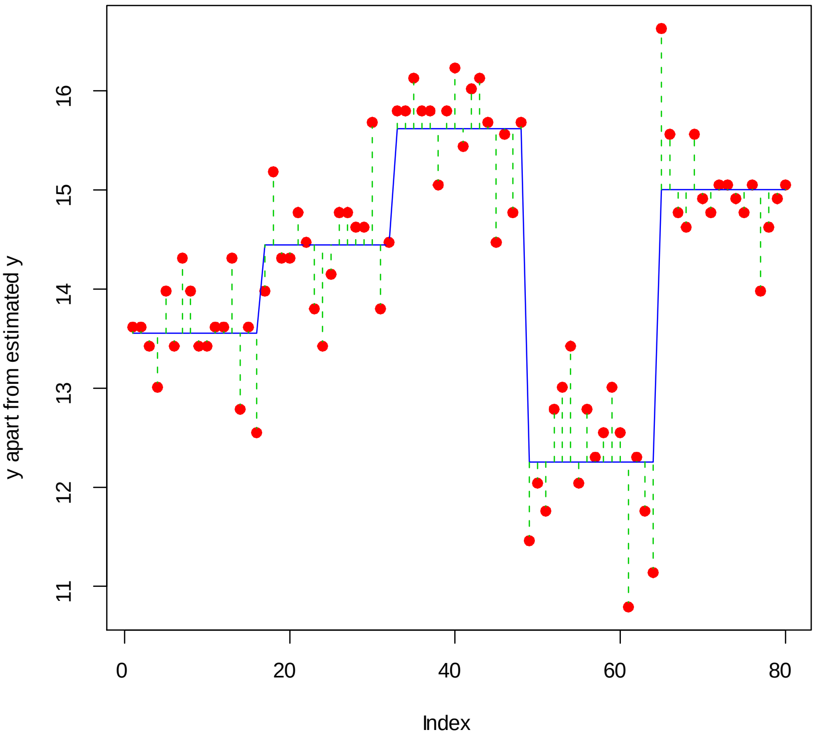

Figure 4.2 Original response data set and the estimates.
The red solid points refer to the original data in y, and the blue line represents the estimates.

These results show a good estimation of our algorithm, and the ratio of $\widehat{\sigma}_v^2/(\widehat{\sigma}^2 + \widehat{\sigma}_v^2) = .8432$ is a large value. From the LRT statistic in the algorithm, we have the statistic value of $S_\lambda = 326.17$ by substituting ℓ_0 with $p_{\beta,\mathbf{v}}(h|\boldsymbol{\tau}_0)$, where $\widehat{\boldsymbol{\tau}}_0 = (\widehat{\sigma}^2, 0)$. Assuming the sample size 80 is large enough, we can approximately find the distribution of S_λ. As shown by Self, S.G. and Liang, K.Y. (1987), S_λ should be 50% : 50% mixture distributed between 0 and a χ^2 distribution. The difference in dimensionality of $\widehat{\boldsymbol{\tau}}_0$ and $\widehat{\boldsymbol{\tau}}$ is 1 which should be the degrees of freedom of the χ^2 distribution. Denoting the mixture distribution as M, we have $S_\lambda = 326.17 \gg M_{.95} = \chi^2_{(.95-.50)/.50}(1) = \chi^2_{.90}(1) = 2.71$, which indicates most of the variability among the measurement data comes from the person effects, say, different menopausal women may have different content of estrone. To check the assumptions of $\mathbf{v}$ and $\mathbf{e}$ separately, the two sets of residuals $\widehat{v}_i$ and $\widehat{e}_i = y_i - \mathbf{X}_i\widehat{\boldsymbol{\beta}} - \mathbf{Z}_i\widehat{\mathbf{v}}$ are considered. The fitting procedure gives the standardization of $\widehat{v}_i/(1 - q_{Mi})$ and $\widehat{e}_i/(1 - q_i)$. Figure 4.3 gives some graphs about standard $\widehat{\mathbf{e}}$ residuals[iv] and the corresponding deviance residuals.

[iv] To avoid some unwanted trend, Lee, Y. and Nelder, J.A. (2001) recommend a plot of $\widehat{e}_i$ against $\mathbf{X}_i\widehat{\boldsymbol{\beta}}$.

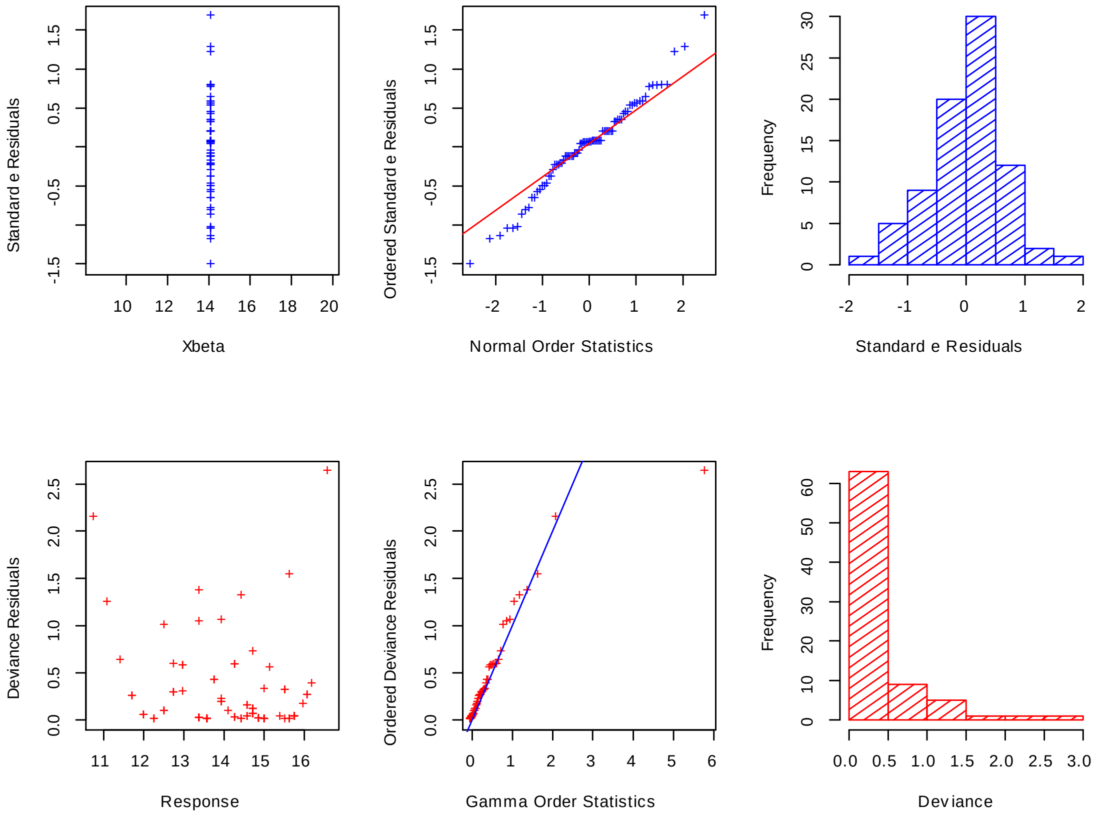

Figure 4.3 Residual plots for estrone measurements data

The first row of subfigures in Figure 4.3 indicate the symmetry from assumptions and approximately normality of the residual set $\widehat{e}_i$. Referring to the deviance residuals, as our definition of deviance, they should be positive and skewed to the right. In fact, in our REML algorithm, gamma GLM is applied to fit the deviance.

5 Genetic Data Experiment

In this section, the algorithm we have implemented previously will be applied to some data in genetics. We shall study how this algorithm works for larger design matrices and how the estimates converge.

5.1 Background

Nowadays in genetics, it is necessary to understand the complicated traits which are controlled by lots of genes and other factors. And there is a useful statistical tool dealing with such a problem - Quantitative Trait Loci (QTL) mapping. The core of this method is using genetic markers to trace the inheritance procedure of alleles through a pedigree. Then relating the phenotypes with different alleles, the allelic effects are estimated from the gene flow through the pedigree. Giving the statistical results and evidence, the position on the chromosome where a QTL is most likely located can be inferred.

In QTL analysis, the genetic effects are assumed random, since the founders of the population are assumed to have effective QTL alleles drawn from some distribution of alleles effects in the entire population, and the inheritance process of the alleles transmitting from ancestor to descendent is random. The model is so-called infinite alleles model when the random effects are assumed to be sampled from a multivariate normal distribution with an infinite number of alleles. It has been shown by simulation

[v] A locus at which there are two possible variations of a given DNA sequence that are detectable in the human population. (www.nature.com/nrg/journal/v6/n10/glossary/nrg1686_glossary.html)

that this model can give unbiased estimates also when the QTL is biallelic[v].

A between-individual covariance structure has to be specified if we want to estimate the variance component of the QTL effects. Using some methods to estimate from marker information, we will obtain a matrix called the identity-by-descent (IBD) matrix. Rönnegård, L. and Carlborg, Ö. (2007) described about this background in detail. Then we have the data with the form of IBD matrices, other fixed and random effects design matrices and the trait vector as response. We will model these data with VC models in the next subsection.

5.2 Data Modeling

The data we will model in this subsection was also analyzed with Henderson's method 3 by Rönnegård, L., Al-Sarraj, R. and von Rosen, D. (2007). We have the response vector **y** which include phenotypes of 190 observations of F_2[vi] pigs. What we are interested in is the chromosomal position of the halothane[vii] gene affecting meat quality. The design matrix **X** of fixed effects contains 6 columns, where column 1 indicates the gender, columns 2 to 5 indicate the batch the observations come from and the last column indicates the body weight in grams. The design matrix $\mathbf{Z}_f$ which include family effects in 26 columns. What's more, we are supposed to have the design matrix **Z** of QTL effects. It is decomposable from the IBD matrix $\mathbf{\Pi}$. Observing that the distribution of **y** is generally normal[viii], we have

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}_f\mathbf{v} + \mathbf{Z}\mathbf{a} + \mathbf{e} \tag{5.1}$$

where $\boldsymbol{\beta}$ is the fixed-effect vector, **v** is the family random-effect vector with covariance matrix $\mathbf{D} = \sigma_v^2\mathbf{I}_{26}$, **a** is the QTL random-effect vector with covariance matrix $\mathbf{D}_a = \sigma_a^2\mathbf{I}_{190}$, and the error term **e** has the covariance matrix $\mathbf{\Sigma} = \sigma^2\mathbf{I}_{190}$. However, the algorithm implemented in the previous section is available for the VC model containing only one random-effect term. Thus, we will first stepwise analyze the random effects.

Considering only the family effects and the fixed effects, we have the model

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}_f\mathbf{v} + \mathbf{e} \tag{5.2}$$

which can be estimated using the previous fitting algorithm. After estimating this model, we get the estimates $\widehat{\sigma}_v^2 = .97$ and $\widehat{\sigma}^2 = 17.63$. The estimates and process of iteration is shown in Figure 5.1, and the residual plots are given in Figure 5.2.

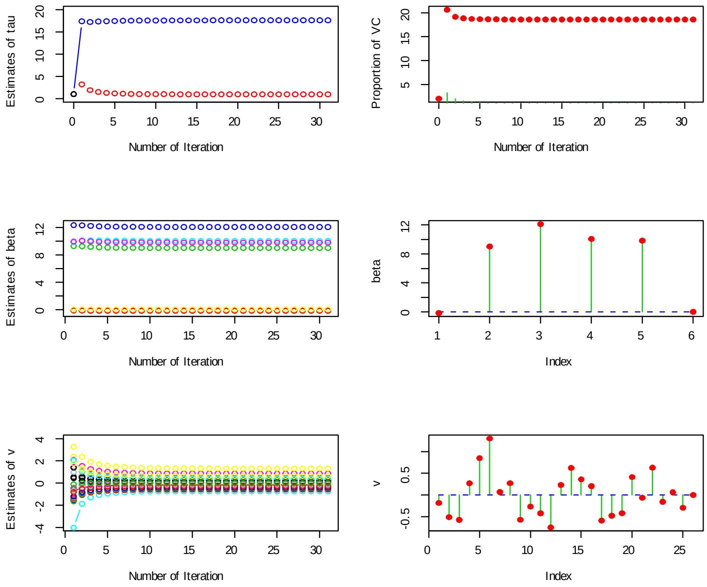

Figure 5.1 Estimation and iteration for the pigs' phenotypes data including fixed effects and family effects

[vi] Usually, F_0 and F_1 are used to denote grandparents and parents, respectively. F_2 represents offsprings.

[vii] A nonflammable inhalation anesthetic that produces general anesthesia; used along with analgesics and muscle relaxants for many types of surgical procedures (wordnet.princeton.edu/perl/webwn)

[viii] Histogram of **y** is given in Appendix.

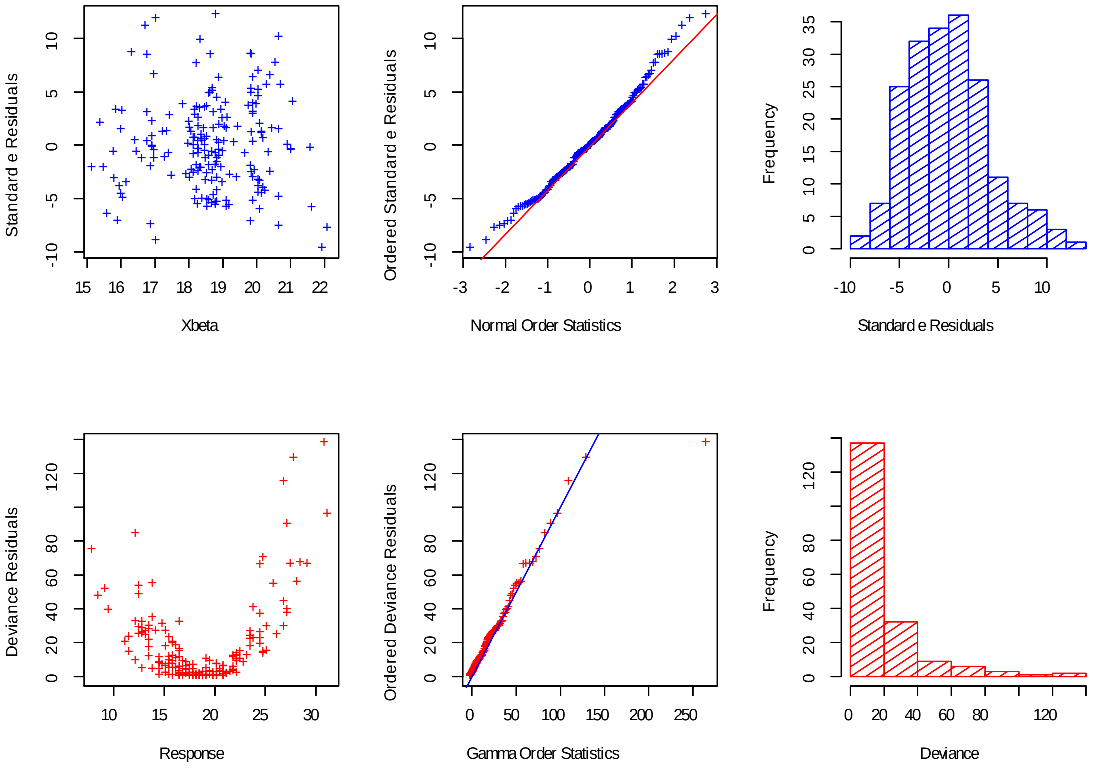

Figure 5.2 Residual plots for pigs' phenotypes data including fixed effects and family effects

In Figure 5.1, the second and third rows of subfigures give the iteration procedures and estimates of $\boldsymbol{\beta}$ and $\mathbf{v}$. Now focus on the subfigures in the first row. Red solid points in the second subfigure give the convergence of the total estimated variance in the response vector, i.e. $\widehat{\sigma}_v^2 + \widehat{\sigma}^2$; the green vertical bars at the bottom show the quantity of $\widehat{\sigma}_v^2$ out of the total estimated variance. This indicates that the variance component from the family effects is quite small, therefore ignoring such effects in the model (5.1) should not affect our estimation too much. Furthermore, we can compare the estimated y_i's with the original ones in Figure 5.3, where the estimated line does not fit the data well, and the variance included in the line mainly comes from the term $\mathbf{e}$ instead of the random family effects term $\mathbf{v}$.

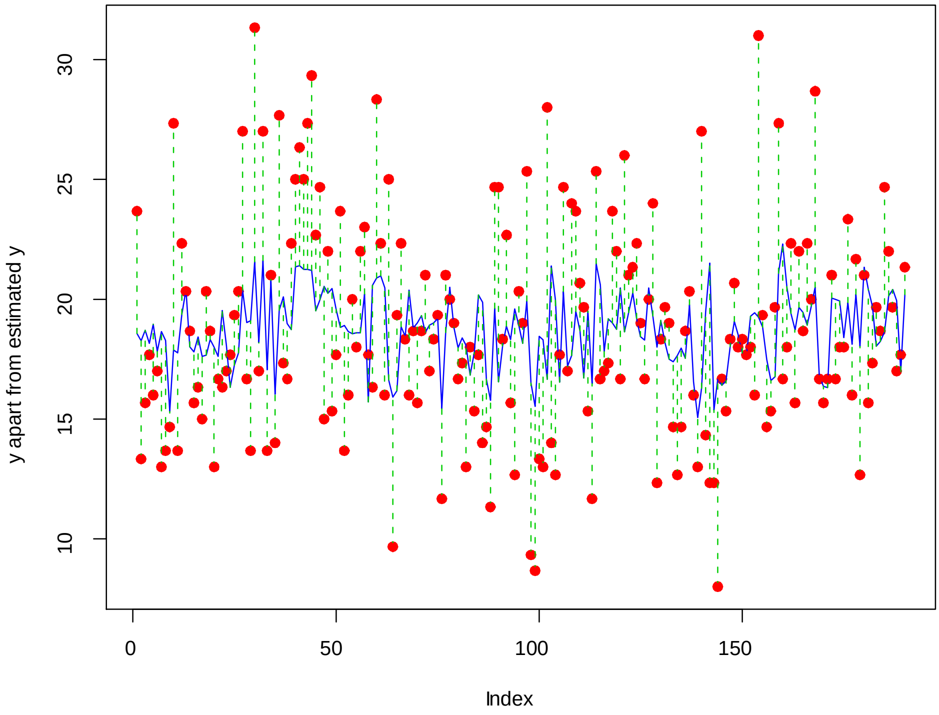

Figure 5.3 Data in response vector (solid points) and their estimates (solid line) for pigs' phenotypes data including fixed effects and family effects

Hence we may include only one random-effect term of the QTL effects instead of the family effects, which gives the model

$$\mathbf{y} = \mathbf{X}\beta + \mathbf{Z}\mathbf{a} + \mathbf{e} \tag{5.3}$$

Similarly, we should be able to estimate this model by using Algorithm 4.1 again, and the difference is that now we have more elements - 190 different genes - in the random-effect vector **a**. Before estimating model in (5.3), we should deal with the problem of decomposing the design matrix **Z** from the IBD matrix **Π**. The IBD matrix describes the covariance structure between each pair of individuals, and we have the relationship

$$\mathbf{\Pi} = \mathbf{Z}\mathbf{Z}' \tag{5.4}$$

Thus, decomposing matrix **Z** requires some methods finding square roots of a square matrix, and in our analysis, the singular value decomposition (SVD) method is applied. The following algorithm gives the process how this method is implemented.

Algorithm 5.1 *Decomposing design matrix* **Z** *from IBD matrix* **Π** *by SVD method:*

- *Decompose* **Π** *by SVD method*[ix] *to obtain*

$$\mathbf{\Pi} = \mathbf{U}\mathbf{D}\mathbf{V}' \tag{5.5}$$

where **U** *and* **V** *are orthogonal, and* **D** *is a diagonal matrix with the singular values* d_{ii}, $i = 1, 2, \cdots, k$.

- *Denote*

$$\sqrt{\mathbf{D}} \equiv diag\left(\sqrt{d_{11}}, \sqrt{d_{22}}, \cdots, \sqrt{d_{kk}}\right) \tag{5.6}$$

Calculate

$$\mathbf{Z} = \mathbf{U}\sqrt{\mathbf{D}}\mathbf{V}' \tag{5.7}$$

[ix] Function {svd} in R can do this decomposition. (For help see http://stat.ethz.ch/R-manual/R-patched/library/base/html/svd.html)

[x] The 35 different positions represented by length in centimeters are: 0cM, 5cM, ···, 170cM. In genetics, a centimorgan (abbreviated cM) or map unit (m.u.) is a unit of recombinant frequency for measuring genetic linkage. It is often used to imply distance along a chromosome. For human, 1 centimorgan on average represents a distance of about 7.5 × 10^5 base pairs. (See Scott, M.P. et al., 2004)

Selecting the F_2 individuals which have been phenotyped, we obtain the 35 IBD matrices at 35 different positions^x on pig's chromosome 6. Combining both algorithms 4.1 and 5.1, we can estimate the model in (5.3). 35 times of applying the algorithms give 35 pairs of estimates $(\widehat{\sigma}^2, \widehat{\sigma}_a^2)$, which are the results we need for inference in QTL analysis. To show a plain estimation procedure, we still give a group of figures as before, taking the case of IBD matrix at 80cM for instance. The second subfigure in the first row of Figure 5.4 gives the proportion of $\widehat{\sigma}_a^2$ out of the total variance $\widehat{\sigma}^2 + \widehat{\sigma}_a^2$, where we can find big difference from that in Figure 5.1. Here the QTL effects instead of the family effects in the model, we obtain a larger variance component from the random effects. This is also confirmed in Figure 5.6. The variation of the estimated line is larger than the line in Figure 5.3. Since we have indicated that the variation of the line in Figure 5.3 is mainly due to σ^2, and here we have more variation, the difference of variation should come from the variance component σ_a^2.

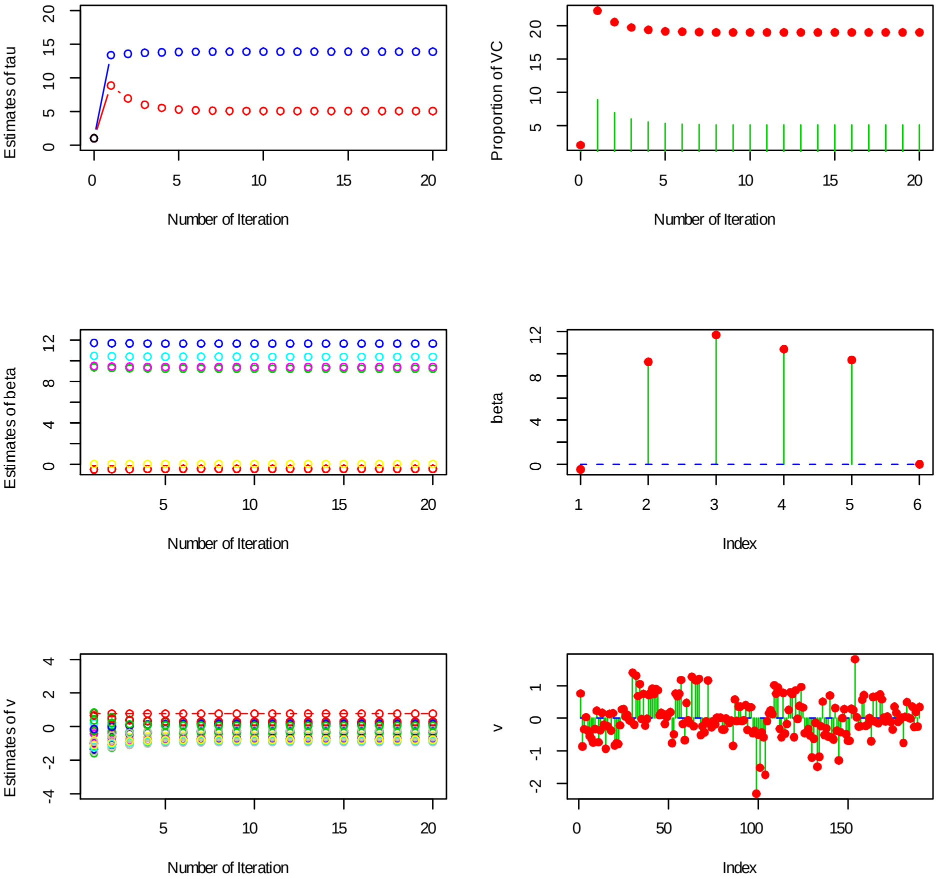

Figure 5.4 Estimation and iteration for the pigs' phenotypes data including fixed effects and QTL effects at 80cM on pig chromosome 6

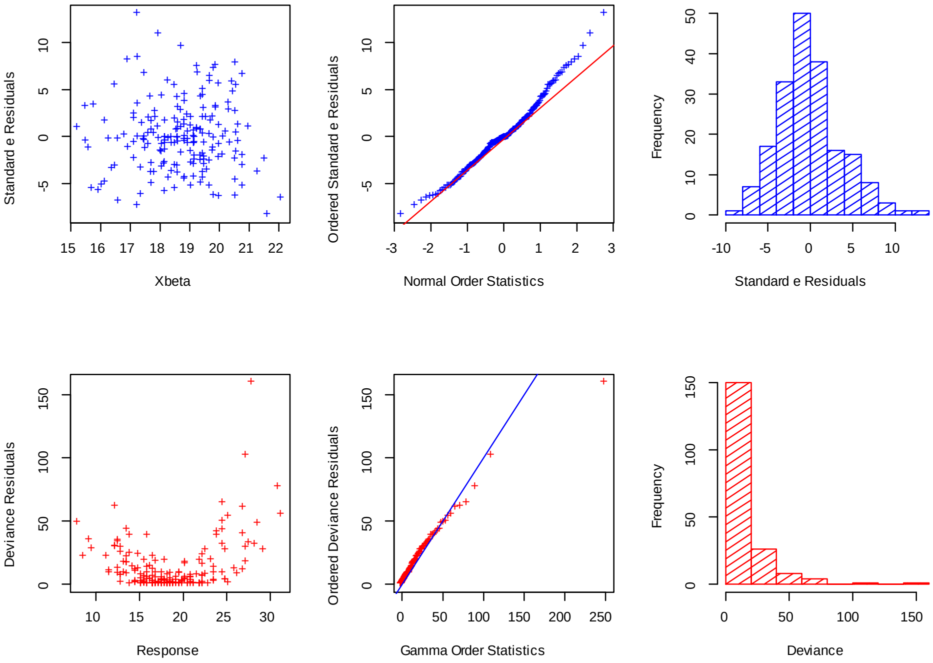

Figure 5.5 Residual plots for pigs' phenotypes data including fixed effects and QTL effects at 80cM on pig chromosome 6

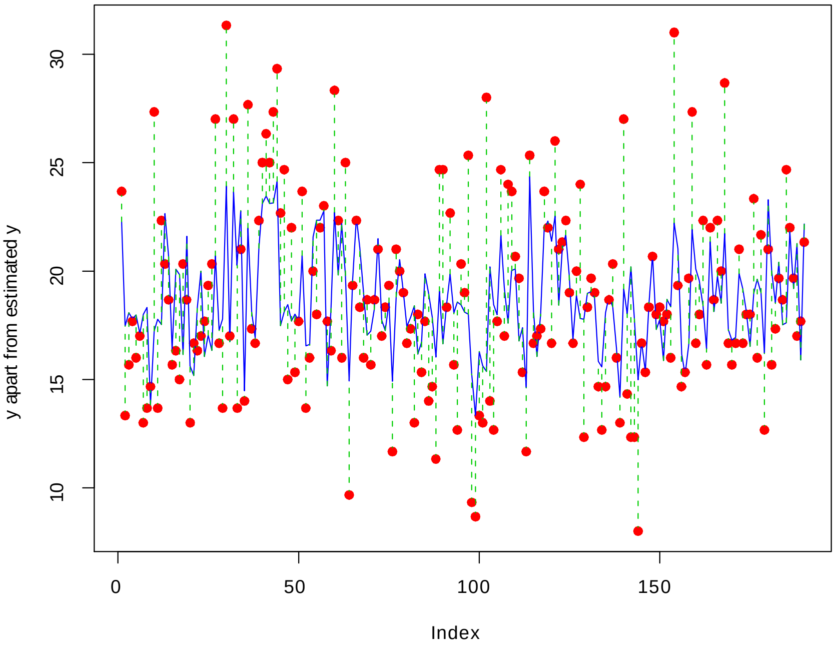

Figure 5.6 Data in response vector (solid points) and their estimates (solid line) for pigs' phenotypes data including fixed effects and QTL effects at 80cM on pig chromosome 6.

Estimation results of the 35 models are summarized in Table 5.1 and Figure 5.8-5.9. A significantly larger LRT statistic value indicates a position that the gene most likely locates at.

Table 5.1 Estimates of variance components for pigs' phenotypes data including fixed effects and QTL effects

Position	$\widehat{\sigma}^2$	$\widehat{\sigma}_a^2$	Position	$\widehat{\sigma}^2$	$\widehat{\sigma}_a^2$
0cM	16.20	3.95	90cM	14.19	6.26
5cM	17.05	2.53	95cM	15.27	4.87
10cM	17.98	.78	100cM	16.10	3.63
15cM	17.96	.83	105cM	15.47	4.35
20cM	18.02	.77	110cM	14.81	5.24
25cM	18.29	.36	115cM	15.44	4.11
30cM	18.55	.00	120cM	16.47	2.45
35cM	18.56	.00	125cM	16.11	2.36
40cM	18.56	.00	130cM	15.47	3.14
45cM	18.56	.00	135cM	14.90	4.32
50cM	18.56	.00	140cM	14.22	5.75
55cM	18.55	.00	145cM	13.81	6.64
60cM	16.58	3.35	150cM	13.43	7.56
65cM	14.83	6.74	155cM	13.46	7.75
70cM	14.14	6.51	160cM	13.99	7.05
75cM	14.05	5.81	165cM	14.80	5.85
80cM	13.92	5.07	170cM	15.57	4.72
85cM	14.40	5.21			

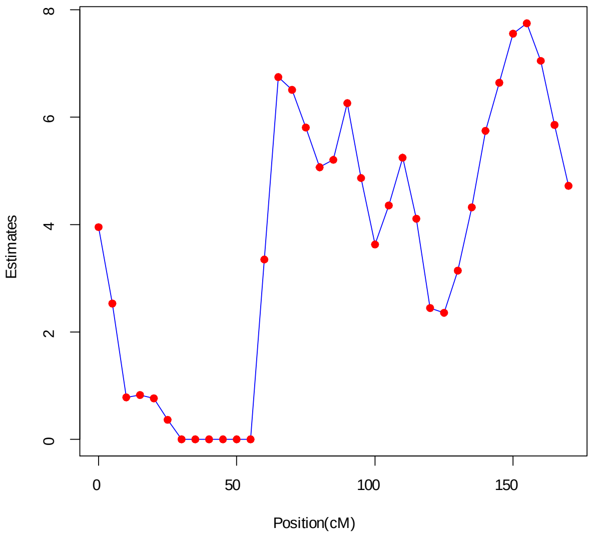

Figure 5.7 Estimates of QTL variance component σ_a^2 for pigs' phenotypes data along pig chromosome 6 (model including fixed effects and QTL effects)

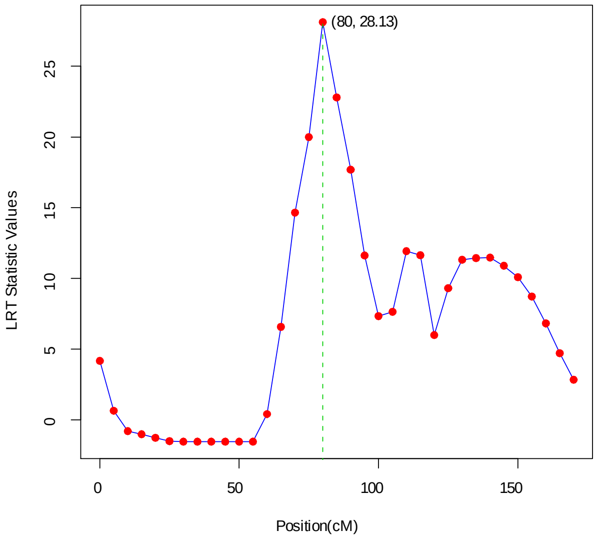

Figure 5.8 Values of likelihood ratio statistics for pigs' phenotypes data along pig chromosome 6 (model including fixed effects and QTL effects). The halothane gene affecting meat quality is located at 80cM.

Notice that in our experiment of pigs' phenotypes data, for LRT, likelihood under the null hypothesis is set as the same all over the chromosome. And this likelihood is determined from the model in (5.2), where none of the QTL effects are involved. The same likelihood under the null hypothesis is applied in Section 7 as well.

We can make a reasonable inference from Figure 5.8. The likelihood ratio curve peaks at the position of 80cM, which gives that the most likely position of the gene is somewhere around here. In fact, the halothane gene really locate here, therefore our QTL analysis gives a good evidence. In Discussion, we will mention the judgement about significance of the gene position.

6 Algorithm Comparison

In this subsection we compare the HGLM method to Fisher Scoring (FS) method. We are interested in two aspects. One is the statistical characteristic of this method comparing to others, and the other is the efficiency of convergence for the algorithm.

6.1 Fisher Scoring Method

We give a summary of the FS method. For more details, see Davidson, A.C. (2003). Let $Y_1, Y_2, \cdots, Y_n$ be independently and identically distributed random variables whose p.d.f. $f(Y;\boldsymbol{\theta})$ is twice differentiable. What we desire to find is the maximum likelihood estimator (MLE) $\widehat{\boldsymbol{\theta}}$ of $\boldsymbol{\theta}$. We have the score function

$$\begin{aligned} S(\boldsymbol{\theta}) &= \frac{\partial}{\partial\boldsymbol{\theta}}\ell(\boldsymbol{\theta};y) && (6.1)\\ &= \frac{\partial}{\partial\boldsymbol{\theta}}\log L(\boldsymbol{\theta};Y) = \frac{1}{L(\boldsymbol{\theta};y)}\frac{\partial}{\partial\boldsymbol{\theta}}L(\boldsymbol{\theta};Y) \end{aligned}$$

Setting a starting point $\boldsymbol{\theta}_0$ for the algorithm, we can derive the Taylor expansion of the score function as

$$S(\boldsymbol{\theta}) \approx S(\boldsymbol{\theta}_0) - \mathcal{J}(\boldsymbol{\theta}_0)(\boldsymbol{\theta}-\boldsymbol{\theta}_0) \qquad (6.2)$$

where

$$\mathcal{J}(\boldsymbol{\theta}_0) = -\sum_{i=1}^{n} \nabla\nabla'\big|_{\boldsymbol{\theta}=\boldsymbol{\theta}_0} \log f(Y_i;\boldsymbol{\theta}) \qquad (6.3)$$

is the observed information matrix at $\boldsymbol{\theta}_0$. Then setting $\boldsymbol{\theta} = \widehat{\boldsymbol{\theta}}$ and $S(\widehat{\boldsymbol{\theta}}) = 0$, we can derive

$$\widehat{\boldsymbol{\theta}} = \boldsymbol{\theta}_0 + \mathcal{J}^{-1}(\boldsymbol{\theta}_0)S(\boldsymbol{\theta}_0) \qquad (6.4)$$

Therefore under some conditions, it can be shown that the algorithm

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \mathcal{J}^{-1}(\boldsymbol{\theta}_k)S(\boldsymbol{\theta}_k) \qquad (6.5)$$

can bring us the convergence $\boldsymbol{\theta}_k \to \widehat{\boldsymbol{\theta}}$. Replacing $\mathcal{J}(\boldsymbol{\theta})$ by the expected fisher information $\mathcal{I}(\boldsymbol{\theta})$, we have the Fisher Scoring algorithm as

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \mathcal{I}^{-1}(\boldsymbol{\theta}_k)S(\boldsymbol{\theta}_k) \qquad (6.6)$$

It is applied to linear models for maximum likelihood estimation.

6.2 Comparison on Algorithm

For the characteristics of the FS algorithm and the HGLM algorithm, we summarize some aspects in the table below, and these show the important advantages of HGLM.

Table 6.1 Advantages of HGLM compared to FS algorithm

	Fisher Scoring	Hierarchical GLM
Estimation of Variance Components	Force back to a tiny positive ε if convergence at negative region	No problem with the sign since gamma-GLM is applied
Flexibility in Distribution Family	Difficult to apply to non-normal random effects	Changing the algorithm somewhat can fit the whole exponential family

The flexibility in distribution of random effects is the essential advantage of HGLM. Although not implemented in this article, it is convenient to extend our algorithm to other distributions in the exponential family. For instance, VC models is common in QTL analysis and binary data are usually obtained, hence developing REML algorithm for other distribution will then be quite important. Fortunately, hierarchical GLM is flexible for dealing with different distributions. In our study, disadvantages also exist for HGLM algorithm, which is discussed in the next subsection.

6.3 Comparison on Efficiency

The REML algorithm with Fisher Scoring method is implemented as well in Appendix. Setting the convergence criterion and starting value the same[xi] for both the FS and the HGLM algorithms, first we estimate the Fears et al. (1996) data in Section 4 with both methods. The difference in convergence is clearly shown in Figure 6.1.

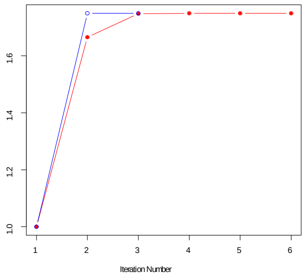

Figure 6.1 Convergence procedure of σ_v^2 for estrone measurements data. Red solid points and blue circles refer to HGLM and FS algorithms, respectively.

The convergence of σ^2 is similar with σ_v^2. According to Figure 6.1, in this example, HGLM takes 5 iterations to get convergence, whereas FS method takes only 2. As shown in the figure, the estimator in HGLM algorithm approaches to the destination value slower. To study this difference further, we do the comparison for model in (5.3)

[xi] Here we set the convergence rule as $\max|\boldsymbol{\tau}_k - \boldsymbol{\tau}_{k-1}| \leqslant 1\times10^{-5}$ (k is the current number of iteration) and $\max(k) = 200$. All the starting values are set to be 1.

with both methods for the IBD matrix at 80cM. The results are shown in Figure 6.2.

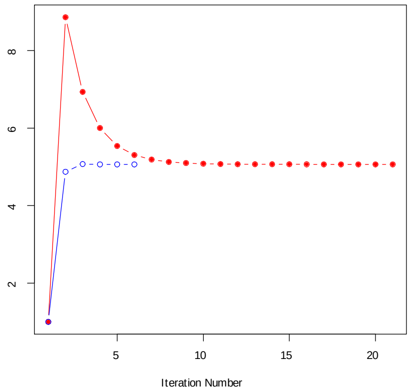

Figure 6.2 Convergence of σ_a^2 for pigs' phenotypes data including fixed effects and QTL effects at 80cM on pig chromosome 6. Red solid points and blue circles refer to HGLM and FS algorithms, respectively.

In Figure 6.2, the FS algorithm takes only 5 iterations but the HGLM algorithm takes even more than quadruple of that. Therefore, we find that the efficiency of convergence for HGLM seems to be quite slow. We summarize the disadvantages of HGLM algorithm in the following table.

Table 6.2 Disadvantages of HGLM compared to FS algorithm

	Fisher Scoring	Hierarchical GLM
Convergence	Efficient	Slow

However, for the slow convergence, there may be some way out. The solid points in Figure 6.2 give us a curve which looks smooth. The destination estimate here appears to be like a limit or an asymptote for the HGLM iteration sequence. Since we find a trend that the curve tend to keep a distance from the estimate we want, we may fit such a convergence curve and use it for some prediction. Therefore, when it takes too many iterations, according to some fitting methods, we may predict the estimate from just a few points at the beginning. Some proposals are given in Discussion.

7 More Random Effects

Now we will try to estimate the model with the form like (5.1). To include two random-effect terms at the same time in the VC models, we should upgrade Algorithm 4.1. We still consider the following augmented linear model, but we need to make some modification.

$$\mathbf{y}_a = \mathbf{T}\boldsymbol{\delta} + \mathbf{e}_a \tag{7.1}$$

where $\mathbf{e}_a \sim MVN(\mathbf{0}, \boldsymbol{\Sigma}_a)$, and

$$\mathbf{y}_a = \begin{pmatrix} \mathbf{y} \\ \psi_{M1} \\ \psi_{M2} \end{pmatrix}, \mathbf{T} = \begin{pmatrix} \mathbf{X} & \mathbf{Z}_1 & \mathbf{Z}_2 \\ \mathbf{0} & \mathbf{I}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I}_2 \end{pmatrix}, \boldsymbol{\delta} = \begin{pmatrix} \boldsymbol{\beta} \\ \mathbf{v}_1 \\ \mathbf{v}_2 \end{pmatrix},$$
$$\mathbf{e}_a = \begin{pmatrix} \mathbf{e} \\ \mathbf{e}_{M1} \\ \mathbf{e}_{M2} \end{pmatrix}, \boldsymbol{\Sigma}_a = \begin{pmatrix} \boldsymbol{\Sigma} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{D}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{D}_2 \end{pmatrix}.$$

Here, $\boldsymbol{\Sigma} = \sigma^2\mathbf{I}_N$, $\mathbf{D}_1 = \sigma_{v_1}^2\mathbf{I}_{q_1}$, $\mathbf{D}_2 = \sigma_{v_2}^2\mathbf{I}_{q_2}$ and $\boldsymbol{\tau} = (\sigma^2, \sigma_{v_1}^2, \sigma_{v_2}^2)$. ψ_{M1} and ψ_{M2} are set to be 0 as responses for estimating the variance component in matrix $\mathbf{D}_1$ and $\mathbf{D}_2$. The following algorithm is constructed through IWLS for the augmented model to estimate $(\boldsymbol{\beta}, \boldsymbol{\tau}, \mathbf{v}_1, \mathbf{v}_2)$.

Algorithm 7.1 *Upgraded IWLS for the augmented linear mixed model:*

- *Set a starting value of variance-component parameter $\boldsymbol{\tau}$.*
- *Given the current estimate of $\boldsymbol{\tau}$, solve the augmented mixed models equation $\mathbf{T}'\boldsymbol{\Sigma}_a^{-1}\mathbf{T}\widehat{\boldsymbol{\delta}} = \mathbf{T}'\boldsymbol{\Sigma}_a^{-1}\mathbf{y}_a$ to get a new estimate of $\boldsymbol{\delta}$:*

$$\widehat{\boldsymbol{\delta}} = \left(\mathbf{T}'\boldsymbol{\Sigma}_a^{-1}\mathbf{T}\right)^{-1}\mathbf{T}'\boldsymbol{\Sigma}_a^{-1}\mathbf{y}_a \tag{7.2}$$

- *Given the current estimate of $\boldsymbol{\delta}$, calculate the deviance components corresponding to $\mathbf{e}_a$ as the squared residuals:*

$$d_i = (y_i - \mathbf{X}_i\widehat{\boldsymbol{\beta}} - \mathbf{Z}_{1i}\widehat{\mathbf{v}}_1 - \mathbf{Z}_{2i}\widehat{\mathbf{v}}_2)^2 \tag{7.3}$$
$$d_{M1i} = (\psi_{M1} - \widehat{v}_{1i})^2 = \widehat{v}_{1i}^2 \tag{7.4}$$
$$d_{M2i} = (\psi_{M2} - \widehat{v}_{2i})^2 = \widehat{v}_{2i}^2 \tag{7.5}$$

where $\mathbf{X}_i$, $\mathbf{Z}_{1i}$ and $\mathbf{Z}_{2i}$ refer to the i'th row vector in $\mathbf{X}$, $\mathbf{Z}_1$ and $\mathbf{Z}_2$.

- *Calculate the hat-matrix:*

$$\mathbf{H}_a = \mathbf{T}\left(\mathbf{T}'\boldsymbol{\Sigma}_a^{-1}\mathbf{T}\right)^{-1}\mathbf{T}'\boldsymbol{\Sigma}_a^{-1} \tag{7.6}$$

and set the leverages:

$$\begin{pmatrix} \mathbf{q} \\ \mathbf{q}_{M1} \\ \mathbf{q}_{M2} \end{pmatrix} = \tag{7.7}$$
$$\begin{pmatrix} (h_{11}, \cdots, h_{NN})' \\ (h_{N+1,N+1}, \cdots, h_{N+q_1,N+q_1})' \\ (h_{N+q_1+1,N+q_1+1}, \cdots, h_{N+q_1+q_2,N+q_1+q_2})' \end{pmatrix}$$

where h_{ii} refers to the i'th diagonal element of $\mathbf{H}_a$.

- *Let*

$$d_i^* = \frac{d_i}{1-q_i} \tag{7.8}$$

$$d_{M1i}^* = \frac{d_{M1i}}{1-q_{M1i}} \tag{7.9}$$

$$d_{M2i}^* = \frac{d_{M2i}}{1-q_{M2i}} \tag{7.10}$$

$$\varpi = \frac{\mathbf{1}-\mathbf{q}}{2} \tag{7.11}$$

$$\varpi_{M1} = \frac{\mathbf{1}-\mathbf{q}_{M1}}{2} \tag{7.12}$$

$$\varpi_{M2} = \frac{\mathbf{1}-\mathbf{q}_{M2}}{2} \tag{7.13}$$

and model a GLM with response $\mathbf{d}^*$, *gamma family*[xii], *identity link, intercept only linear predictor and prior weight* ϖ *to get an updated estimate for* σ^2; *model a GLM with response* $\mathbf{d}^*_{M1}$, *gamma family, identity link, intercept only linear predictor and prior weight* ϖ_{M1} *to get an updated estimate for* $\sigma^2_{v_1}$; *model a GLM with response* $\mathbf{d}^*_{M2}$, *gamma family, identity link, intercept only linear predictor and prior weight* ϖ_{M2} *to get an updated estimate for* $\sigma^2_{v_2}$. *Note that* $E(d_i^*) = \sigma^2$, $Var(d_i^*) = 2\sigma^2/(1-q_i)$, $E(d_{M1i}^*) = \sigma^2_{v_1}$, $Var(d_{M1i}^*) = 2\sigma^2_{v_1}/(1-q_{M1i})$ *and* $E(d_{M2i}^*) = \sigma^2_{v_2}$, $Var(d_{M2i}^*) = 2\sigma^2_{v_2}/(1-q_{M2i})$.

- *Iterate from the second step until convergence. At convergence, calculate standard errors of* $\widehat{\boldsymbol{\beta}}$ *and* $\widehat{\mathbf{v}}-\mathbf{v}$ *from* $\mathbf{H}^{-1}$, *where*

$$\mathbf{H} = \mathbf{T}'\boldsymbol{\Sigma}_a^{-1}\mathbf{T} \tag{7.14}$$

After checking singularity, calculate the standard errors of $\widehat{\boldsymbol{\tau}}$ *from the negative inverse of the Hessian*[xiii] *of*

$$p_{\boldsymbol{\beta},\mathbf{v}_1,\mathbf{v}_2}(h|\boldsymbol{\tau}) = \ell(\widehat{\boldsymbol{\beta}}_{\boldsymbol{\tau}},\boldsymbol{\tau}) - \frac{1}{2}\log\left|\frac{\mathbf{X}'\mathbf{V}^{-1}\mathbf{X}}{2\pi}\right| \tag{7.15}$$

where $\mathbf{V} = \mathbf{Z}_1\mathbf{D}_1\mathbf{Z}_1' + \mathbf{Z}_2\mathbf{D}_2\mathbf{Z}_2' + \boldsymbol{\Sigma}$ *and*

$$\ell(\boldsymbol{\beta},\boldsymbol{\tau}) = -\frac{1}{2}\log|2\pi\mathbf{V}| - \frac{1}{2}(\mathbf{y}-\mathbf{X}\boldsymbol{\beta})'\mathbf{V}^{-1}(\mathbf{y}-\mathbf{X}\boldsymbol{\beta}) \tag{7.16}$$

- *Calculate the likelihood ratio test (LRT) statistic of test, for instance,*

$$\begin{aligned} H_0 &: \sigma^2_{v_2} = 0 \\ H_1 &: \sigma^2_{v_2} > 0 \end{aligned}$$

in accordance with the profile likelihood in (7.15), we have

$$S_\lambda = -2\left(\ell_0 - p_{\boldsymbol{\beta},\mathbf{v}_1,\mathbf{v}_2}(h|\widehat{\boldsymbol{\tau}})\right) \tag{7.17}$$

where ℓ_0 *is the likelihood under the null hypothesis.*

The upgraded algorithm can be applied for estimating the model in (5.1) and the results from the QTL analysis is given in Table 7.1 and in Figures 7.1-7.2.

[xii] We can obtain REML estimators from fitting gamma GLMs.

[xiii] The Hessian matrix of a function at a certain value of an argument can be calculated using the Richardson approximation. Package {`numDeriv`} in R can do this approximation (for help see http://rss.acs.unt.edu/Rdoc/library/numDeriv/html/hessian.html).

Table 7.1 Estimates of variance components for pigs' phenotypes data including fixed effects, family effects and QTL effects

Position	$\widehat{\sigma}^2$	$\widehat{\sigma}_v^2$	$\widehat{\sigma}_a^2$
0cM	16.20	.02	3.92
5cM	16.95	.56	1.64
10cM	17.36	.87	.49
15cM	17.41	.90	.41
20cM	17.54	.94	.17
25cM	17.63	.97	.00
30cM	17.63	.97	.00
35cM	17.63	.97	.00
40cM	17.63	.97	.00
45cM	17.63	.97	.00
50cM	17.63	.97	.00
55cM	17.46	1.04	.17
60cM	15.60	1.39	3.04
65cM	14.52	1.06	5.29
70cM	14.01	.37	6.08
75cM	13.96	.18	5.66
80cM	13.80	.16	5.03
85cM	13.88	.71	4.95
90cM	13.65	.82	5.87
95cM	14.86	.58	4.70
100cM	15.91	.31	3.45
105cM	15.37	.24	4.14
110cM	14.64	.31	5.07
115cM	15.00	.59	4.03
120cM	15.63	1.01	2.42
125cM	15.20	.94	2.69
130cM	14.61	.90	3.49
135cM	14.16	.83	4.58
140cM	13.75	.68	5.70
145cM	13.57	.50	6.39
150cM	13.37	.31	7.22
155cM	13.46	.16	7.52
160cM	13.99	.07	6.94
165cM	14.80	.06	5.75
170cM	15.56	.16	4.48

To compare with the estimates from model in (5.3), the results in Section 5 are also presented in Figures 7.1 and 7.2.

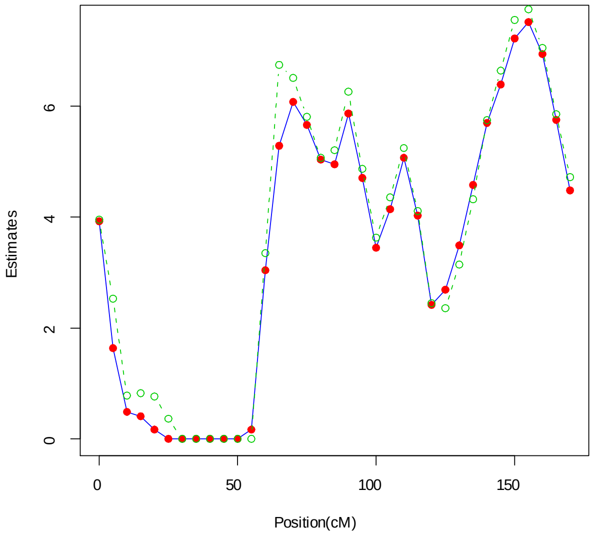

Figure 7.1 Estimates of QTL variance component σ_a^2 for pigs' phenotypes data along pig chromosome 6 (model including fixed effects, family effects and QTL effects). Dashed line with circles refers to Figure 5.7.

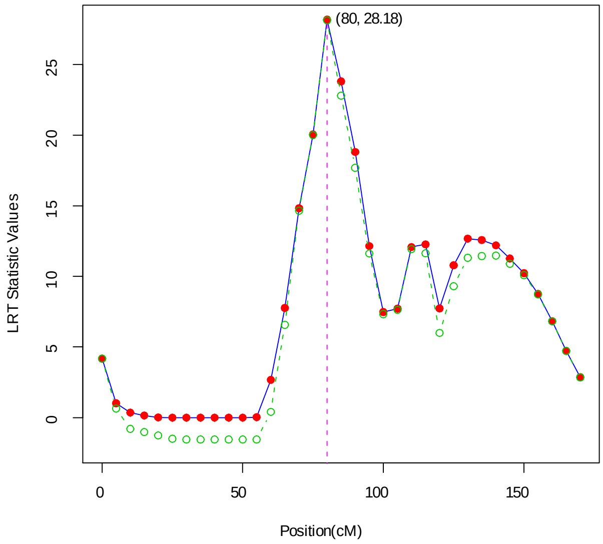

Figure 7.2 Values of likelihood ratio statistics for pigs' phenotypes data along pig chromosome 6 (model including fixed effects, family effects and QTL effects). The halothane gene affecting meat quality is located at 80cM. Dashed line with circles refers to Figure 5.8.

Difference between each two curves in both Figure 7.1 and 7.2 is due to the omission of family effects in (5.3). Estimates and the values of LRT statistics in both models are quite close. The trend of LRT statistic curve leads to similar inference for the gene position.

8 Discussion

As mentioned in the article, other cases of HGLMs like those in Table 3.1 can also be carried out by modify parts of the implemented algorithm. Indicated by Lee, Y., Nelder, J.A. and Pawitan, Y. (2006), for other cases, we still have the augmented model with response $\left(\mathbf{y}', \psi_M'\right)'$, where $E\left(\mathbf{y}\right) = \boldsymbol{\mu}$, $Var\left(\mathbf{y}\right) = \phi V\left(\boldsymbol{\mu}\right)$, $E\left(\psi_M\right) = \mathbf{u}$ and $Var\left(\psi_M\right) = \lambda V_M\left(\mathbf{u}\right)$, and the augmented linear predictor $\boldsymbol{\eta}_{Ma} = \left(\boldsymbol{\eta}', \boldsymbol{\eta}_M'\right)' = \mathbf{T}_M \boldsymbol{\omega}$, where $\boldsymbol{\eta} = g\left(\boldsymbol{\mu}\right) = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\mathbf{v}$, $\boldsymbol{\eta}_M = g_M\left(\mathbf{u}\right) = \mathbf{v}$ and $\boldsymbol{\omega} = \left(\boldsymbol{\beta}', \mathbf{v}'\right)'$. The augmented model matrix is

$$\mathbf{T}_M = \begin{pmatrix} \mathbf{X} & \mathbf{Z} \\ \mathbf{0} & \mathbf{I} \end{pmatrix} \tag{8.1}$$

We will have $V_M() = V()$ if the HGLMs are conjugate, i.e. integrating out the scale parameters. For different GLM families, modification should be done in the variance function and the link function. In the IWLS algorithm, the model equation is

$$\mathbf{T}_M' \boldsymbol{\Sigma}_M^{-1} \mathbf{T}_M \widehat{\omega} = \mathbf{T}_M' \boldsymbol{\Sigma}_M^{-1} \mathbf{z}_{Ma} \tag{8.2}$$

where $\mathbf{z}_{Ma} = \left(\mathbf{z}', \mathbf{z}_M'\right)'$ and $\boldsymbol{\Sigma}_M = \boldsymbol{\Gamma}_M \mathbf{W}_{Ma}^{-1}$ with $\boldsymbol{\Gamma}_M = diag\left(\boldsymbol{\Phi}, \boldsymbol{\Lambda}\right)$, $\boldsymbol{\Phi} = diag\left(\phi_i\right)$ and $\boldsymbol{\Lambda} = diag\left(\lambda_i\right)$. The dependent variables $z_{Mai} = \left(z_i, z_{Mi}\right)$ for data $\left(y_i, \psi_M\right)$ are defined by

$$z_i = \eta_i + \left(y_i - \mu_i\right) \frac{\partial \eta_i}{\partial \mu_i} \tag{8.3}$$

$$z_{Mi} = v_i + \left(\psi_M - u_i\right) \frac{\partial v_i}{\partial u_i} \tag{8.4}$$

with iterative weight matrix

$$\mathbf{W}_{Ma} = diag\left(\mathbf{W}_{M0}, \mathbf{W}_{M1}\right) \tag{8.5}$$

where

$$\mathbf{W}_{M0i} = \left(\frac{\partial \mu_i}{\partial \eta_i}\right)^2 V\left(\mu_i\right)^{-1} \tag{8.6}$$

$$\mathbf{W}_{M1i} = \left(\frac{\partial u_i}{\partial v_i}\right)^2 V_M\left(u_i\right)^{-1} \tag{8.7}$$

Now that estimating VC models by using HGLM is available, we may notice the advantages about model checking as well. HGLMs fit the random effects with a generalized linear model, and during the estimation procedure, we obtain both residual sets of e_i and v_i. So that checking the distribution of the response variable as well as that of random effects will be easy. Especially when fitting a non-normal HGLM, if the sample size of random effects is large enough, we can not only fit a non-normal distribution of random effects but also simply check how are the estimated v_i residuals distributed.

In QTL analysis, if we don't actually realize where the gene is, we should give stronger evidence to show the significance of the peak. This is not that easy. There is no mature method to carry out a statistical test for testing such a problem in accordance with the figure. One way of solving this is Monte-Carlo simulation, but it takes time. Considering the null hypothesis that no significant gene exists along the chromosome, we can generate data from a null model, say the model in (5.2), and estimate for many times. After obtaining a figure like Figure 5.8, by comparing them we can construct a statistic to test whether they are significantly different from each other. If significant, we can then believe such a peak and find the gene. We will not give the test by simulation, but we can generally judge the peak from some other aspects. Relatively above all, the peak is obviously higher than any other parts of the curve. Since the points near this position have a trend of "climbing up" to the peak, we can regard that the figure should not be randomly generated but meant to be like this. Sometimes, these can also give us some evidence to notice such a peak in QTL analysis.

We discovered the poor convergence efficiency of the HGLM algorithm, and to propose a way overcoming this weakness, we consider an example from the article again. We denote data of the HGLM convergence curve in Figure 6.2 to be $\boldsymbol{\alpha} = \left(\alpha_0, \alpha_1, \cdots, \alpha_n\right)$, then α_i should be the value of the variance component σ_a^2 after i iterations with the initial value 1. Let $\Delta_i = \alpha_{i+1} - \alpha_i$ be the difference between each two consecutive values, and let $\phi_i = \Delta_{i-1} / \Delta_i$. Then we obtain the results in Table 8.1.

Table 8.1 Values for convergence analysis

Iteration No.	α_i	Δ_i	ϕ_i
0	1.000000	7.862563	
1	8.862563	−1.927950	−4.078200
2	6.934614	−0.931480	2.069771
3	6.003134	−0.462891	2.012310
4	5.540243	−0.233007	1.986596
5	5.307236	−0.118058	1.973659
6	5.189178	−0.060024	1.966860
7	5.129154	−0.030573	1.963286
8	5.098581	−0.015587	1.961426
9	**5.082994**	**−0.007951**	**1.960467**
10	**5.075043**	−0.004057	1.959974
11	5.070986	−0.002070	1.959722
12	5.068916	−0.001056	1.959593
13	5.067860	−0.000539	1.959527
14	5.067321	−0.000275	1.959493
15	5.067046	−0.000140	1.959476
16	5.066905	−0.000072	1.959467
17	5.066834	−0.000037	1.959463
18	5.066797	−0.000019	1.959460
19	5.066779	−0.000010	1.959459
20	5.066769		

By observing the ϕ_i values, we can assume it to be a constant for all iterations since it varies rather small after some iterations, say $\phi_i = \phi$. Thus we have $\Delta_{i-1}/\Delta_i = \phi$ for all iterations, from which we can derive the result[xiv]

$$\widehat{\alpha} = \alpha_\infty = \alpha_i + \Delta_i \left(1 - \phi^{-1}\right)^{-1} \qquad (8.8)$$

for all i. Since the ϕ_i values do not get stable in the beginning iterations, we may not use the values like α_0 and Δ_0. And choosing a ϕ value really affect the prediction. To eliminate the iteration number in estimation procedure, for instance, we can force the iteration number to be no more than 10. This means, given the convergence criterion, if the iterations converge within 10 times, the final value of the iterations will be regarded as the estimate; whereas if the iterations are not able to converge at the tenth one, we will predict the estimate by substituting $i = 9$ into Equation (8.8) and taking $\phi = \phi_9$. For data in Table 8.1, considering the line in bold, we have $\widehat{\alpha} = \alpha_9 + \Delta_9 \left(1 - \phi_9^{-1}\right)^{-1} = 5.066765$. The true estimate of the variance component is 5.066769. Hence the predicting value is quite close to the true value.

Naturally, there is discredit about such a prediction, i.e. the prediction becomes meaningless if the iterations actually diverge! In fact, in some cases, we may have already realized that the iterations must converge. However, sometimes we still need some method to judge convergence before the algorithm. Schaeffer, L.R. (1979) introduced the Common Intercept Approach (CIA) which is shown to be equivalent to assuming a nonlinear model for the convergence curve. This method can reduce the required number of iterations to attain convergence, but it can also be used for determining whether the system will converge or not. Given two different starting values α_{x0} and α_{z0}, by assuming the ϕ introduced previously to be the same for both cases, we can derive the predictor

$$\widehat{\alpha} = \alpha_\infty = \left(\Delta_{x0}\alpha_{z0} - \Delta_{z0}\alpha_{x0}\right) / \left(\Delta_{x0} - \Delta_{z0}\right) \qquad (8.9)$$

where Δ_{x0} and Δ_{z0} have the same definition as previous corresponding to α_{x0} and α_{z0}. We have two ways of applying this approach. A negative CIA predictor from (8.9) indicates the iterations will not converge. Thus, we may determine the convergence just after the first iteration by setting up two different starting values and then apply the previous method for prediction. We can also predict the estimate directly from CIA. After determining the convergence, substituting, say, α_{x5}, α_{z5}, Δ_{x5} and Δ_{z5} may give a prediction close to the true value. The codes for HGLM algorithm given in Appendix have several flexible parameters in the function, where the prediction methods we discussed are included. Overcoming the efficiency problem of convergence for HGLM algorithm will make this method more powerful when applying to various situations and larger data sets.

In this article, the HGLM algorithm for normal linear mixed models is implemented and evaluated with applications. The most important advantage of the algorithm is that it can be easily extended to non-normal HGLMs, when we have non-normal random effects. We have shown that the HGLM algorithm is feasible in estimating variance components, especially in QTL analysis. Comparing to Fisher Scoring method, the HGLM algorithm has no problem of negative variance component estimates because of gamma-GLM fitting. During this comparison, we have also discovered that the HGLM algorithm has very low efficiency in convergence. To solve this problem, we have discussed some prediction methods such as the Common Intercept Approach. Involving predictors can make the HGLM algorithm more efficient.

[xiv] Proof in detail for Equation (8.8) and (8.9) is given in Appendix.

Appendix

Distribution of y (Response of Pigs' Phenotypes Data)

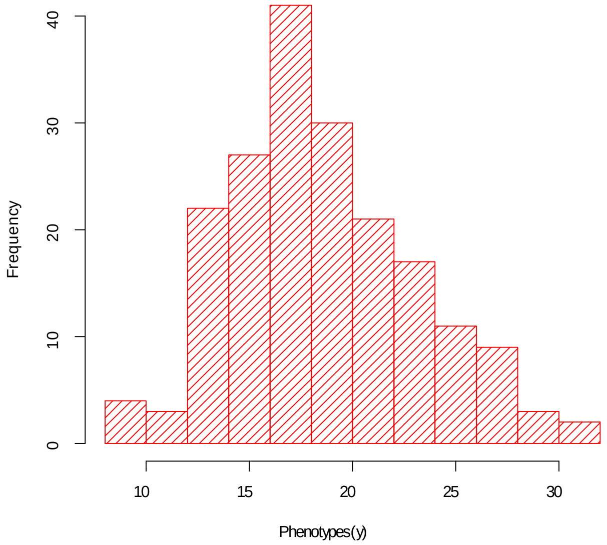

Figure A.1 Histogram of response variable y for pigs' phenotypes data. It is assumed to be normally distributed.

Proof of Equation (8.8)

Assuming $\phi_i = \phi$ for all iterations, we have

$$\begin{aligned} \Delta_i &= \Delta_{i-1}/\phi \\ &= \Delta_{i-2}/\phi^2 \\ &= \cdots \\ &= \Delta_0/\phi^i \end{aligned} \tag{A.1}$$

then

$$\begin{aligned} \alpha_i &= \alpha_{i-1} + \Delta_{i-1} \\ &= \alpha_{i-2} + \Delta_{i-2} + \Delta_{i-1} \\ &= \cdots \\ &= \alpha_0 + \sum_{j=0}^{i-1} \Delta_j \\ &= \alpha_0 + \sum_{j=0}^{i-1} \Delta_0/\phi^j \\ &= \alpha_0 + \Delta_0 \sum_{j=0}^{i-1} \phi^{-j} \end{aligned} \tag{A.2}$$

Notice that

$$\begin{aligned} \sum_{j=0}^{\infty} \phi^{-j} &= 1 + \sum_{j=1}^{\infty} \phi^{-j} \\ &= 1 + \phi^{-1} \sum_{j=0}^{\infty} \phi^{-j} \end{aligned} \tag{A.3}$$

which gives

$$\sum_{j=0}^{\infty} \phi^{-j} = \left(1 - \phi^{-1}\right)^{-1} \tag{A.4}$$

Thus, let i in (A.2) go to infinity, then we have the predictor

$$\begin{aligned} \widehat{\alpha} &= \alpha_\infty \\ &= \alpha_0 + \Delta_0 \sum_{j=0}^{\infty} \phi^{-j} \\ &= \alpha_0 + \Delta_0 \left(1 - \phi^{-1}\right)^{-1} \end{aligned} \tag{A.5}$$

Setting α_i to be the starting value gives Equation (8.8).

Proof of Equation (8.9)

CIA assumes a common ϕ for both cases with starting values α_{x0} and α_{z0}. From Equation (A.5), we have

$$\alpha_{x\infty} = \alpha_{x0} + \Delta_{x0} \left(1 - \phi^{-1}\right)^{-1} \tag{A.6}$$

or

$$\left(1 - \phi^{-1}\right)^{-1} = \left(\alpha_{x\infty} - \alpha_{x0}\right) \Delta_{x0}^{-1} \tag{A.7}$$

Then for the other case, we obtain

$$\begin{aligned} \alpha_{z\infty} &= \alpha_{z0} + \Delta_{z0} \left(1 - \phi^{-1}\right)^{-1} \\ &= \alpha_{z0} + \Delta_{z0} \left(\alpha_{x\infty} - \alpha_{x0}\right) \Delta_{x0}^{-1} \end{aligned} \tag{A.8}$$

From convergence and the limit theory, there should be

$$\alpha_{z\infty} = \alpha_{x\infty} = \alpha_\infty \tag{A.9}$$

Substituting (A.9) into (A.8) and re-arranging the equation give Equation (8.9).

R Codes of HGLM Algorithm[i]

```
####################################################
## Hierarchical Generalized Linear Models (HGLMs) ##
####################################################
HGLM <- function(family="normal", random.effect.number=1, y, X, Z, Z1, Z2, PI, specify.PI=F, starting=1,
critical.change=1e-5, max.n=200, prediction=F, pred.max.n=10, CIA=F, CIA.starting=2, std.error=F, profile=T, graphs=T,
printing=F, name="This Model"){
#################################################################################################################################
## Responsible Programmer:  Xia Shen (h07xiash@du.se)
## Version:  1.0
## Date:  15 MAY 2008
## family is only available for normal case in this version.
## random.effect.number should be 1 or 2.  For 1, Z1 and Z2 are not available.
## y is the response vector.
## X is the design matrix for fixed effects.
## Z, Z1 or Z2 are the design matrices for random effects.
## Default specify.PI is set to be FALSE, which requires Z or Z2.  If TRUE, PI=ZZ' or Z2Z2' is required.
## Default starting is set to be 1, which is the initial value for variance components.
## Default critical.change is set to be .00001, which is the the critical value of change in variance components.
## Default max.n is set to be 200, which is the most iterations allowed.
## Default prediction is set to be FALSE. If TRUE, estimates may be obtained by prediction.
## Default pred.max.n is set to be 10, which indicates when to make a prediction and is only available if prediction=TRUE.
## Default CIA is set to be FALSE. If TRUE, Common Intercept Approach will be applied for judging convergence after 1 iteration.
## Default CIA.starting is set to be 2, which is the other initial value that only meaningful if CIA=TRUE.
## Default std.error is set to be FALSE. If TRUE, standard errors for variance components will be calculated.({numDeriv}required)
## Default profile is set to be TRUE, which gives the value of profile likelihood function.
## Default graphs is set to be TURE, which gives related figures.
## Default printing is set to be FALSE. If TRUE, detailed results of fitting will be printed out.
## Default name is set to be "This Model", which can include the specified name of the current model.
#################################################################################################################################
##########################################
## Model with Only 1 Random Effect Term ##
##########################################
if(random.effect.number==1) {
   if(specify.PI==F) {psi_M <- rep(0, ncol(Z))} else {psi_M <- rep(0, ncol(PI))}
   make.y_a <- function(y, psi_M) {c(y, psi_M)}
   make.T <- function(X, Z) {
      T1 <- cbind(X, Z)
      T0 <- matrix(rep(0,ncol(Z)*ncol(X)),ncol(Z),ncol(X))
      T2 <- cbind(T0, diag(ncol(Z)))
      T <- rbind(T1, T2)
      return(T)
   }
   make.Sigma_a <- function(Sigma, D) {
      S0 <- matrix(rep(0,ncol(Sigma)*ncol(D)),ncol(Sigma),ncol(D))
      T1 <- cbind(Sigma, S0)
      T2 <- cbind(t(S0), D)
      T <- rbind(T1, T2)
      return(T)
   }
   sigma2 <- sigma2_v <- starting
   tau0 <- c(sigma2, sigma2_v)
   rec.tau <- tau0
   Sigma <- tau0[1]*diag(nrow(X))
   if(specify.PI==F) {D <- tau0[2]*diag(ncol(Z))}
   else {
      A <- svd(PI)
      Z <- A$u%*%diag(sqrt(A$d))%*%t(A$v)
      D <- tau0[2]*diag(ncol(Z))
   }
   Sigma_a <- make.Sigma_a(Sigma, D)
   T. <- make.T(X, Z)
   y_a <- make.y_a(y, psi_M)
   delta <- solve(t(T.)%*%solve(Sigma_a)%*%T.)%*%t(T.)%*%solve(Sigma_a)%*%y_a
   beta <- delta[1:ncol(X)]
   v <- delta[(ncol(X)+1):length(delta)]
   rec.beta <- beta
   rec.v <- v
   H <- T.%*%solve(t(T.)%*%solve(Sigma_a)%*%T.)%*%t(T.)%*%solve(Sigma_a)
   q <- q_M <- NULL
   for(i in 1:nrow(X)) {q[i] <- H[i,i]}
   for(i in (nrow(X)+1):(nrow(X)+ncol(Z))) {q_M[i-nrow(X)] <- H[i,i]}
   d <- (y-X%*%beta-Z%*%v)^2
   d_M <- (psi_M-v)^2
   d.star <- d/(1-q)
   d.star_M <- d_M/(1-q_M)
   sigma2 <- glm(d.star~1, family=Gamma(link=identity), weights=(1-q)/2)$coefficients
   sigma2_v <- glm(d.star_M~1, family=Gamma(link=identity), weights=(1-q_M)/2)$coefficients
   tau <- c(sigma2, sigma2_v)
   ###############################
   ## CIA Convergence Judgement ##
   ###############################
   if(CIA==T) {
      sigma2 <- sigma2_v <- CIA.starting
      tau0 <- c(sigma2, sigma2_v)
      Sigma <- tau0[1]*diag(nrow(X))
      if(specify.PI==F) {D <- tau0[2]*diag(ncol(Z))}
      else {
         A <- svd(PI)
         Z <- A$u%*%diag(sqrt(A$d))%*%t(A$v)
         D <- tau0[2]*diag(ncol(Z))
      }
      Sigma_a <- make.Sigma_a(Sigma, D)
      T. <- make.T(X, Z)
      y_a <- make.y_a(y, psi_M)
      delta <- solve(t(T.)%*%solve(Sigma_a)%*%T.)%*%t(T.)%*%solve(Sigma_a)%*%y_a
      beta <- delta[1:ncol(X)]
      v <- delta[(ncol(X)+1):length(delta)]
```

[i] This part of Appendix is the HGLM algorithm codes version for Windows©, and the version for Mac OS© X is also available. For the executable *.R files, please visit Xia Shen's homepage: http://xiashen.co.cc.

```
   H <- T.%*%solve(t(T.)%*%solve(Sigma_a)%*%T.)%*%t(T.)%*%solve(Sigma_a)
   q <- q_M <- NULL
   for(i in 1:nrow(X)) {q[i] <- H[i,i]}
   for(i in (nrow(X)+1):(nrow(X)+ncol(Z))) {q_M[i-nrow(X)] <- H[i,i]}
   d <- (y-X%*%beta-Z%*%v)^2
   d_M <- (psi_M-v)^2
   d.star <- d/(1-q)
   d.star_M <- d_M/(1-q_M)
   sigma2 <- glm(d.star~1, family=Gamma(link=identity), weights=(1-q)/2)$coefficients
   sigma2_v <- glm(d.star_M~1, family=Gamma(link=identity), weights=(1-q_M)/2)$coefficients
   tau_CIA <- c(sigma2, sigma2_v)
   pred_CIA <- ((tau-starting)*CIA.starting-(tau_CIA-CIA.starting)*starting)/((tau-starting)-(tau_CIA-CIA.starting))
   if(min(pred_CIA)>=0) {
      print(list(CIA=paste("CIA judgement indicates the iterations may converge with predictor", pred_CIA, ".")))}
   if(max(pred_CIA)<0) {
      print(list(CIA=paste("CIA judgement indicates the iterations may NOT converge with predictor",
      pred_CIA, "!")))}
}
rec.tau <- cbind(rec.tau, tau)
n <- 1
conv <- F
dis <- max(abs(tau-tau0))
if(dis<=critical.change) {conv <- T}
if(prediction==T) {max.n <- pred.max.n}
while(dis>critical.change & n<=max.n-1) {
   tau0 <- tau
   Sigma <- tau0[1]*diag(nrow(X))
   if(specify.PI==F) {D <- tau0[2]*diag(ncol(Z))}
   else {
      A <- svd(PI)
      Z <- A$u%*%diag(sqrt(A$d))%*%t(A$v)
      D <- tau0[2]*diag(ncol(Z))
   }
   Sigma_a <- make.Sigma_a(Sigma, D)
   delta <- solve(t(T.)%*%solve(Sigma_a)%*%T.)%*%t(T.)%*%solve(Sigma_a)%*%y_a
   beta <- delta[1:ncol(X)]
   rec.beta <- cbind(rec.beta, beta)
   v <- delta[(ncol(X)+1):length(delta)]
   rec.v <- cbind(rec.v, v)
   Hat <- T.%*%solve(t(T.)%*%solve(Sigma_a)%*%T.)%*%t(T.)%*%solve(Sigma_a)
   q <- q_M <- NULL
   for(i in 1:nrow(X)) {q[i] <- Hat[i,i]}
   for(i in (nrow(X)+1):(nrow(X)+ncol(Z))) {q_M[i-nrow(X)] <- Hat[i,i]}
   d <- (y-X%*%beta-Z%*%v)^2
   d_M <- (psi_M-v)^2
   d.star <- d/(1-q)
   d.star_M <- d_M/(1-q_M)
   sigma2 <- glm(d.star~1, family=Gamma(link=identity), weights=(1-q)/2)$coefficients
   sigma2_v <- glm(d.star_M~1, family=Gamma(link=identity), weights=(1-q_M)/2)$coefficients
   tau <- c(sigma2, sigma2_v)
   rec.tau <- cbind(rec.tau, tau)
   n <- n+1
   dis <- max(abs(tau-tau0))
}
if(prediction==T & n==max.n) {
   alpha <- rec.tau
   Delta <- alpha[,2:ncol(alpha)]-alpha[,1:(ncol(alpha)-1)]
   phi <- Delta[,1:(ncol(Delta)-1)]/Delta[,2:ncol(Delta)]
   alpha_inf <- alpha[,n-1]+Delta[,n-1]/(1-1/phi[,n-2])
   tau <- alpha_inf
   rec.tau <- cbind(rec.tau, tau)
   rec.beta <- cbind(rec.beta, beta)
   rec.v <- cbind(rec.v, v)
   n <- n+1
   print(list(PREDICTION=paste("Estimates of variance components were predicted after", max.n, "iterations.")))
}
if(n<=max.n | prediction==T) {conv <- T} else {conv <- F}
if(conv==T) {
   Cov.Effects <- solve(H)
   H <- t(T.)%*%solve(Sigma_a)%*%T.
   prof.tau <- function(tau) {
      Sigma <- tau[1]*diag(nrow(X))
      D <- tau[2]*diag(ncol(Z))
      V <- Z%*%D%*%t(Z)+Sigma
      p <- -.5*log(det(V))-.5*t(y-X%*%beta)%*%solve(V)%*%(y-X%*%beta)-.5*log(det(t(X)%*%solve(V)%*%X))
      return(p)
   }
   sing <- F
   if(std.error==T) {
      Hessian <- hessian(prof.tau, tau, method="Richardson")
      if(det(Hessian)<1e-10) {
         sing <- T
         print(list(HESSIAN=paste("Hessian matrix is detected to be singular,
         standard errors of variance components are set to be 0.")))
      }
      else {Cov.tau <- -solve(Hessian)}
   }
   est <- matrix(rep(0,2*(length(beta)+length(v)+2)), length(beta)+length(v)+2, 2)
   name.beta <- name.v <- NULL
   for(i in 1:length(beta)) {
      est[i,1] <- beta[i]
      est[i,2] <- Cov.Effects[i,i]
      name.beta <- c(name.beta, paste("beta_", as.character(i), sep=""))
   }
   for(i in (length(beta)+1):(length(beta)+length(v))) {
      est[i,1] <- v[i-length(beta)]
      est[i,2] <- Cov.Effects[i-length(beta)+1,i-length(beta)+1]
      name.v <- c(name.v, paste("v_", as.character(i-length(beta)), sep=""))
   }
   est[length(beta)+length(v)+1,1] <- tau[1]
   est[length(beta)+length(v)+2,1] <- tau[2]
   if(std.error==T & sing==F) {
      est[length(beta)+length(v)+1,2] <- Cov.tau[1,1]
      est[length(beta)+length(v)+2,2] <- Cov.tau[2,2]
      }
      dimnames(est) <- list(c(name.beta, name.v, "sigma.square", "sigma.square_v"), c("Estimate","Std.Error"))
      est <- round(est, digits=6)
      num.iteration <- paste("IWLS iterations for", name, "converged after", n, "times.")
      if(prediction==T & n==max.n) {num.iteration <- paste("IWLS iterations for", name, "converged after",
      n-1, "times.")}
      if(profile==T) {
         prof.likelihood <- paste("The profile likelihood function equals", prof.tau(tau), ".")
         output <- list(ESTIMATES=est, LIKELIHOOD=prof.likelihood, ITERATION=num.iteration)
      }
      else {output <- list(ESTIMATES=est, ITERATION=num.iteration)}
      print(output)
      e <- y-X%*%beta-Z%*%v
      if(printing==T) {
         esty <- cbind(y, X%*%beta+Z%*%v, e)
         dimnames(esty) <- list(1:length(y), c("y","Estimate","e-resid"))
         print(esty)
      }
      if(graphs==T) {
         m <- 0:n
         ########################
         ## Estimation Figures ##
         ########################
         par(mfrow=c(3,2))
         std.resid.e <- e/sqrt(1-q)
         plot(0, rec.tau[1,1], xlim=c(0,n), ylim=c(min(rec.tau),max(rec.tau)), xlab="Number of Iteration",
         ylab="Estimates of tau")
         lines(m, rec.tau[1,], type="b", col=4)
         lines(m, rec.tau[2,], type="b", col=2)
         lines(0, rec.tau[1,1], type="p")
         plot(m, rec.tau[1,]+rec.tau[2,], pch=19, col=2, xlab="Number of Iteration", ylab="Proportion of VC")
         lines(m, rec.tau[2,], type="h", col=3)
         plot(1:n, rec.beta[1,], type="b", ylim=c(min(rec.beta),max(rec.beta)), col=2, xlab="Number of Iteration",
         ylab="Estimates of beta")
         lines(1:n, rec.beta[1,], type="p", col=2)
         if(ncol(X)>1) {for(i in 2:ncol(X)) {lines(1:n, rec.beta[i,], type="b", col=i+1)}}
         plot(beta, type="h", col=3)
         lines(1:length(beta), rep(0, length(beta)), col=4, lty=2)
         lines(beta, type="p", col=2, pch=19)
         plot(1:n, rec.v[1,], col=2, ylim=c(min(rec.v),max(rec.v)), xlab="Number of Iteration", ylab="Estimates of v")
         lines(1:n, rec.v[1,], lty=2, col=2)
         if(ncol(Z)>1) {for(i in 2:ncol(Z)) {lines(1:n, rec.v[i,], type="b", col=i+1)}}
         plot(v, type="h", col=3)
         lines(1:length(v), rep(0, length(v)), col=4, lty=2)
         lines(v, type="p", col=2, pch=19)
         ############################
         ## Model Checking Figures ##
         ############################
         windows()
         par(mfrow=c(2,3))
         plot(X%*%beta, std.resid.e, pch="+", col=4, xlab="Xbeta", ylab="Standard e Residuals")
         qqnorm(std.resid.e, pch="+", xlab="Normal Order Statistics", ylab="Ordered Standard e Residuals",
         col=4, main="")
         qqline(std.resid.e, col=2)
         hist(std.resid.e, density=20, col=4, xlab="Standard e Residuals", main="")
         plot(y, d, pch="+", xlab="Response", ylab="Deviance Residuals", col=2)
         sp <- mean(d)^2/var(d)
         sc <- var(d)/mean(d)
         qqplot(rgamma(9999,shape=sp,scale=sc),d, pch="+", xlab="Gamma Order Statistics", ylab="Ordered Deviance Residuals", col=2, main="")
         abline(0,1, col=4)
         hist(d, density=20, col=2, xlab="Deviance", main="")
         ###########################
         ## Model Fitting Figures ##
         ###########################
         windows()
         plot(y, col=2, pch=19, ylab="y apart from estimated y")
         lines(X%*%beta+Z%*%v, col=4)
         for(i in 1:length(y)) {lines(c(i,i),c(min(y[i], (X%*%beta+Z%*%v)[i]),max(y[i], (X%*%beta+Z%*%v)[i])),
         lty=2, col=3)}
         lines(y, type="p", col=2, pch=20)
      }
   }
   else {print(paste("Iterations did NOT converge for", name, "!"))}
}
######################################
## Model with 2 Random Effect Terms ##
######################################
if(random.effect.number==2) {
   psi_M1 <- rep(0, ncol(Z1))
   if(specify.PI==F) {psi_M2 <- rep(0, ncol(Z2))} else {psi_M2 <- rep(0, ncol(PI))}
   make2.y_a <- function(y, psi_M1, psi_M2) {c(y, psi_M1, psi_M2)}
   make2.T <- function(X, Z1, Z2) {
      T1 <- cbind(X, Z1, Z2)
      T0 <- matrix(rep(0,(ncol(Z1)+ncol(Z2))*ncol(X)),ncol(Z1)+ncol(Z2),ncol(X))
      T2 <- cbind(T0, diag(ncol(Z1)+ncol(Z2)))
      T <- rbind(T1, T2)
      return(T)
   }
   make2.Sigma_a <- function(Sigma, D1, D2) {
      S12 <- matrix(rep(0,ncol(Sigma)*ncol(D1)),ncol(Sigma),ncol(D1))
      S13 <- matrix(rep(0,ncol(Sigma)*ncol(D2)),ncol(Sigma),ncol(D2))
      S23 <- matrix(rep(0,ncol(D1)*ncol(D2)),ncol(D1),ncol(D2))
      T1 <- cbind(Sigma, S12, S13)
      T2 <- cbind(t(S12), D1, S23)
      T3 <- cbind(t(S13), t(S23), D2)
      T <- rbind(T1, T2, T3)
      return(T)
   }
   sigma2 <- sigma2_v1 <- sigma2_v2 <- starting
   tau0 <- c(sigma2, sigma2_v1, sigma2_v2)
   rec.tau <- tau0
   Sigma <- tau0[1]*diag(nrow(X))
   D1 <- tau0[2]*diag(ncol(Z1))
   if(specify.PI==F) {D2 <- tau0[3]*diag(ncol(Z2))}
```

```
else {
   A <- svd(PI)
   Z2 <- A$u%*%diag(sqrt(A$d))%*%t(A$v)
   D2 <- tau0[3]*diag(ncol(Z2))
}
Sigma_a <- make2.Sigma_a(Sigma, D1, D2)
T. <- make2.T(X, Z1, Z2)
y_a <- make2.y_a(y, psi_M1, psi_M2)
delta <- solve(t(T.)%*%solve(Sigma_a)%*%T.)%*%t(T.)%*%solve(Sigma_a)%*%y_a
beta <- delta[1:ncol(X)]
v1 <- delta[(ncol(X)+1):(ncol(X)+ncol(Z1))]
v2 <- delta[(ncol(X)+ncol(Z1)+1):length(delta)]
rec.beta <- beta
rec.v1 <- v1
rec.v2 <- v2
H <- T.%*%solve(t(T.)%*%solve(Sigma_a)%*%T.)%*%t(T.)%*%solve(Sigma_a)
q <- q_M1 <- q_M2 <- NULL
for(i in 1:nrow(X)) {q[i] <- H[i,i]}
for(i in (nrow(X)+1):(nrow(X)+ncol(Z1))) {q_M1[i-nrow(X)] <- H[i,i]}
for(i in (nrow(X)+ncol(Z1)+1):(nrow(X)+ncol(Z1)+ncol(Z2))) {q_M2[i-nrow(X)-ncol(Z1)] <- H[i,i]}
d <- (y-X%*%beta-Z1%*%v1-Z2%*%v2)^2
d_M1 <- (psi_M1-v1)^2
d_M2 <- (psi_M2-v2)^2
d.star <- d/(1-q)
d.star_M1 <- d_M1/(1-q_M1)
d.star_M2 <- d_M2/(1-q_M2)
sigma2 <- glm(d.star~1, family=Gamma(link=identity), weights=(1-q)/2)$coefficients
sigma2_v1 <- glm(d.star_M1~1, family=Gamma(link=identity), weights=(1-q_M1)/2)$coefficients
sigma2_v2 <- glm(d.star_M2~1, family=Gamma(link=identity), weights=(1-q_M2)/2)$coefficients
tau <- c(sigma2, sigma2_v1, sigma2_v2)
###############################
## CIA Convergence Judgement ##
###############################
if(CIA==T) {
   sigma2 <- sigma2_v1 <- sigma2_v2 <- CIA.starting
   tau0 <- c(sigma2, sigma2_v1, sigma2_v2)
   Sigma <- tau0[1]*diag(nrow(X))
   D1 <- tau0[2]*diag(ncol(Z1))
   if(specify.PI==F) {D2 <- tau0[3]*diag(ncol(Z2))}
   else {
      A <- svd(PI)
      Z2 <- A$u%*%diag(sqrt(A$d))%*%t(A$v)
      D2 <- tau0[3]*diag(ncol(Z2))
   }
   Sigma_a <- make2.Sigma_a(Sigma, D1, D2)
   T. <- make2.T(X, Z1, Z2)
   y_a <- make2.y_a(y, psi_M1, psi_M2)
   delta <- solve(t(T.)%*%solve(Sigma_a)%*%T.)%*%t(T.)%*%solve(Sigma_a)%*%y_a
   beta <- delta[1:ncol(X)]
   v1 <- delta[(ncol(X)+1):(ncol(X)+ncol(Z1))]
   v2 <- delta[(ncol(X)+ncol(Z1)+1):length(delta)]
   H <- T.%*%solve(t(T.)%*%solve(Sigma_a)%*%T.)%*%t(T.)%*%solve(Sigma_a)
   q <- q_M1 <- q_M2 <- NULL
   for(i in 1:nrow(X)) {q[i] <- H[i,i]}
   for(i in (nrow(X)+1):(nrow(X)+ncol(Z1))) {q_M1[i-nrow(X)] <- H[i,i]}
   for(i in (nrow(X)+ncol(Z1)+1):(nrow(X)+ncol(Z1)+ncol(Z2))) {q_M2[i-nrow(X)-ncol(Z1)] <- H[i,i]}
   d <- (y-X%*%beta-Z1%*%v1-Z2%*%v2)^2
   d_M1 <- (psi_M1-v1)^2
   d_M2 <- (psi_M2-v2)^2
   d.star <- d/(1-q)
   d.star_M1 <- d_M1/(1-q_M1)
   d.star_M2 <- d_M2/(1-q_M2)
   sigma2 <- glm(d.star~1, family=Gamma(link=identity), weights=(1-q)/2)$coefficients
   sigma2_v1 <- glm(d.star_M1~1, family=Gamma(link=identity), weights=(1-q_M1)/2)$coefficients
   sigma2_v2 <- glm(d.star_M2~1, family=Gamma(link=identity), weights=(1-q_M2)/2)$coefficients
   tau_CIA <- c(sigma2, sigma2_v1, sigma2_v2)
   pred_CIA <- ((tau-starting)*CIA.starting-(tau_CIA-CIA.starting)*starting)/((tau-starting)-(tau_CIA-CIA.starting))
   if(min(pred_CIA)>=0) {
      print(list(CIA=paste("CIA judgement indicates the iterations may converge with predictor", pred_CIA, ".")))}
   if(max(pred_CIA)<0) {
      print(list(CIA=paste("CIA judgement indicates the iterations may NOT converge with predictor",
      pred_CIA, "!")))}
}
rec.tau <- cbind(rec.tau, tau)
n <- 1
conv <- F
dis <- max(abs(tau-tau0))
if(dis<=critical.change) {conv <- T}
if(prediction==T) {max.n <- pred.max.n}
while(dis>critical.change & n<=max.n-1) {
   tau0 <- tau
   Sigma <- tau0[1]*diag(nrow(X))
   D1 <- tau0[2]*diag(ncol(Z1))
   if(specify.PI==F) {D2 <- tau0[3]*diag(ncol(Z2))}
   else {
      A <- svd(PI)
      Z2 <- A$u%*%diag(sqrt(A$d))%*%t(A$v)
      D2 <- tau0[3]*diag(ncol(Z2))
   }
   Sigma_a <- make2.Sigma_a(Sigma, D1, D2)
   T. <- make2.T(X, Z1, Z2)
   y_a <- make2.y_a(y, psi_M1, psi_M2)
   delta <- solve(t(T.)%*%solve(Sigma_a)%*%T.)%*%t(T.)%*%solve(Sigma_a)%*%y_a
   beta <- delta[1:ncol(X)]
   v1 <- delta[(ncol(X)+1):(ncol(X)+ncol(Z1))]
   v2 <- delta[(ncol(X)+ncol(Z1)+1):length(delta)]
   rec.beta <- cbind(rec.beta,beta)
   rec.v1 <- cbind(rec.v1,v1)
   rec.v2 <- cbind(rec.v2,v2)
   H <- T.%*%solve(t(T.)%*%solve(Sigma_a)%*%T.)%*%t(T.)%*%solve(Sigma_a)
   q <- q_M1 <- q_M2 <- NULL
   for(i in 1:nrow(X)) {q[i] <- H[i,i]}
   for(i in (nrow(X)+1):(nrow(X)+ncol(Z1))) {q_M1[i-nrow(X)] <- H[i,i]}
   for(i in (nrow(X)+ncol(Z1)+1):(nrow(X)+ncol(Z1)+ncol(Z2))) {q_M2[i-nrow(X)-ncol(Z1)] <- H[i,i]}
```

```
      d <- (y-X%*%beta-Z1%*%v1-Z2%*%v2)^2
      d_M1 <- (psi_M1-v1)^2
      d_M2 <- (psi_M2-v2)^2
      d.star <- d/(1-q)
      d.star_M1 <- d_M1/(1-q_M1)
      d.star_M2 <- d_M2/(1-q_M2)
      sigma2 <- glm(d.star~1, family=Gamma(link=identity), weights=(1-q)/2)$coefficients
      sigma2_v1 <- glm(d.star_M1~1, family=Gamma(link=identity), weights=(1-q_M1)/2)$coefficients
      sigma2_v2 <- glm(d.star_M2~1, family=Gamma(link=identity), weights=(1-q_M2)/2)$coefficients
      tau <- c(sigma2, sigma2_v1, sigma2_v2)
      rec.tau <- cbind(rec.tau, tau)
      n <- n+1
      dis <- max(abs(tau-tau0))
   }
   if(prediction==T & n==max.n) {
      alpha <- rec.tau
      Delta <- alpha[,2:ncol(alpha)]-alpha[,1:(ncol(alpha)-1)]
      phi <- Delta[,1:(ncol(Delta)-1)]/Delta[,2:ncol(Delta)]
      alpha_inf <- alpha[,n-1]+Delta[,n-1]/(1-1/phi[,n-2])
      tau <- alpha_inf
      rec.tau <- cbind(rec.tau, tau)
      rec.beta <- cbind(rec.beta, beta)
      rec.v1 <- cbind(rec.v1,v1)
      rec.v2 <- cbind(rec.v2,v2)
      n <- n+1
      print(list(PREDICTION=paste("Estimates of variance components were predicted after", max.n, "iterations.")))
   }
   if(n<=max.n | prediction==T) {conv <- T} else {conv <- F}
   if(conv==T) {
      H <- t(T.)%*%solve(Sigma_a)%*%T.
      Cov.Effects <- solve(H)
      prof.tau <- function(tau) {
         Sigma <- tau[1]*diag(nrow(X))
         D1 <- tau[2]*diag(ncol(Z1))
         D2 <- tau[3]*diag(ncol(Z2))
         V <- Z1%*%D1%*%t(Z1)+Z2%*%D2%*%t(Z2)+Sigma
         p <- -.5*log(det(V))-.5*t(y-X%*%beta)%*%solve(V)%*%(y-X%*%beta)-.5*log(det(t(X)%*%solve(V)%*%X))
         return(p)
      }
      sing <- F
      if(std.error==T) {
         Hessian <- hessian(prof.tau, tau, method="Richardson")
         if(det(Hessian)<1e-10) {
            sing <- T
            print(list(HESSIAN=paste("Hessian matrix is detected to be singular,
            standard errors of variance components are set to be 0.")))
         }
         else {Cov.tau <- -solve(Hessian)}
      }
      est <- matrix(rep(0,2*(length(beta)+length(v1)+length(v2)+3)), length(beta)+length(v1)+length(v2)+3, 2)
      name.beta <- name.v1 <- name.v2 <- NULL
      for(i in 1:length(beta)) {
         est[i,1] <- beta[i]
         est[i,2] <- Cov.Effects[i,i]
         name.beta <- c(name.beta, paste("beta_", as.character(i), sep=""))
      }
      for(i in (length(beta)+1):(length(beta)+length(v1))) {
         est[i,1] <- v1[i-length(beta)]
         est[i,2] <- Cov.Effects[i-length(beta)+1,i-length(beta)+1]
         name.v1 <- c(name.v1, paste("v1_", as.character(i-length(beta)), sep=""))
      }
      for(i in (length(beta)+length(v1)+1):(length(beta)+length(v1)+length(v2))) {
         est[i,1] <- v2[i-length(beta)-length(v1)]
         est[i,2] <- Cov.Effects[i-length(beta)-length(v1)+1,i-length(beta)-length(v1)+1]
         name.v2 <- c(name.v2, paste("v2_", as.character(i-length(beta)-length(v1)), sep=""))
      }
      est[length(beta)+length(v1)+length(v2)+1,1] <- tau[1]
      est[length(beta)+length(v1)+length(v2)+2,1] <- tau[2]
      est[length(beta)+length(v1)+length(v2)+3,1] <- tau[3]
      if(std.error==T & sing==F) {
         est[length(beta)+length(v1)+length(v2)+1,2] <- Cov.tau[1,1]
         est[length(beta)+length(v1)+length(v2)+2,2] <- Cov.tau[2,2]
         est[length(beta)+length(v1)+length(v2)+3,2] <- Cov.tau[3,3]
      }
      dimnames(est) <- list(c(name.beta, name.v1, name.v2, "sigma.square", "sigma.square_v1", "sigma.square_v2"),
      c("Estimate","Std.Error"))
      est <- round(est, digits=6)
      num.iteration <- paste("IWLS iterations for", name, "converged after", n, "times.")
      if(prediction==T & n==max.n) {num.iteration <- paste("IWLS iterations for", name, "converged after",
      n-1, "times.")}
      if(profile==T) {
         prof.likelihood <- paste("The profile likelihood function equals", prof.tau(tau), ".")
         output <- list(ESTIMATES=est, LIKELIHOOD=prof.likelihood, ITERATION=num.iteration)
      }
      else {output <- list(ESTIMATES=est, ITERATION=num.iteration)}
      print(output)
      e <- y-X%*%beta-Z1%*%v1-Z2%*%v2
      if(printing==T) {
         esty <- cbind(y, X%*%beta+Z1%*%v1+Z2%*%v2, e)
         dimnames(esty) <- list(1:length(y), c("y","Estimate","e-resid"))
         print(esty)
      }
      if(graphs==T) {
         m <- 0:n
         ########################
         ## Estimation Figures ##
         ########################
         par(mfrow=c(4,2))
         std.resid.e <- e/sqrt(1-q)
         plot(0, rec.tau[1,1], xlim=c(0,n), ylim=c(min(rec.tau),max(rec.tau)), xlab="Number of Iteration",
         ylab="Estimates of tau")
         lines(m, rec.tau[1,], type="b", col=4)
         lines(m, rec.tau[2,], type="b", col=2)
         lines(m, rec.tau[3,], type="b", col=3)
```

```
            lines(0, rec.tau[1,1], type="p")
            plot(m, rec.tau[1,]+rec.tau[2,]+rec.tau[3,], pch=19, col=2, xlab="Number of Iteration",
            ylab="Proportion of VC")
            lines(m, rec.tau[2,]+rec.tau[3,], type="h", col=4)
            lines(m, rec.tau[2,], type="h", col=3)
            plot(1:n, rec.beta[1,], type="b", ylim=c(min(rec.beta),max(rec.beta)), col=2, xlab="Number of Iteration",
            ylab="Estimates of beta")
            lines(1:n, rec.beta[1,], type="p", col=2)
            if(ncol(X)>1) {for(i in 2:ncol(X)) {lines(1:n, rec.beta[i,], type="b", col=i+1)}}
            plot(beta, type="h", col=3)
            lines(1:length(beta), rep(0, length(beta)), col=4, lty=2)
            lines(beta, type="p", col=2, pch=19)
            plot(1:n, rec.v1[1,], col=2, ylim=c(min(rec.v1),max(rec.v1)), xlab="Number of Iteration",
            ylab="Estimates of v1")
            lines(1:n, rec.v1[1,], lty=2, col=2)
            if(ncol(Z1)>1) {for(i in 2:ncol(Z1)) {lines(1:n, rec.v1[i,], type="b", col=i+1)}}
            plot(v1, type="h", col=3)
            lines(1:length(v1), rep(0, length(v1)), col=4, lty=2)
            lines(v1, type="p", col=2, pch=19)
            plot(1:n, rec.v2[1,], col=2, ylim=c(min(rec.v2),max(rec.v2)), xlab="Number of Iteration",
            ylab="Estimates of v2")
            lines(1:n, rec.v2[1,], lty=2, col=2)
            if(ncol(Z2)>1) {for(i in 2:ncol(Z2)) {lines(1:n, rec.v2[i,], type="b", col=i+1)}}
            plot(v2, type="h", col=3)
            lines(1:length(v2), rep(0, length(v2)), col=4, lty=2)
            lines(v2, type="p", col=2, pch=19)
            ############################
            ## Model Checking Figures ##
            ############################
            windows()
            par(mfrow=c(2,3))
            plot(X%*%beta, std.resid.e, pch="+", col=4, xlab="Xbeta", ylab="Standard e Residuals")
            qqnorm(std.resid.e, pch="+", xlab="Normal Order Statistics", ylab="Ordered Standard e Residuals",
            col=4, main="")
            qqline(std.resid.e, col=2)
            hist(std.resid.e, density=20, col=4, xlab="Standard e Residuals", main="")
            plot(y, d, pch="+", xlab="Response", ylab="Deviance Residuals", col=2)
            sp <- mean(d)^2/var(d)
            sc <- var(d)/mean(d)
            qqplot(rgamma(9999,shape=sp,scale=sc),d, pch="+", xlab="Gamma Order Statistics", ylab="Ordered Deviance Residuals", col=2, main="")
            abline(0,1, col=4)
            hist(d, density=20, col=2, xlab="Deviance", main="")
            ###########################
            ## Model Fitting Figures ##
            ###########################
            windows()
            plot(y, col=2, pch=19, ylab="y apart from estimated y")
            lines(X%*%beta+Z1%*%v1+Z2%*%v2, col=4)
            for(i in 1:length(y)) {lines(c(i,i),c(min(y[i],(X%*%beta+Z1%*%v1+Z2%*%v2)[i]),
            max(y[i],(X%*%beta+Z1%*%v1+Z2%*%v2)[i])), lty=2, col=3)}
            lines(y, type="p", col=2, pch=20)
        }
    }
    else {print(paste("Iterations did NOT converge for", name, "!"))}
}
}
```

R Codes of FS Algorithm

```
REML_FS<-function(reml.method="FS",y,X,n_comp,conv_crit,n_maxiter,lambda_start,delta,IBDformat=FALSE,
Z1=0,Z2=0,Zepi=0,IBD1=0,IBD2=0,IBD3=0,IBD4=0,IBD5=0,print_results=FALSE,step=1,neg.Hessian.OK=FALSE) {
#############################################################################################################
## Responsible programmer:  Lars Rönnegård (lrn@du.se)
## Latest Modification by Xia Shen (h07xiash@du.se)
## Only the reml.method of FS (Fisher Scoring) is given.
## y:  Response vector
## X: Design matrix for fixed effects
## n_comp:  Number of different random effects in the model (<6 if IBDformat==TRUE, otherwise <4)
## conv_crit:  Value that the change in variance components should be less than
## n_maxiter:  Maximum number of iterations
## lambda_start:  Initial ratio of variance components
## delta:  Value added to truncation at zero
## IBDformat:  The structure of the random effects can be given either as incidence matrices or IBD matrices.
#############################################################################################################
print("REML iteration number")
min.error<-10^-8
count_test=0
n_comp1<-n_comp+1
n_rows<-length(y)
A<-matrix(0,(n_comp1*n_rows),n_rows)
Aj<-matrix(0,n_rows,n_rows)
if (n_comp>0) {
if (IBDformat==FALSE) {
for (i_comp in 1:n_comp) {
   if (i_comp==1) Aj<-Z1%*%t(Z1)
   if (i_comp==2) Aj<-Z2%*%t(Z2)
   if (i_comp==3) Aj<-Zepi%*%t(Zepi)
   A[((i_comp-1)*n_rows+1):(n_rows*i_comp),1:n_rows]<-Aj
   }
}
if (IBDformat==TRUE) {
for (i_comp in 1:n_comp) {
   if (i_comp==1) Aj<-IBD1
   if (i_comp==2) Aj<-IBD2
   if (i_comp==3) Aj<-IBD3
   if (i_comp==4) Aj<-IBD4
   if (i_comp==5) Aj<-IBD5
   A[((i_comp-1)*n_rows+1):(n_rows*i_comp),1:n_rows]<-Aj
   }
}
}
A[(n_comp*n_rows+1):(n_comp1*n_rows),1:n_rows]<-diag(rep(1,n_rows))
   #Algorithm from Johnson&Thompson
#A is a matrix with all ibd-matrices on top of each other with the identity matrix at the bottom
res_var<-var(y-X%*%solve(t(X)%*%X)%*%t(X)%*%y)
phi_start<-numeric(n_comp1)
for (i in 1:(n_comp1-1)) {
   phi_start[i]<-1
   }
phi_start[n_comp1]<-1
#############################
##if (reml.method=="FS") { ##
#############################
   dimIBD<-min(dim(A))
   M_phi<-matrix(0,(n_maxiter+1),n_comp1)
   M_phi[1,1:n_comp1]<-phi_start[1:n_comp1]
   phi<-numeric(n_comp1)
   DL<-numeric(n_comp1)
   DL[1:n_comp1]<-conv_crit+1
   FS<-matrix(0,n_comp1,n_comp1)
   Aj<-matrix(0,dimIBD,dimIBD)
   Ak<-matrix(0,dimIBD,dimIBD)
   llh.prev<-1+conv_crit
   llh<-0
   phi0<-0
   phi1<-1
   rec.val<-rec.phi<-NULL
   for (i in 1:n_maxiter){
      V<-matrix(0,dimIBD,dimIBD)
      if (max(abs(phi1-phi0))>conv_crit & count_test<3){
         phi<-M_phi[i,]
         phi0<-phi
         for (j in 1:n_comp1){
            Aj<-A[((j-1)*dimIBD+1):(j*dimIBD),1:dimIBD]
            V<-V+phi[j]*Aj
            }
         invV<-solve(V)
         temp<-solve(t(X)%*%invV%*%X)
         P<-invV-invV%*%X%*%(temp)%*%t(X)%*%invV
         for (j in 1:n_comp1){
            Aj<-A[((j-1)*dimIBD+1):(j*dimIBD),1:dimIBD]
            DL[j]<-sum(diag(Aj%*%P))-t(y)%*%P%*%Aj%*%P%*%y
            for (k in j:n_comp1) {
               Ak<-A[((k-1)*dimIBD+1):(k*dimIBD),1:dimIBD]
               FS[j,k]<-sum(diag(Aj%*%P%*%Ak%*%P))
               FS[k,j]<-FS[j,k]
            }
         }
         FS.egen<-eigen(FS, only.values=TRUE)
         #Condition number based 070330
         FS.min<-FS.egen$values[n_comp1]/FS.egen$values[1]
         if (FS.min>min.error | neg.Hessian.OK) {
            M_phi[(i+1),]<-phi-step*solve(FS)%*%DL
            count_test=0
            }
         if (FS.min<=min.error) {
            print("Negative Hessian")
            count_test=count_test+1
            identitet<-diag(rep((0.3+abs(min(FS.egen$values))),min(dim(FS))))
            if (!neg.Hessian.OK) M_phi[(i+1),]<-phi-step*solve(FS+identitet)%*%DL
         }

         egen<-eigen(V, only.values=TRUE)
         if (min(egen$values>0))ldV<-sum(log(egen$values))
         egen2<-eigen(t(X)%*%invV%*%X, only.values=TRUE)
         ldXVX<-sum(log(egen2$values))
         llh.prev<-llh
         if (min(egen$values)>0) llh<-(ldV+ldXVX+t(y)%*%P%*%y)*(-0.5)
         if (min(egen$values)<=0) llh<-llh.prev-1
         #Truncation at zero
         M_phi[(i+1),]<-0.5*(M_phi[(i+1),]+delta+abs(M_phi[(i+1),]-delta))
         phi1<-M_phi[(i+1),]
         rec.phi <- rbind(rec.phi,phi1)
         conv_val<-max(abs(phi1-phi0))
         rec.val <- c(rec.val, conv_val)
         if (print_results==TRUE) {
            print("---------------------")
            print("Iteration:")
            print(i)
            print("Convergence criteria:  Change in phi")
            print(conv_val)
            print("log-likelihood")
            print(llh)
            print("REML estimates of variance components")
            print("Genotype variance [1] and residual variance [2]")
            print(M_phi[(i+1),])
         }
         if (print_results==FALSE) print(paste(" ",i))
      }
   }
}
conv_test<-1
if (max(abs(phi1-phi0))>conv_crit) conv_test<-0
beta_hat<-numeric(min(dim(X)))
beta_hat<-solve(t(X)%*%invV%*%X)%*%t(X)%*%invV%*%y
if (print_results==TRUE) {
   print("Estimates of fixed effects")
   print(beta_hat)
   }
list(beta_hat=beta_hat,conv_test=conv_test,conv_val=conv_val,phi=phi,phi_iteration=M_phi,llh=llh)
res <- cbind(rec.phi,rec.val)
dimnames(res) <- list(NULL, c("sigma.square_v", "sigma.square", "Change in tau"))
return(res)
}
```

Bibliography

- Berger, J.O. and Wolpert, R. (1984). **The Likelihood Principle**. Hayward: Institute of Mathematical Statistics Monograph Series.
- Bjørnstad, J.F. (1996). **On the Generalization of the Likelihood Function and Likelihood Principle**. Journal of the American Statistical Association, Vol. 91, 791-806.
- Butler, R.W. (1986). **Predictive Likelihood Inference with Applications (with Discussion)**. Journal of the Royal Statistical Society, Series B (Methodological), Vol. 48, 1-38.
- Cox, D.R. and Hinkley, D.V. (1974). **Theoretical Statistics**. London: Chapman and Hall.
- Dempster, A.P., Laird, N.M. and Rubin, D.B. (1977). **Maximum Likelihood from Incomplete Data via the EM Algorithm**. Journal of the Royal Statistical Society, Series B (Methodological), Vol. 39, 1-38.
- Davidson, A.C. (2003). **Statistical Models**. Cambridge University Press.
- Fears, T.R., Benichou, J. and Gail, M.H. (1996). **A Reminder of the Fallibility of the Wald Statistic**. American Statistician, Vol. 50, 226-227.
- Gelfand, A.E. and Smith, A.F.M. (1990). **Sampling-based Approaches to Calculating Marginal Densities**. Journal of the American Statistical Association, Vol. 87, 523-532.
- Harville, D.A. (1977). **Maximum Likelihood Approaches to Variance Component Estimation and to Related Problems**. Journal of the American Statistical Association, Vol. 72, No. 358 (Jun., 1977), 320-338.
- Lee, Y. and Nelder, J.A. (1996). **Hierarchical Generalized Linear Models (with Discussion)**. Journal of the Royal Statistical Society, Series B (Methodological), Vol. 58, No. 4, 619-678.
- Lee, Y. and Nelder, J.A. (2001). **Hierarchical Generalised Linear Models: A Synthesis of Generalised Linear Models, Random-effect Model and Structured Dispersion**. Biometrika, Vol. 88, 987-1006.
- Lee, Y., Nelder, J.A. and Pawitan, Y. (2006). **Generalized Linear Models with Random Effects: Unified Analysis via H-likelihood**. Chapman & Hall/CRC, Taylor & Francis Group.
- Patterson, H.D. and Thompson, R. (1971). **Recovery of Inter-Block Information when Block Sizes are Unequal**. Biometrika, 58, 545-54.
- Pawitan, Y. (2001). **In All Likelihood: Statistical Modelling and Inference Using Likelihood**. Oxford: Clarendon Press.
- Robinson, G.K. (1991). **That BLUP is a Good Thing: The Estimation of Random Effects**. Statistical Science, Vol. 6, No. 1 (Feb., 1991), 15-32.
- Rönnegård, L. and Carlborg, Ö. (2007). **Separation of Base Allele and Sampling Term Effects Gives New Insights in Variance Component QTL Analysis**. BMC Genetics, 8:1, doi: 10.1186/1471-2156-8-1.
- Rönnegård, L., Al-Sarraj R. and von Rosen D. (2007). **Non-iterative Variance Component Estimation in QTL Analysis**.
- Schaeffer, L.R. (1979). **Estimation of Variance and Covariance Components for Average Daily Gain and Backfat Thickness in Swine**. Page 123 in Proc. Conf. in Honor of Henderson, C.R. on Variance Components Anim. Breed.
- Scott M.P., Matsudaira P., Lodish H., Darnell J., Zipursky L., Kaiser C.A., Berk A, Krieger M. (2004). **Molecular Cell Biology, Fifth Edition**. San Francisco: W. H. Freeman, 396. ISBN 0-7167-4366-3.
- Self, S.G. and Liang, K.Y. (1987). **Asymptotic Properties of Maximum Likelihood Estimators and Likelihood Ratio Tests under Non-standard Conditions**. Journal of the American Statistical Association. Vol. 82, 605-610.
- Vaida, F. and Meng, X.L. (2004). **Mixed Linear Models and the EM Algorithm in Applied Bayesian and Causal Inference from an Incomplete Data Perspective**. Gelman, A. and Meng, X.L. (editors): John Wiley and Sons.