Generalized Linear Mixed Models (GLMMs) and Generalized Estimating Equations (GEE) Notes

Daniel J. Eck

Contents

Generalized linear mixed models (GLMMs) are a natural extension that combines the GLM modeling framework with the ideas of random effects modeling. In this setting the response variables Y_i take observed values y_i which follow a (potentially over/under dispersed) exponential family with density of the form

$$f(y_i|\theta_i, \phi) = \exp\left(\frac{y_i\theta_i - c(\theta_i)}{a_i(\phi)} - b(y_i, \phi)\right), \tag{1}$$

where $y_i \in \mathbb{R}$ and $\theta_i \in \mathbb{R}$ are as before except that they are scalars, ϕ is a dispersion parameter, $a_i(\phi)$ is now a subject specific function of the dispersion parameter ϕ, and $b(y_i, \phi)$ is a function of the data y_i and the dispersion parameter ϕ. From the perspective of the canonical exponential families that we have motivated throughout, the function $b(y_i, \phi)$ is similar to the base measure h that was dropped from consideration in log likelihood based arguments that focused on the parameters.

Notice that the density (1) is a generalization of the exponential family density which specifies that $a_i(\phi) = 1$ and $b(y_i, \phi) = \log(h(y_i))$ and $c(\theta_i)$ is as before.

Let $\mathrm{E}Y_i = \mu_i$ and let this be connected to the linear predictor using the change-of-parameter mapping g by $\mu_i = g(\theta_i)$. Now let the random effects b have distribution $h(b|V)$ for parameters V. The fixed effects are β. Conditional on the random effects b,

$$\theta_i = x_i^T\beta + z_i^T b$$

where x_i and z_i are the corresponding rows of the the design matrices X and Z for the respective fixed and random effects. Now, the likelihood can be written as:

$$L(\beta, \phi, V|y) = \prod_{i=1}^{n} \int f(y_i|\beta, \phi, b)h(b|V)db.$$

Typically, the random effects are also assumed to be normal $b \sim N(0, D)$. However, unless f is also normal like in LMMs, the integral remains in the likelihood, which becomes difficult to compute, particularly if the random effects structure is complicated.

Overview of implementations

A variety of possible approaches are available for fitting GLMMs and obtaining inferences of model parameters. These approaches each have their own strengths and weaknesses, there is not one approach that is universally recognized as the best in all scenarios. We will present an overview of the theory behind these approaches before demonstrating the implementation on examples.

Penalized Quasi-Likelihood (PQL): Earlier in the course, we discussed how GLM optimization relates to weighted least squares (WLS) regression in linear models via the IRLS algorithm (see the exponential family notes for details). A modified version of this algorithm, using slightly different notation (described below), can be applied to fit GLMMs. In this context, the pseudo-response at iteration k is defined as

$$\tilde{y}^k = \hat{\theta}^k + (y - \hat{\mu}^k)\frac{\partial \theta}{\partial \mu}|_{\hat{\theta}^k}.$$

Here we will suppose that the conditional expectation of the pseudo-response is written as

$$\mathrm{E}(\tilde{y}_i|\beta, b) \approx x_i^T\beta + z_i^T b.$$

This transforms the GLMM fitting problem at iteration k into a weighted linear mixed model problem. To apply standard LMM machinery, we also approximate the conditional variance $\mathrm{Var}(\tilde{y}_i^k \mid b)$ using exponential family theory and a first-order Taylor expansion around the current estimates. This yields the working weights used in each iteration of the IRLS algorithm (see derivation below).

At each step, we can then fit a weighted LMM to the pseudo-response using standard LMM optimization techniques. While the method is often referred to as penalized quasi-likelihood (PQL), the name is somewhat misleading. True quasi-likelihood methods make minimal distributional assumptions (as we will see later with generalized estimating equations), whereas PQL still relies on normality assumptions for the random effects and uses a working Gaussian likelihood at each iteration. The PQL method is relatively easy to implement, as it allows existing LMM optimization techniques to be adapted for the GLMM setting. However, inference based on PQL is only asymptotically valid. Biased estimates are most likely to occur for binomial responses with small group sizes (i.e., few observations per covariate pattern), and are particularly problematic for Bernoulli responses. Similar issues arise for Poisson responses when counts are low. A PQL method is implemented using `glmmPQL` in the `MASS` package.

Numerical Integration: When the dimension of the random effects is not too large then it is possible to use numerical integration methods to approximate the likelihood. The Laplace approximation is one of the least demanding methods for computing integrals of the form $\int \exp(g(x))dx$. We need only find the maximum of g and the second derivative of $g(x)$ at the maximizer. For the integral in the GLMM likelihood, this can provide a surprisingly good approximation despite the integrand being evaluated at just one point. We can do better with more function evaluations. For these kinds of integrals, Gauss-Hermite quadrature is appropriate. The Gauss-Hermite quadrature method approximates integrals of the form:

$$\int g(x)\exp(-x^2)dx \approx \sum_k w_k g(x_k),$$

where the best choice of weights w_k and knotpoints x_k have been determined. This method is more accurate than the Laplace approach, but its computational cost can become prohibitive. Furthermore, the Gauss-Hermite quadrature is noted to provide solid performance when the function g is of low polynomial order which typically is the case in the GLMM setting (exponential family density and normal random effects). The Laplace approximation is a special case of The Gauss-Hermite quadrature method. Numerical integration methods can be performed using `glmer` in the `lme4` package. Faraway [2016] suggests that numerical integration methods are superior to PQL. The drawback is that they may be time-consuming or impossible to compute for more complex models.

Bayes: As with LMMs, there is good reason to consider Bayesian methods as an alternative to the standard likelihood-based methods. There are several advantages: Complex models can be fit with a high degree of accuracy; we can incorporate useful prior information; we have the flexibility to modify the models to allow for nonstandard features (such as non-Gaussian random effects, spatial correlation structures, or custom link functions). The disadvantages are that these models may require more programming to implement and may take substantial computing resources. Furthermore, one must address technical concerns about the quality of the fit (posterior approximation). Finally, the inferential conclusions are of a different form. This is either an advantage or disadvantage depending on your point of view. We can use the integrated nested laplace approximation (INLA), implemented in the `INLA` package, for a Bayesian approach to fitting GLMMs (see the intro_INLA slide deck which is in the same directory as these notes). One could also use `STAN` software but we will not discuss this implementation. See Faraway [2016] for more details.

Monte Carlo Likelihood Approximation: The `R` package `glmm` enables likelihood-based inference for GLMMs with a canonical link [Knudson et al., 2021]. No other publicly-available software accurately conducts likelihood-based inference for generalized linear mixed models with crossed random effects. `glmm` is able to do so by approximating the likelihood function and two derivatives using importance sampling. The importance sampling distribution is an essential piece of Monte Carlo likelihood approximation and developing a good one is the main challenge in implementing it. The package `glmm` uses the data to tailor the importance sampling distribution and is constructed to ensure finite Monte Carlo standard errors. While this software is great for the models that it works for, it is currently only applicable for canonical GLMMs with Binomial or Poisson responses. It can also be time consuming and a streamlined convergence threshold is not yet provided by `glmm`.

Revisting the IRLS algorithm

Here, the pseudo-response at iteration k will be defined as

$$\tilde{y}^k = \hat{\theta}^k + (y - \hat{\mu}^k)\frac{\partial \theta}{\partial \mu}|_{\hat{\theta}^k}.$$

Assume density (1) and suppose that $a_i(\phi) = \phi/w_i$. Then the log likelihood as a function of β is

$$l(\beta) \propto \frac{1}{\phi}\sum_i w_i\left[y_i\theta_i - c(\theta_i)\right],$$

where we note that our submodel is of the form $\theta_i = x_i^T\beta$. We now take a derivative with respect to components β_j:

$$\frac{\partial l(\beta)}{\partial \beta_j} = \frac{1}{\phi}\sum_i w_i\left[y_i\frac{\partial \theta_i}{\partial \beta_j} - c'(\theta_i)\frac{\partial \theta_i}{\partial \beta_j}\right],$$

where $c'(\theta_i)$ denotes the first derivative with respect to θ_i. The chain rule gives us:

$$\frac{\partial \theta_i}{\partial \beta_j} = \frac{\partial \theta_i}{\partial \mu_i}\frac{\partial \mu_i}{\partial \beta_j},$$

where, from exponential family theory, we have that $\frac{\partial \mu_i}{\partial \theta_i} = c''(\theta_i)$, and this gives us

$$\frac{\partial l(\beta)}{\partial \beta_j} = \frac{1}{\phi}\sum_i\left[\frac{y_i - c'(\theta)}{c''(\theta_i)/w_i}\right]\frac{\partial \mu_i}{\partial \beta_j}.$$

We now substitute in known quantities given to us for free from exponential family theory, and we pose the above as a maximum likelihood estimation problem which arrives at:

$$\sum_i \left[\frac{y_i - \mu_i}{\mathrm{Var}(\mu_i)}\right]\frac{\partial \mu_i}{\partial \beta_j} = 0, \qquad \text{for all } j,$$

where we note that $\mathrm{Var}(\mu_i) = c''(\theta_i)a_i(\phi)$ in the dispersed model (1). Now suppose for a minute that we know $\mathrm{Var}(\mu_i)$, then the above obtimization problem is equivalent to minimizing the WLS loss:

$$\sum_i \frac{(y_i - \mu_i)^2}{\mathrm{Var}(\mu_i)}$$

This motivates an IRLS algorithm like the one seen before where one initializes $\hat{\mu}^0$ and $\hat{\beta}^0$ and, at iteration k:

1. form the pseudo response $\tilde{y}_i^k = \hat{\theta}^k + (y - \hat{\mu}^k)\frac{\partial\theta}{\partial\mu}|_{\hat{\theta}^k}$.
2. form the weights $1/w^k = (\frac{\partial\theta}{\partial\mu})^2|_{\hat{\theta}^k}\mathrm{Var}(\hat{\mu}^k)$
3. Reestimate to get $\hat{\beta}^{k+1}$ and hence $\hat{\theta}^{k+1}$.

Repeat until convergence. See the exponential family notes, Section 8.2 of Faraway [2016], and Chapter 4 of Agresti [2013] for more details. Here's a short script to try:

```r
## use built in GLM
data(bliss, package="faraway")
?bliss
modl = glm(cbind(dead,alive) ~ conc, family=binomial, bliss)
summary(modl)$coef

## IRLS algorithm
y = bliss$dead/30; mu = y
library(faraway)
theta = logit(mu)
z = theta + (y-mu)/(mu*(1-mu))
w = 30*mu*(1-mu)
lmod = lm(z ~ conc, weights=w, bliss)

for(i in 1:5){
  theta = lmod$fit
  mu = ilogit(theta)
  z = theta + (y-mu)/(mu*(1-mu))
  w = 30*mu*(1-mu)
  lmod = lm(z ~ bliss$conc, weights=w)
  cat(i,coef(lmod),"\n")
}
```

Binary response implementations

We will first consider GLMM modeling for binary response data.

Example

An experiment was conducted to study the effects of surface and vision on balance. The balance of subjects was observed for two different surfaces and for restricted and unrestricted vision. Balance was assessed qualitatively on an ordinal four-point scale based on observation by the experimenter. Forty subjects were studied, 20 males and 20 females ranging in age from 18 to 38, with heights given in cm and weights in kg. The subjects were tested while standing on foam or a normal surface and with their eyes closed or open or with a dome placed over their head. Each subject was tested twice in each of the surface and eye combinations for a total of 12 measures per subject.

For the purposes of this analysis, we will reduce the response to a two-point scale: whether the subject was judged completely stable (=1) or not (=0).

```r
library(faraway)
library(tidyverse)
library(ggplot2)
data(ctsib)
ctsib$stable = ifelse(ctsib$CTSIB==1,1,0)
```

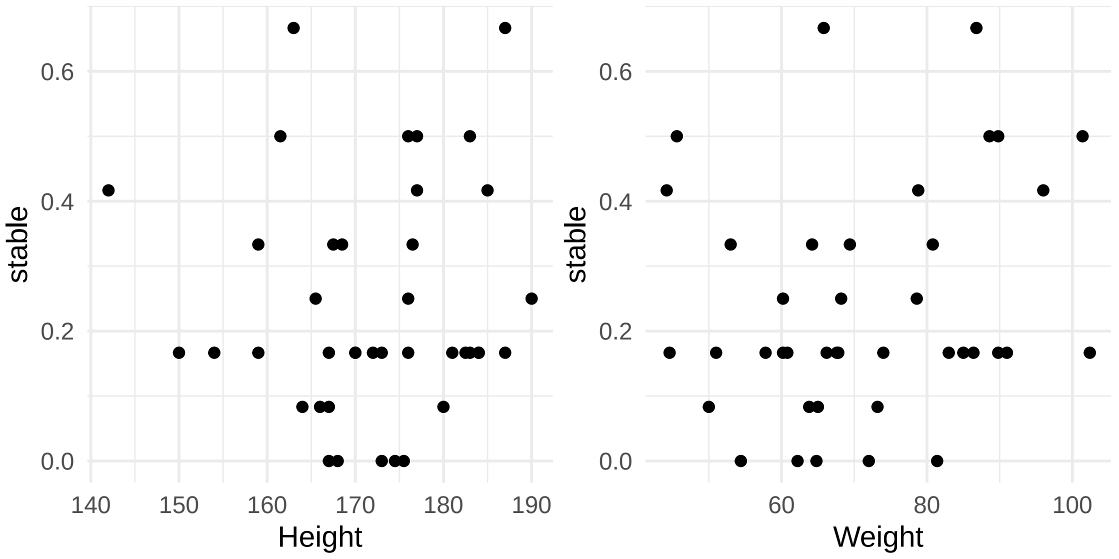
Here is the mean response for the combined conditions:

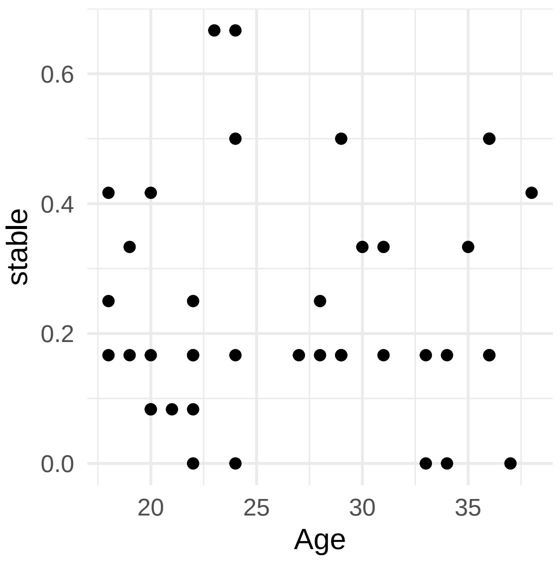
```r
xtabs(stable ~ Surface + Vision, ctsib)/80
```

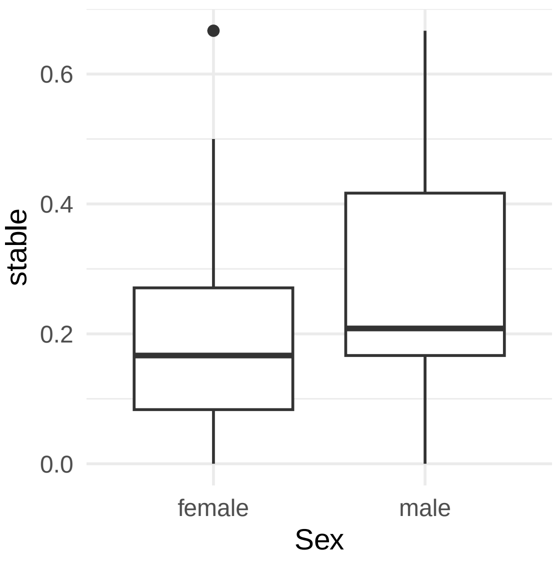
```
##        Vision
## Surface closed   dome   open
##    foam 0.0000 0.0000 0.1250
##    norm 0.2125 0.2750 0.8125
```

We see that the normal surface with open vision leads to the highest stability. We can group the data by subject and average over the 12 observations (6 conditions, replicated twice). The plots are seen below.

```r
subsum = ctsib %>% group_by(Subject) %>% summarise(Height=Height[1],
  Weight=Weight[1], stable=mean(stable), Age=Age[1], Sex=Sex[1])
par(mfrow = c(2,2))
ggplot(subsum, aes(x=Height,y=stable)) + theme_minimal() + geom_point()
ggplot(subsum, aes(x=Weight,y=stable)) + theme_minimal() + geom_point()
ggplot(subsum, aes(x=Age,y=stable)) + theme_minimal() + geom_point()
ggplot(subsum, aes(x=Sex,y=stable)) + theme_minimal() + geom_boxplot()
```







We could fit a logistic regression model that ignores subject information entirely:

```r
gf = glm(stable ~ Sex+Age+Height+Weight+Surface+Vision,binomial,data=ctsib)
sumary(gf)
```

```
##                Estimate Std. Error z value  Pr(>|z|)
## (Intercept)  7.2774488  3.8039871  1.9131 0.0557339
## Sexmale      1.4015773  0.5162309  2.7150 0.0066272
## Age          0.0025212  0.0243073  0.1037 0.9173896
## Height      -0.0964134  0.0268369 -3.5926 0.0003274
## Weight       0.0435030  0.0180016  2.4166 0.0156652
## Surfacenorm  3.9675152  0.4471789  8.8723 < 2.2e-16
## Visiondome   0.3637528  0.3832178  0.9492 0.3425157
## Visionopen   3.1875007  0.4160007  7.6622 1.827e-14
##
## n = 480 p = 8
## Deviance = 295.20261 Null Deviance = 526.25381 (Difference = 231.05120)
```

This assumes we have 480 independent observations but, in reality, we have only 40 subjects whose responses will be correlated. This analysis is likely to underestimate the standard errors and so exaggerate the significance of the experimental effects. **Why?**

PQL methods

We now try GLMM methods. First we demonstrate the PQL method implemented in the `MASS` package:

```r
## glmer's convergence is quite finicky, best to rescale predictors
ctsib = ctsib %>%
  mutate(Age = scale(Age), Height = scale(Height), Weight = scale(Weight))
library(MASS)
modpql = glmmPQL(stable ~ Sex + Age + Height + Weight + Surface +
  Vision, random=~1|Subject, family=binomial,data=ctsib)
summary(modpql)
```

```
## Linear mixed-effects model fit by maximum likelihood
##   Data: ctsib
##   AIC BIC logLik
##    NA  NA     NA
```

```
##
## Random effects:
##  Formula: ~1 | Subject
##         (Intercept)  Residual
## StdDev:    3.060712 0.5906232
##
## Variance function:
##  Structure: fixed weights
##  Formula: ~invwt
## Fixed effects:  stable ~ Sex + Age + Height + Weight + Surface + Vision
##                  Value Std.Error  DF   t-value p-value
## (Intercept) -12.494594 1.3351456 437 -9.358225  0.0000
## Sexmale       3.355340 1.7526135  35  1.914478  0.0638
## Age          -0.042189 0.5209025  35 -0.080992  0.9359
## Height       -1.998376 0.9637226  35 -2.073601  0.0455
## Weight        1.069415 0.9676606  35  1.105155  0.2766
## Surfacenorm   7.724078 0.5735776 437 13.466492  0.0000
## Visiondome    0.726464 0.3259333 437  2.228873  0.0263
## Visionopen    6.485257 0.5439796 437 11.921876  0.0000
##  Correlation:
##             (Intr) Sexmal Age    Height Weight Srfcnr Visndm
## Sexmale     -0.735
## Age         -0.083  0.110
## Height       0.330 -0.388  0.041
## Weight       0.213 -0.374 -0.168 -0.555
## Surfacenorm -0.613  0.116  0.023 -0.114  0.055
## Visiondome  -0.185  0.011  0.004 -0.017  0.011  0.087
## Visionopen  -0.606  0.125  0.026 -0.116  0.049  0.788  0.377
##
## Standardized Within-Group Residuals:
##           Min            Q1           Med            Q3           Max
## -7.3825387654 -0.2333403345 -0.0233564300 -0.0004216629  9.9310682837
##
## Number of Observations: 480
## Number of Groups: 40
```

The SD for the subject effect is 3.06. We can use the same ideas from logistic regression to interpret this value. We have exp(3.06) = 21.3 so the odds of stability are multiplied by this factor. Hence we can see that there is substantial variation in the inherent stability of individuals. Indeed, this variation is of comparable magnitude to the treatment effects.

We see strongly significant surface and vision effects while some other effects have marginally significant p-values. However, this inference is based on the linearized model and rather dubious assumptions as explained in Section 10.2 in Faraway [2016], so these results cannot be relied upon. Furthermore, the Bernoulli response may lead to biased estimates of regression coefficients.

Numerical integration

The numerical integration-based methods are implemented in the `lme4` package. The default choice of method is the Laplace approximation. We can change this to the Gauss-Hermite quadrature by specifying the `nACQ` argument to be any integer between 2 and 25. The default value of 1 corresponds to the Laplace approximation. Note that `glmer` is quite sensitive to measurement scale so we have rescaled the predictors. For more convergence issues see here.

```
library(lme4)
system.time({
  modlap = glmer(stable ~ Sex + Age + Height + Weight +
    Surface + Vision + (1|Subject), family=binomial,
    data=ctsib)
})
```

```
##    user  system elapsed
##   0.639   0.008   0.647
```

```
system.time({
  modgh = glmer(stable ~ Sex + Age + Height + Weight + Surface +
    Vision + (1|Subject), nAGQ=25, family=binomial, data=ctsib,
    control=glmerControl(optimizer="bobyqa",optCtrl=list(maxfun=2e5)))
})
```

```
##    user  system elapsed
##   0.380   0.001   0.380
```

```
summary(modgh)
```

```
## Generalized linear mixed model fit by maximum likelihood (Adaptive
##   Gauss-Hermite Quadrature, nAGQ = 25) [glmerMod]
##  Family: binomial  ( logit )
## Formula: stable ~ Sex + Age + Height + Weight + Surface + Vision + (1 |
##     Subject)
##    Data: ctsib
## Control: glmerControl(optimizer = "bobyqa", optCtrl = list(maxfun = 2e+05))
##
##       AIC       BIC    logLik -2*log(L)  df.resid
##     247.9     285.5    -114.9     229.9       471
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -5.0175 -0.1334 -0.0185 -0.0006  4.9238
##
## Random effects:
##  Groups  Name        Variance Std.Dev.
##  Subject (Intercept) 7.657    2.767
## Number of obs: 480, groups:  Subject, 40
##
## Fixed effects:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -11.94594    1.97473  -6.049 1.45e-09 ***
## Sexmale       3.18721    1.74639   1.825   0.0680 .
## Age          -0.02587    0.49899  -0.052   0.9587
## Height       -1.96489    0.95012  -2.068   0.0386 *
## Weight        1.08598    0.93139   1.166   0.2436
## Surfacenorm   7.40243    1.08472   6.824 8.84e-12 ***
## Visiondome    0.68230    0.53048   1.286   0.1984
## Visionopen    6.19147    1.00073   6.187 6.13e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) Sexmal Age    Height Weight Srfcnr Visndm
## Sexmale     -0.657
## Age         -0.038  0.097
## Height       0.400 -0.426  0.043
## Weight       0.050 -0.332 -0.167 -0.564
## Surfacenorm -0.833  0.263 -0.005 -0.263  0.125
## Visiondome  -0.242  0.035  0.001 -0.041  0.023  0.114
## Visionopen  -0.821  0.271  0.000 -0.262  0.116  0.836  0.376
```

Notice that we have AIC/BIC values for model comparison purposes. These are not available from PQL because it is not a true likelihood method. As it happens, the parameter estimates are quite similar to PQL which provides some reassurance.

We might ask whether any of the subject-specific variables have an effect. We can test this by fitting a model without these terms and comparing the two:

```
modgh2 = glmer(stable ~ Surface + Vision + (1|Subject), nAGQ=25,
                family=binomial, data=ctsib)
anova(modgh, modgh2)
```

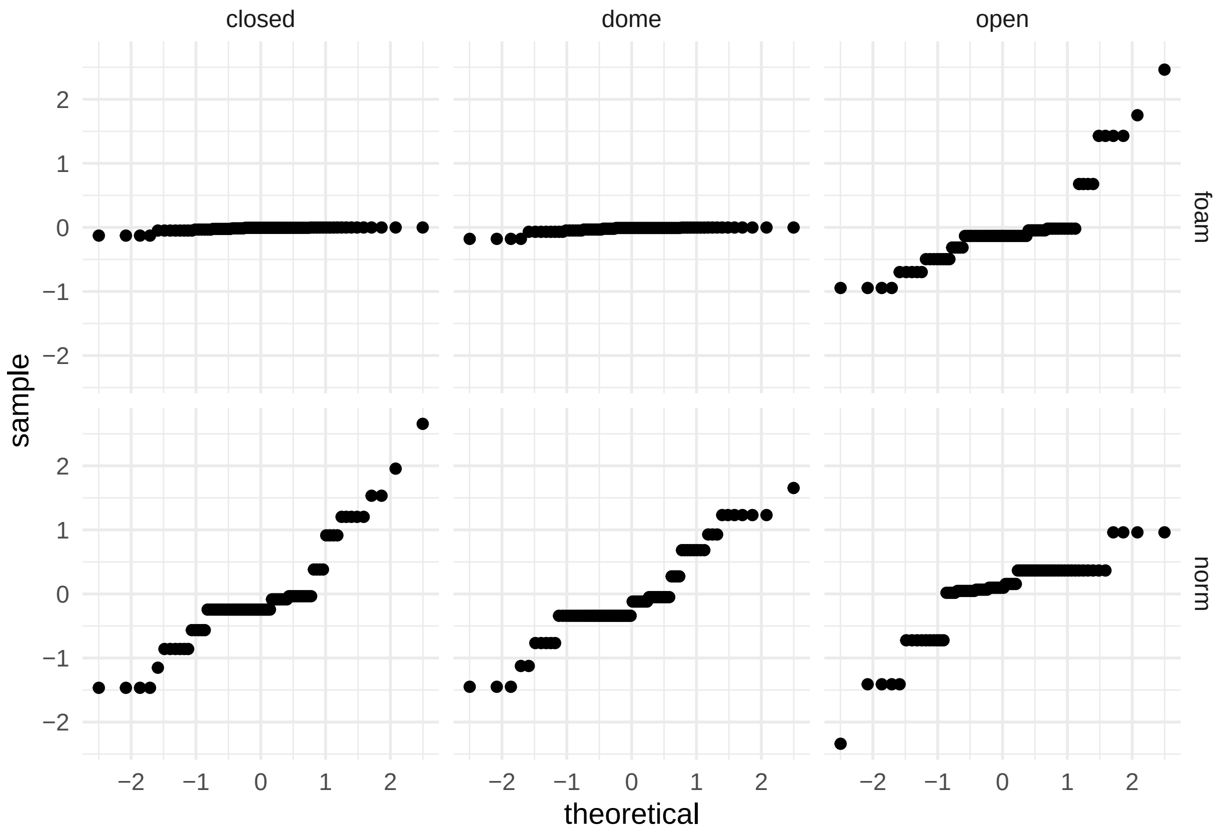
```
## Data: ctsib
## Models:
## modgh2: stable ~ Surface + Vision + (1 | Subject)
## modgh: stable ~ Sex + Age + Height + Weight + Surface + Vision + (1 | Subject)
##        npar    AIC    BIC  logLik -2*log(L) Chisq Df Pr(>Chisq)
## modgh2    5 247.30 268.17 -118.65    237.30
## modgh     9 247.89 285.46 -114.95    229.89 7.407  4     0.1159
```

This uses the standard likelihood-based methods to construct a chi-squared test. We have the same reasons as with LMMs to view these results with some skepticism.

Even so, this is a balanced experiment of a reasonable size so this provides some confidence in the result. We see that a simplification to just the treatment variables as fixed effects seems reasonable. If we feel uncomfortable with this conclusion, we may further point to the minimization of AIC (or BIC) as a justification for choosing the smaller model. One could also run a parametric bootstrap to help decide which model fits the best.

We now check model diagnostics. Here are the QQ plots subsetted by the treatment variables:

```
dd = fortify.merMod(modgh2)
ggplot(dd, aes(sample=.resid)) + stat_qq() +
  theme_minimal() + facet_grid(Surface~Vision)
```



We see that the residuals are close to zero for two of the six combinations. This is because these were universally unstable conditions and have been predicted as such by the model. In the most stable, normal and open condition, larger positive residuals are not seen because there is no headroom for such cases. It would be a mistake to view this plot as indicating heteroscedascity as we have seen there are more convincing explanations for the differences in spread.

Bayesian methods

We use INLA for a Bayesian approach to fitting generalized linear mixed-effects models. INLA is a fast, deterministic alternative to Markov Chain Monte Carlo (MCMC) for approximate Bayesian inference in latent Gaussian models (LGMs), a class that includes most GLMMs. These models consist of:

- A data model $p(y_i|\theta_i, \psi_1)$ where ψ_1 are likelihood-specific hyperparameters (dispersion for example).
- A latent Gaussian field $\theta|\psi_2 \sim N(\mu(\psi_2), Q(\psi_2)^{-1})$ where Q is a precision matrix. Precision matrices are inverse covariance matrices and they encode conditional independence which is relevant in this modeling formulation.
- Low-dimensional hyperparameters $\psi = (\psi_1, \psi_2)$

INLA approximates the marginal posteriors $p(\theta_i|y)$ and $p(\psi_j|y)$, where y is the full data set, using nested Laplace approximations. Specifically, it first approximates $p(\psi|y)$, then approximates $p(\theta_i|\psi, y)$ for each ψ, and then finally integrates over ψ numerically. This requires one to specify a prior distribution for ψ. INLA is particularly effective when ψ is low-dimensional.

The code below installs the `INLA` package.

```r
## code chuck does not run (eval = FALSE)
install.packages("INLA",repos=c(getOption("repos"),
  INLA="https://inla.r-inla-download.org/R/stable"),
  dep=TRUE)
```

See Section 12.2 in Faraway [2016] for an introduction. For ease of exposition, we use only the surface and vision as fixed effect predictors. The default, noninformative priors, are satisfactory:

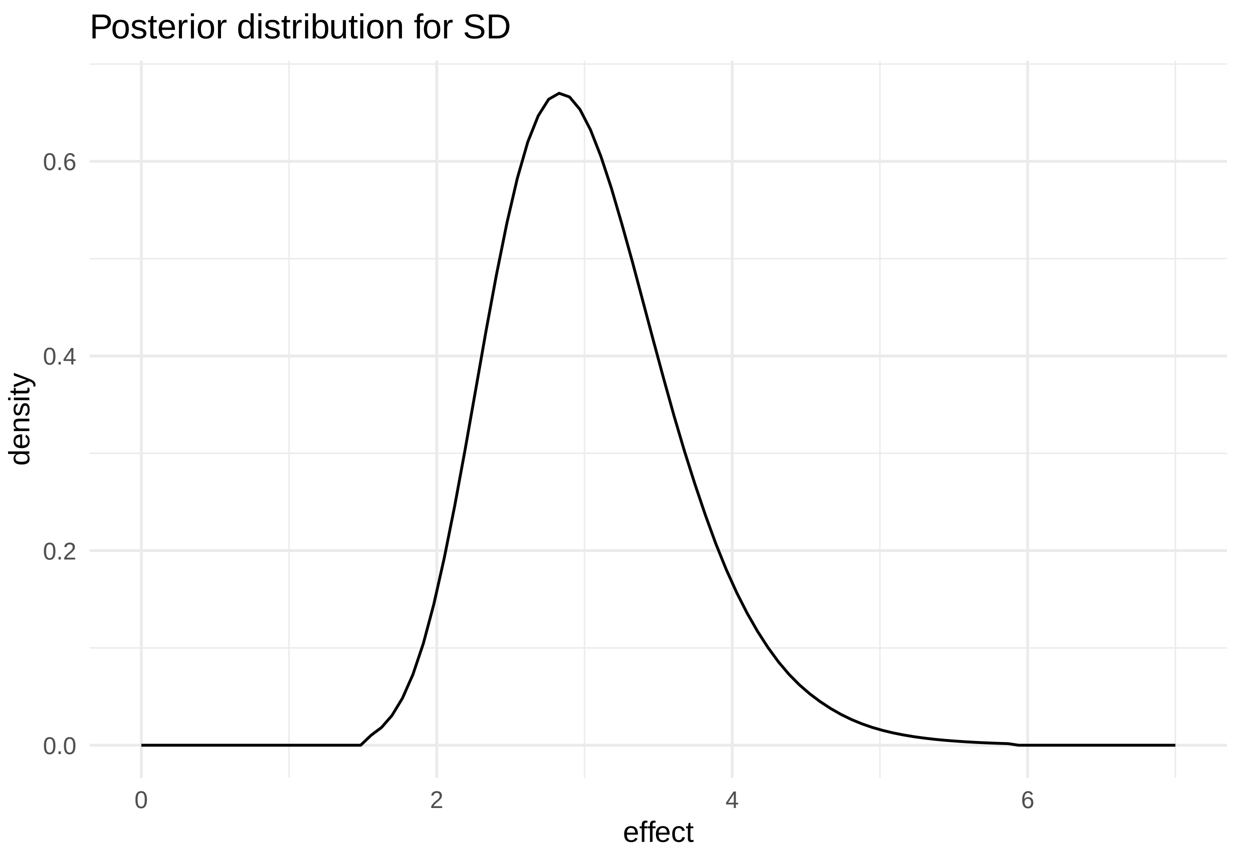
```r
library(INLA)
formula = stable ~ Surface + Vision + f(Subject, model="iid")
result = inla(formula, family="binomial", data=ctsib)
```

We compute the SD for the subject random effect:

```r
sigmaalpha = inla.tmarginal(function(x) 1/sqrt(x),
    result$marginals.hyperpar$"Precision for Subject")
```

The posterior density for this SD is shown below:

```r
x = seq(0,7,length.out = 100)
sdf = data.frame(effect = x, density=inla.dmarginal(x, sigmaalpha))
ggplot(sdf,aes(x=effect,y=density)) +
  ggtitle("Posterior distribution for SD") +
  geom_line() +
  theme_minimal()
```



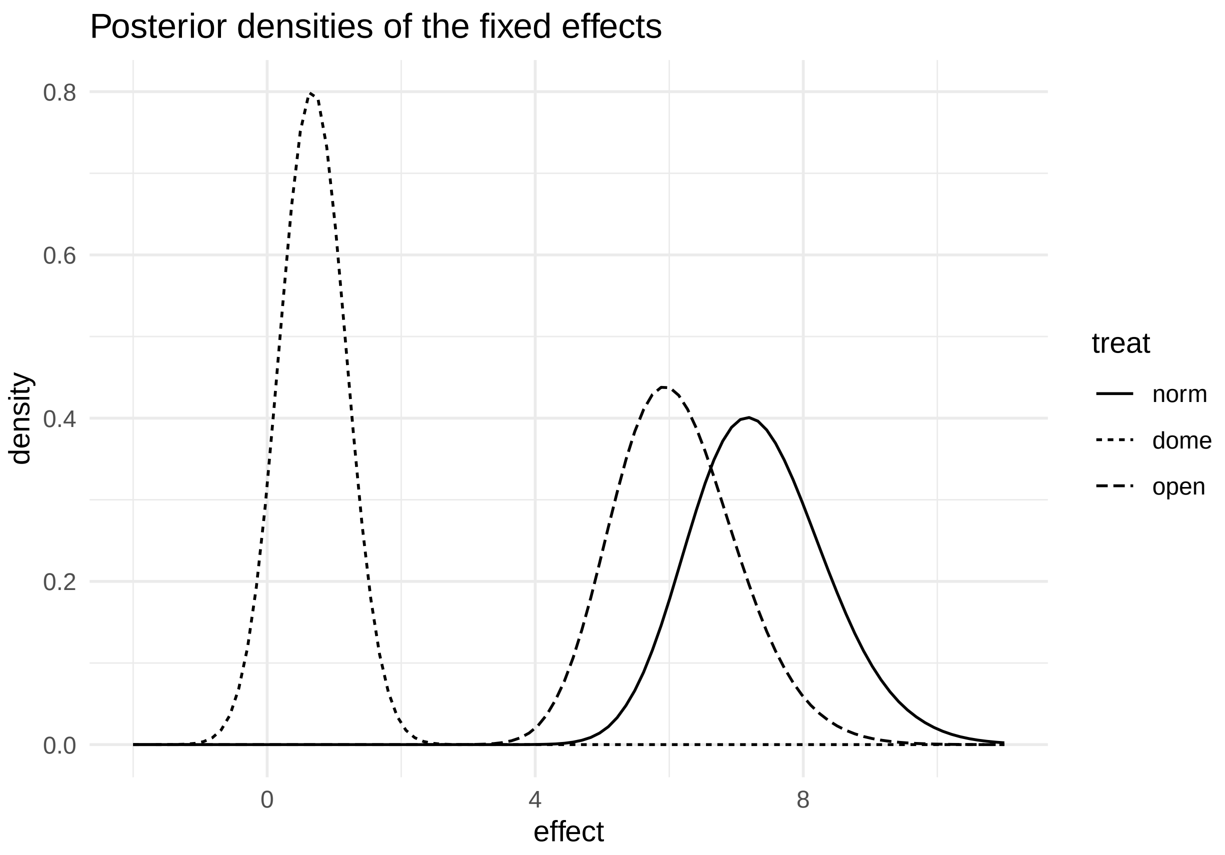
We see that the subject effect is clear since the distribution is well away from zero but there is some uncertainty regarding the size of the effect. We can produce a numerical summary of the posteriors:

```r
restab = sapply(result$marginals.fixed,
  function(x) inla.zmarginal(x, silent=TRUE))
restab = cbind(restab, inla.zmarginal(sigmaalpha,silent=TRUE))
colnames(restab) = c("intercept","norm","dome","open","SD")
data.frame(restab)
```

```
##              intercept     norm       dome      open        SD
## mean        -10.23829 7.348791  0.6809012  6.109129  3.025683
## sd            1.471018 1.011728    0.49963 0.9283375 0.6291476
## quant0.025 -13.40504 5.524635 -0.2946215  4.445377  1.978195
## quant0.25  -11.16948  6.63806   0.342213  5.456548  2.576491
## quant0.5   -10.13997 7.290154  0.6779032  6.051656  2.959888
## quant0.75   -9.205864 7.995737   1.015391  6.699071  3.401968
## quant0.975 -7.646502 9.486762   1.665337  8.082161  4.443831
```

We see that the posterior means are quite similar to the last `glmer`-based fit. We can plot the posterior densities of the fixed effects as seen below:

```
x = seq(-2,11,length.out = 100)
rden = sapply(result$marginals.fixed,
  function(y) inla.dmarginal(x, y))[,-1]
ddf = data.frame(effect=rep(x,3), density=as.vector(rden),
  treat=gl(3,100, labels=c("norm","dome","open")))
ggplot(ddf, aes(x=effect, y=density, linetype=treat)) +
  ggtitle("Posterior densities of the fixed effects") +
  theme_minimal() +
  geom_line()
```



Note: The reference level for the surface factor is `foam` and the reference level for the vision factor is `closed`. Thus the above distributions are for the effect that `norm` has over `foam` and the effects that `open` and `dome` have over `closed`. The plot does not explicitly present this information.

The `norm` level of surface and the `open` level of vision are clearly different from the respective reference levels since the densities are well separated from zero. In contrast, we see there may not be much difference between the `dome` and `closed` levels of vision as this density overlaps zero. We can compute a “Bayesian p-value” as:

```r
2*inla.pmarginal(0, result$marginals.fixed$Visiondome)
```

```
## [1] 0.17231
```

We have multiplied by two to account for the usual two-sided testing argument. In this context, p-values do not have the same meaning (Exercise for the reader: Why does this Bayesian p-value not have the same meaning as the conventional p-value? What is an intuitive meaning for this Bayesian p-value?). Nonetheless, it does serve as a measure of how the posterior density relates to zero. This confirms our impression that there is not much difference between the levels.

Monte Carlo likelihood approximation

With the release of the R package `glmm`, frequentist likelihood-based inference is available in Binomial or Poisson GLMMs, including calculating Fisher information, hypothesis testing, and constructing likelihood-based confidence intervals. It is the only publicly-available software that accurately conducts likelihood-based inference for GLMMs for both nested and crossed random effects. The `glmm` package uses Monte Carlo likelihood approximation (MCLA) to approximate the likelihood function [Geyer, 1994, Geyer and Thompson, 1992]. MCLA is an instance of importance sampling and the essential innovation that allowed effective use of MCLA in `glmm` is the development of a novel importance sampling distribution.

We will reexpress the likelihood of the GLMM in terms of the response vector y rather than a product of independent components y_i,

$$L(\beta, \phi, V|y) = \int f(y|\beta, \phi, b)h(b|V)db = \int f(y, b|\beta, \phi, V)db. \tag{2}$$

It is important to recognize that the likelihood that we wish to maximize requires one to integrate out the random effects. MCLA uses importance sampling to approximate the above integral. This technique requires an importance sampling distribution, which we will denote as $\tilde{f}(b)$. With this in mind, we can rewrite (2) as

$$L(\beta, \phi, V|y) = \mathrm{E}_{\tilde{f}}\left[\frac{f(y, b|\beta, \phi, V)}{\tilde{f}(b)}\right].$$

If $b_1, \ldots, b_m$ are realizations from $\tilde{f}$, then the Monte Carlo likelihood approximation is

$$L_m(\beta, \phi, V|y) = \frac{1}{m}\sum_{k=1}^{m}\frac{f(y, b_k|\beta, \phi, V)}{\tilde{f}(b_k)}. \tag{3}$$

We then maximize the Monte Carlo likelihood approximation (3). The maximizer of (3) is called the Monte Carlo maximum likelihood estimator (MCMLE). Under a Wald-type integrability condition, the likelihood approximation converges almost surely to the likelihood function and the MCMLE converges to the MLE almost surely as the Monte Carlo sample size increases [Geyer, 1994].

The `glmm` package constructs a novel importance sampling distribution $\tilde{f}$ to compute (3) and obtain the MCMLE. We now run MCLA using `glmm`.

```r
library(glmm)
# subject needs to be a factor
ctsib$SubjectF = as.factor(ctsib$Subject)

set.seed(13)
nCores = detectCores() - 2
clust = makeCluster(nCores)
system.time({
  m1 = glmm(stable ~ Sex + Age + Height +
    # random effect distribution centered at 0; no reference group
    Weight + Surface + Vision, random = list(~0+SubjectF),
    family.glmm=bernoulli.glmm, m = 2e4,
    varcomps.names = c("Subject"), cluster = clust,
    data=ctsib)
})
```

```
##    user  system elapsed
##  11.767   0.520 144.590
```

```
summary(m1)
```

```
##
## Call:
## glmm(fixed = stable ~ Sex + Age + Height + Weight + Surface +
##     Vision, random = list(~0 + SubjectF), varcomps.names = c("Subject"),
##     data = ctsib, family.glmm = bernoulli.glmm, m = 20000, cluster = clust)
##
##
## Link is: "logit (log odds)"
##
## Fixed Effects:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -11.250      0.840 -13.450  < 2e-16 ***
## Sexmale        3.860      0.700   5.512 3.54e-08 ***
## Age           -0.240      0.200  -1.165  0.24418
## Height        -2.370      0.390  -6.085 1.17e-09 ***
## Weight         1.300      0.400   3.213  0.00131 **
## Surfacenorm    6.700      0.600  11.117  < 2e-16 ***
## Visiondome     0.586      0.487   1.204  0.22873
## Visionopen     5.630      0.560   9.973  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Variance Components for Random Effects (P-values are one-tailed):
##         Estimate Std. Error z value Pr(>|z|)/2
## Subject      6.8        1.6   4.164   1.57e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Note that the above code allowed for one to specify a cluster as an argument of the `glmm` fitting function. This greatly speeds up the MCLA technique and is valid since Monte Carlo simulation is parallelizable (each iteration is independent of all other iterations). Also note that the p-value for the fixed effects is calculated using a two-sided alternative hypothesis ($H_A : \beta \neq 0$) while the p-value for the variance components is calculated using a one-sided alternative hypothesis ($H_A : \nu > 0$) because variance components must be nonnegative.

With maximum likelihood performed by Monte Carlo likelihood approximation, there are two sources of variability: there is variability inherent in the sample and from the Monte Carlo sample (of generated random effects). The first source of variability is measured using the usual standard error. The second source of variability is assessed by the Monte Carlo standard error. The standard error decreases as the (observed data) sample size increases; similarly, the Monte Carlo standard error decreases as the Monte Carlo sample size increases. These errors are displayed below.

```
# standard error
se_glmm = se(m1)
se_glmm
```

```
## (Intercept)     Sexmale         Age      Height      Weight Surfacenorm
##   0.8365953   0.7006532   0.2036920   0.3891351   0.4047227   0.6030634
##  Visiondome  Visionopen     Subject
##   0.4865684   0.5641288   1.6342016
```

```
# Monte Carlo standard error
MCse_glmm = mcse(m1)
MCse_glmm
```

```
## (Intercept)     Sexmale         Age      Height      Weight Surfacenorm
## 0.026046324 0.031947887 0.012390847 0.029795903 0.028710716 0.033854870
##  Visiondome  Visionopen     Subject
## 0.005176514 0.040118257 0.216516230
```

Users can inspect the Monte Carlo standard error to decide whether the Monte Carlo sample was large enough: each Monte Carlo standard error should be small compared to its corresponding standard error.

```
se_glmm / MCse_glmm
```

```
## (Intercept)     Sexmale         Age      Height      Weight Surfacenorm
##   32.119516   21.931128   16.438907   13.060021   14.096571   17.813195
##  Visiondome  Visionopen     Subject
##   93.995372   14.061647    7.547709
```

Count response implementations

We now consider GLMM modeling for count response data.

Example

In this example, we have data from a clinical trial of 59 epileptics. For a baseline, patients were observed for 8 weeks and the number of seizures recorded. The patients were then randomized to treatment by the drug Progabide (31 patients) or to the placebo group (28 patients). They were observed for four 2-week periods and the number of seizures recorded. We are interested in determining whether Progabide reduces the rate of seizures.

We first perform some data manipulations and then look at the first two patients:

```
data(epilepsy, package="faraway")
epilepsy$period = rep(0:4, 59)
epilepsy$drug = factor(c("placebo","treatment")[epilepsy$treat+1])
epilepsy$phase = factor(c("baseline","experiment")[epilepsy$expind +1])
epilepsy[epilepsy$id < 2.5,]
```

```
##    seizures id treat expind timeadj age period    drug      phase
## 1        11  1     0      0       8  31      0 placebo   baseline
## 2         5  1     0      1       2  31      1 placebo experiment
## 3         3  1     0      1       2  31      2 placebo experiment
## 4         3  1     0      1       2  31      3 placebo experiment
## 5         3  1     0      1       2  31      4 placebo experiment
## 6        11  2     0      0       8  30      0 placebo   baseline
## 7         3  2     0      1       2  30      1 placebo experiment
## 8         5  2     0      1       2  30      2 placebo experiment
## 9         3  2     0      1       2  30      3 placebo experiment
## 10        3  2     0      1       2  30      4 placebo experiment
```

Both of these individuals were not treated. The `expind` variable indicates the baseline phase by 0 and the treatment phase by 1. The length of these time phases is recorded in the `timeadj` variable. Three new convenience variables are created: `period`, denoting the 2- or 8- week periods, `drug` recording the type of treatment in nonnumeric form and `phase` indicating the phase of the experiment.

We now compute the mean number of seizures per week broken down by the treatment and baseline vs. experimental period.

```
epilepsy %>%
  group_by(drug, phase) %>%
  summarise(rate=mean(seizures/timeadj)) %>%
xtabs(formula=rate ~ phase + drug)
```

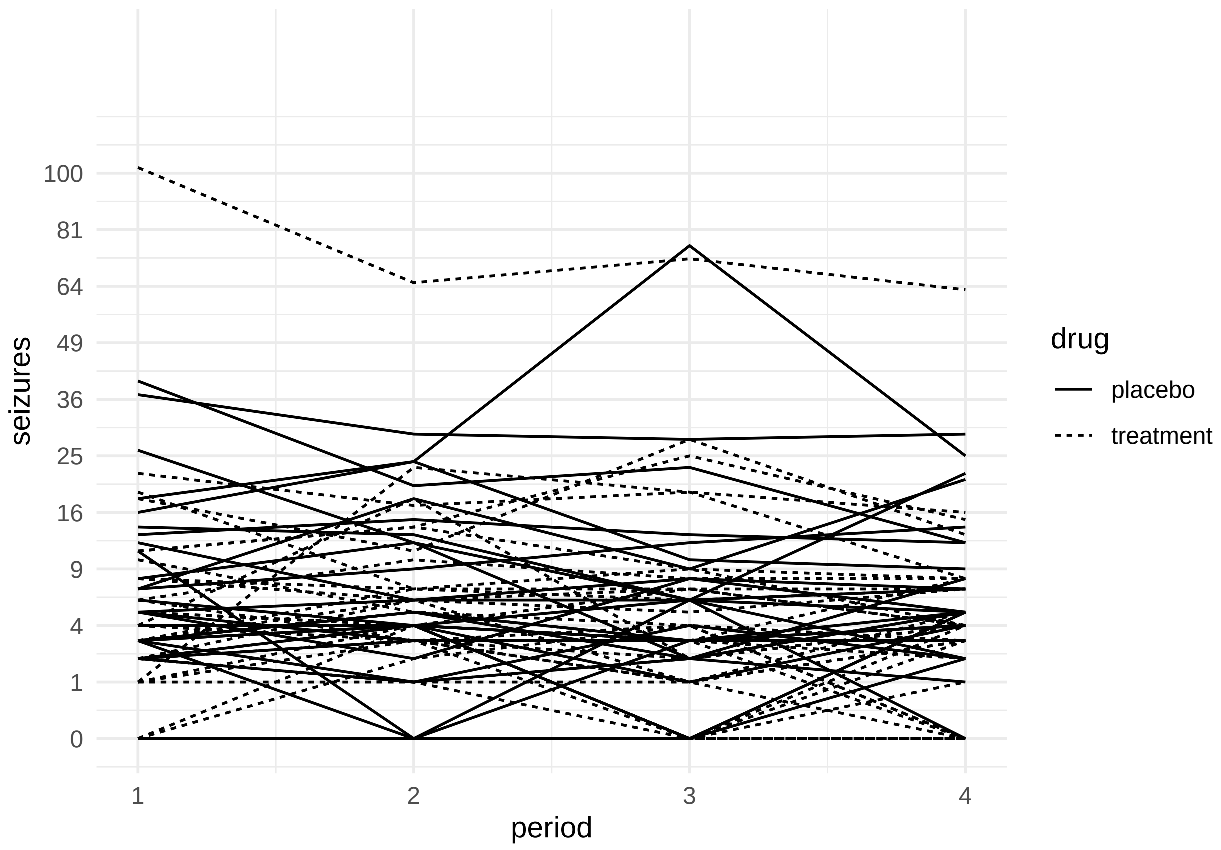
```
##               drug
## phase          placebo treatment
##   baseline   3.848214  3.955645
##   experiment 4.303571  3.983871
```

We see that the rate of seizures in the treatment group actually increases during the period in which the drug was taken. The rate of seizures increases even more in the placebo group. Perhaps some other factor is causing the rate of seizures to increase during the treatment period and the drug is actually having a beneficial effect.

Now we make some plots to show the difference between the treatment and the control. The first plot shows the difference between the two groups during the experimental period only:

```
ggplot(epilepsy, aes(x=period, y=seizures, linetype=drug, group=id)) +
  geom_line() +
  xlim(1,4) +
  scale_y_sqrt(breaks=(0:10)^2) +
  theme(legend.position = "top", legend.direction = "horizontal") +
  theme_minimal()
```

```
## Warning: Removed 59 rows containing missing values or values outside the scale range
## (`geom_line()`).
```

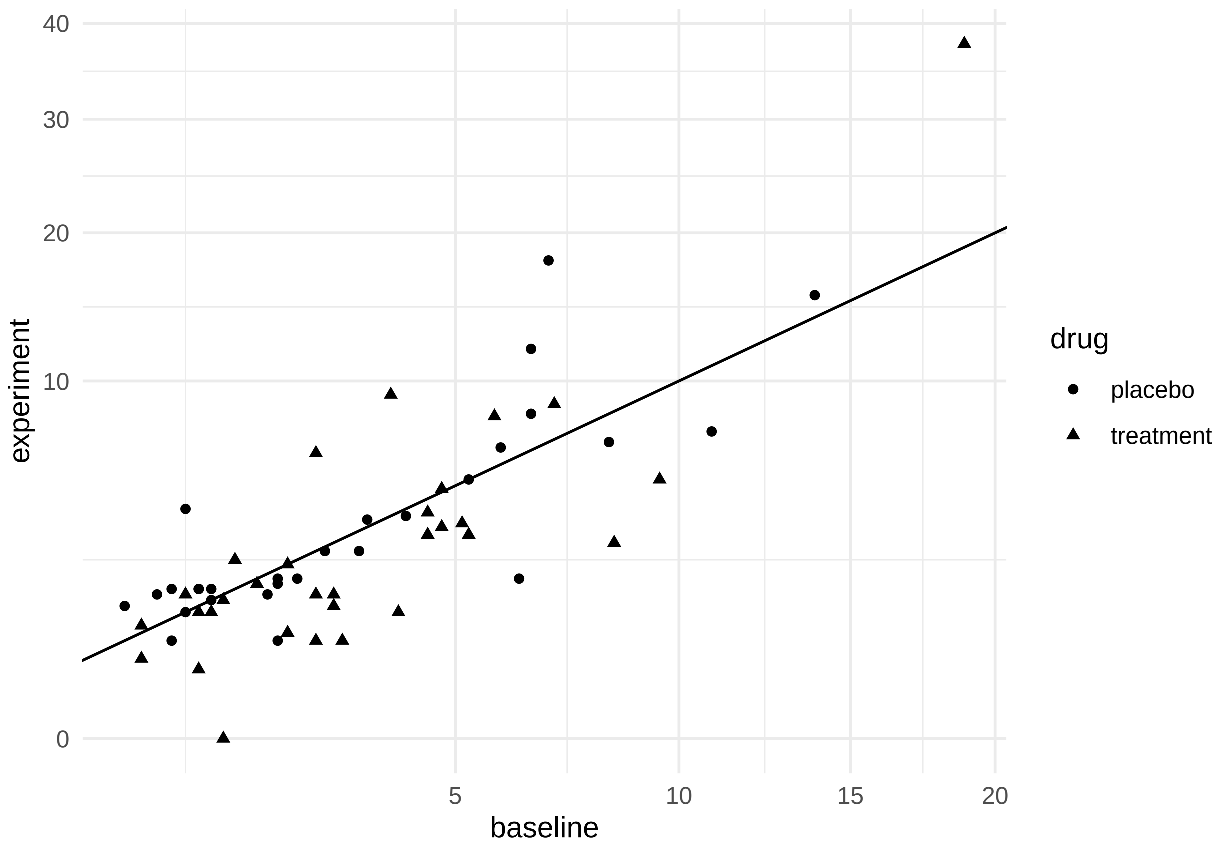


We now compare the average seizure rate to the baseline for the two groups. The square-root transform is used to stabilize the variance; this is often used with count data.

```
ratesum = epilepsy %>%
  group_by(id, phase, drug) %>%
  summarise(rate=mean(seizures/timeadj))
```

```
## `summarise()` has grouped output by 'id', 'phase'. You can override using the
## `.groups` argument.
```

```
comsum = spread(ratesum, phase, rate)
ggplot(comsum, aes(x=baseline, y=experiment, shape=drug)) +
  geom_point() +
  scale_x_sqrt() +
  scale_y_sqrt() +
  geom_abline(intercept=0, slope=1) +
  theme(legend.position = "top", legend.direction = "horizontal") +
  theme_minimal()
```



A treatment effect, if one exists, is not readily apparent. Now we fit GLMM models. Patient #49 is unusual because of the high rate of seizures observed. We exclude it:

```
epilo = filter(epilepsy, id != 49)
```

Excluding a case should not be taken lightly. For projects where the analyst works with producers of the data, it will be possible to discuss substantive reasons for excluding cases.

It is worth starting with a GLM even though the model is not correct due to the grouping of the observations. We must use an offset to study seizure rate instead of seizures. This is due to data collection having different time windows at different periods in the study design.

```
modglm = glm(seizures ~offset(log(timeadj)) + expind + treat +
  I(expind*treat), family=poisson, data=epilo)
summary(modglm)
```

```
##
## Call:
## glm(formula = seizures ~ offset(log(timeadj)) + expind + treat +
##     I(expind * treat), family = poisson, data = epilo)
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)        1.34761    0.03406  39.566  < 2e-16 ***
## expind             0.11184    0.04688   2.386    0.017 *
## treat             -0.10682    0.04863  -2.197    0.028 *
## I(expind * treat) -0.30238    0.06971  -4.338 1.44e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 2485.1  on 289  degrees of freedom
## Residual deviance: 2411.5  on 286  degrees of freedom
## AIC: 3449.7
##
## Number of Fisher Scoring iterations: 5
```

The interaction term is the primary parameter of interest. All the subjects were untreated in the baseline, even the ones who were subsequently treated. This means that the main effect for treatment does not properly measure the response to treatment because it includes the baseline period.

As we have observed already, we suspect the response may have been different during the baseline time and the active period of the experiment. The interaction term represents the effect of the treatment during the baseline period after adjustment. In the output above we see that this interaction seems highly significant and negative (which is good since we want to reduce seizures).

But this inference is suspect because we have made no allowance for the correlated responses within individuals. The p-value is far smaller than it should be. We might also consider allowing for overdispersion in the response by using a quasi-Poisson model. However, this is a different consideration to the correlated response.

PQL methods

We move through the estimation options in the same order as with the binary response example earlier, starting with PQL:

```r
modpql = glmmPQL(seizures ~offset(log(timeadj)) + expind + treat +
  I(expind*treat), random = ~1|id, family=poisson, data=epilo)
summary(modpql)
```

```
## Linear mixed-effects model fit by maximum likelihood
##   Data: epilo
##   AIC BIC logLik
##    NA  NA     NA
##
## Random effects:
##  Formula: ~1 | id
##         (Intercept) Residual
## StdDev:   0.6820012 1.605385
##
## Variance function:
##  Structure: fixed weights
##  Formula: ~invwt
## Fixed effects:  seizures ~ offset(log(timeadj)) + expind + treat + I(expind *      treat)
##                         Value  Std.Error  DF   t-value p-value
## (Intercept)         1.0761832 0.09990514 230 10.772050  0.0000
## expind              0.1125119 0.07412152 230  1.517939  0.1304
## I(expind * treat) -0.3037615 0.10819095 230 -2.807642  0.0054
##  Correlation:
##                   (Intr) expind
## expind            -0.198
## I(expind * treat) -0.014 -0.656
##
## Standardized Within-Group Residuals:
##        Min         Q1        Med         Q3        Max
## -2.2934834 -0.5649468 -0.1492931  0.3224895  6.3123337
##
## Number of Observations: 290
## Number of Groups: 58
```

The parameter estimates from the PQL fit are comparable to the GLM fit. However, the standard errors are larger in the PQL fit as might be expected given that the correlated responses have been allowed for. As with the binary response example, we still have some doubts about the accuracy of the inference. This is a particular concern when some count responses are small.

Numerical integration

Numerical quadrature can also be used. We use Gauss-Hermite in preference to Laplace as the model random effect structure is simple and so the computation is fast even though we have used the most expensive `nAGQ=25` setting.

```
modgh = glmer(seizures ~offset(log(timeadj)) + expind + treat +
  I(expind*treat)+ (1|id), nAGQ=25, family=poisson, data=epilo)
summary(modgh)
```

```
## Generalized linear mixed model fit by maximum likelihood (Adaptive
##   Gauss-Hermite Quadrature, nAGQ = 25) [glmerMod]
##  Family: poisson  ( log )
## Formula: seizures ~ offset(log(timeadj)) + expind + treat + I(expind *
##     treat) + (1 | id)
##    Data: epilo
##
##      AIC       BIC    logLik -2*log(L)  df.resid
##     877.7     896.1    -433.9     867.7       285
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.8724 -0.8482 -0.1722  0.5697  9.8941
##
## Random effects:
##  Groups Name        Variance Std.Dev.
##  id     (Intercept) 0.515    0.7176
## Number of obs: 290, groups:  id, 58
##
## Fixed effects:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)         1.035998   0.141256   7.334 2.23e-13 ***
## expind              0.111838   0.046877   2.386    0.017 *
## treat              -0.008152   0.196524  -0.041    0.967
## I(expind * treat) -0.302387   0.069714  -4.338 1.44e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##              (Intr) expind treat
## expind      -0.175
## treat       -0.718  0.126
## I(xpnd*trt)  0.118 -0.672 -0.173
```

We see that the interaction effect is significant. Notice that the estimate of this effect has been quite consistent over all the estimation methods so we draw some confidence from this. We have

```
exp(-0.302)
```

```
## [1] 0.7393381
```

So the drug is estimated to reduce the rate of seizures by about 26%. However, the subject SD is more than twice the drug effect of -0.3 at 0.718. This indicates that the expected improvement in the drug is substantially less than the variation between individuals.

Interpretation of the main effect terms is problematic in the presence of an interaction. For example, the treatment effect reported here represents the predicted difference in the response during the baseline period (i.e., `expind=0`). Since none of the subjects are treated during the baseline period, we are reassured to see that this effect is not significant. However, this does illustrate the danger in naively presuming that this is the treatment effect.

Bayesian methods

We can also take a Bayesian approach using `INLA`.

```
formula = seizures ~ offset(log(timeadj)) + expind + treat +
  I(expind*treat) + f(id,model="iid")
result = inla(formula, family="poisson", data = epilo)
```

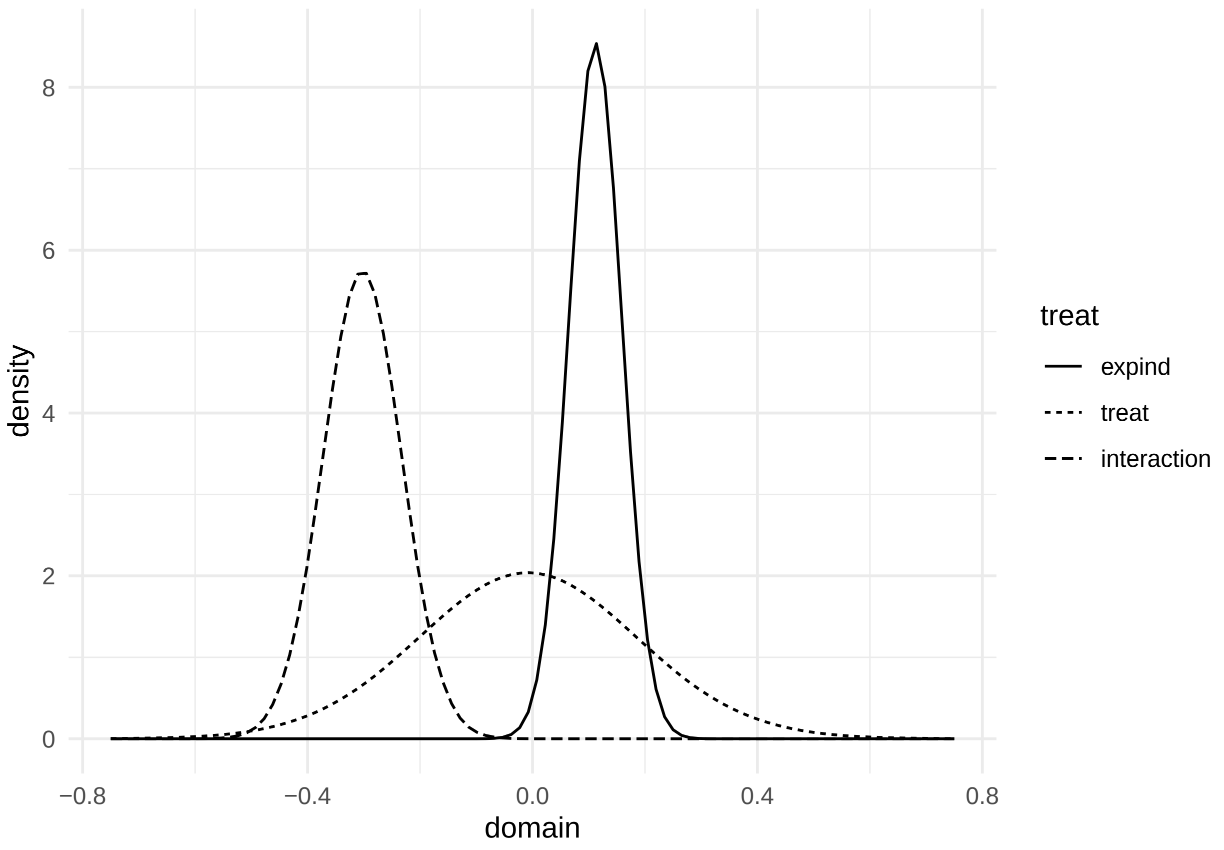
We obtain a summary of the posteriors as:

```
sigmaalpha = inla.tmarginal(function(x) 1/sqrt(x),
  result$marginals.hyperpar$"Precision for id")
restab = sapply(result$marginals.fixed,
  function(x) inla.zmarginal(x, silent=TRUE))
restab = cbind(restab,
  inla.zmarginal(sigmaalpha, silent=TRUE))
colnames(restab) = c("mu","expind","treat",
  "interaction","alpha")
data.frame(restab)
```

```
##                    mu     expind        treat interaction      alpha
## mean         1.035894  0.1118965  -0.00825535  -0.3025702  0.7256715
## sd          0.1428916 0.04685183    0.1987886  0.06967632 0.07183045
## quant0.025  0.7534163 0.01995062   -0.3997304  -0.4393087   0.599834
## quant0.25   0.9404007 0.08018645   -0.1413901  -0.3497282  0.6747957
## quant0.5     1.035962  0.1117989 -0.008693313  -0.3027154   0.720221
## quant0.75     1.13117  0.1434113     0.124003  -0.2557025  0.7705117
## quant0.975   1.315566  0.2036472    0.3823328   -0.166122  0.8818145
```

We see that the results are similar to those obtained previously. We observe that the 95% credible interval for the interaction is (-0.44,-0.17) so we are sure that this parameter differs from zero. We compute similar plots as we did in the binary response example.

```
x = seq(-0.75,0.75,length.out = 100)
rden = sapply(result$marginals.fixed,function(y) inla.dmarginal(x, y))[,-1]
ddf = data.frame(domain=rep(x,3),
  density=as.vector(rden),
  treat=gl(3,100, labels=c("expind","treat","interaction")))
ggplot(ddf, aes(x=domain, y=density, linetype=treat)) +
  geom_line() +
  theme_minimal()
```



Monte Carlo likelihood approximation

We can use the `glmm` package to implement the MCLA approach to fitting GLMM models with Poisson responses.

```
epilo$idF = as.factor(epilo$id)
epilo$seizures = as.integer(epilo$seizures)
set.seed(13)
nCores = detectCores() - 2
clust = makeCluster(nCores)
system.time({
  m1 = glmm(seizures ~ offset(log(timeadj)) +
    expind + treat + I(expind*treat), random = list(~0+idF),
    family.glmm = poisson.glmm, m = 7e4,
    varcomps.names = c("idF"), cluster = clust, data=epilo)
})
```

```
##    user  system elapsed
##   5.794   0.667  65.580
```

We obtain summary information. However, the fit is buggy. The Monte Carlo standard error is not returned and the summary table estimates are not depicted.

```
# summary table (takes awhile to load)
summary(m1)
```

```
##
## Call:
## glmm(fixed = seizures ~ offset(log(timeadj)) + expind + treat +
##     I(expind * treat), random = list(~0 + idF), varcomps.names = c("idF"),
##     data = epilo, family.glmm = poisson.glmm, m = 70000, cluster = clust)
##
##
## Link is: "log"
##
## Fixed Effects:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)              0          0  31.010  < 2e-16 ***
## expind                   0          0 -27.187  < 2e-16 ***
## treat                    0          0   0.018    0.986
## I(expind * treat)        0          0  -4.338 1.44e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Variance Components for Random Effects (P-values are one-tailed):
##     Estimate Std. Error z value Pr(>|z|)/2
## idF        0          0   5.098   1.72e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Monte Carlo standard errors
mcse_glmm = mcse(m1)
mcse_glmm
```

```
##       (Intercept)            expind             treat I(expind * treat)
##               NaN               NaN               NaN               NaN
##               idF
##               NaN
```

That being said, we can obtain estimates of fixed effects and their standard errors from objects in the `glmm` object.

```
# standard errors
se_glmm = se(m1)

# table for fixed effects
tab = cbind(m1$beta, se_glmm[-5], m1$beta/se_glmm[-5])
colnames(tab) = c("Estimate", "Std. Error", "z value")
round(tab, 3)
```

```
##                   Estimate Std. Error z value
## (Intercept)          3.106      0.100  31.010
## expind              -1.274      0.047 -27.187
## treat                0.003      0.185   0.018
## I(expind * treat)   -0.302      0.070  -4.338
```

We can also obtain estimates of random effect parameters and their standard errors from objects in the `glmm` object.

```
c(m1$nu, se_glmm[5])
```

```
##       idF       idF
## 0.5152511 0.1010674
```

Parameter estimates are similar to the other fitting techniques which instills confidence.

Generalized Estimating Equations (GEE)

We will now develop a quasi-likelihood approach that is called generalized estimating equations. The advantage of this approach is that one does not to make strong distributional assumptions on the response variable in order to estimate model parameters. Instead, we only need to specify the link function and the variance.

In this section we will let Y_i, $i = 1, \ldots, N$ be a vector of random variables representing the responses on a given individual or cluster, and let $E(Y_i) = \mu_i$ which is then linked to the linear predictor using $\mu_i = g(x_i^T \beta)$ where g is a change of parameters mapping. We will also specify a variance function a that satisfies

$$\mathrm{Var}(Y_i) = \phi a(\mu_i)$$

Appropriate choices of the a function will depend on the type of response being modeled. The ϕ term is a scale parameter.

We also must specify how the responses within an individual or cluster are correlated with each other. We set a *working correlation matrix* $R_i(\alpha)$ depending on a parameter α which requires estimation. This yields a *working covariance matrix* for Y_i:

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

where A_i is a diagonal matrix formed from $a(\mu_i)$.

Provided estimates of ϕ and α, we can estimate β by setting the multivariate score function equal to zero and solving:

$$\sum_{i=1} D_i^T V_i^{-1} (Y_i - \mu_i) = 0.$$

where $D_i = \frac{\partial \mu_i}{\partial \beta}$. These are called generalized estimating equations [Liang and Zeger, 1986].

Since $\mathrm{Var}(Y)$ also depends on α, we can substitute any $\sqrt{N}$ consistent estimate of α in this equation and still obtain an estimate as asymptotically efficient as if α were known. We will express this estimator as $\hat{\alpha} = \hat{\alpha}(\beta, \phi)$. Except for particular choices of R and α, the scale parameter ϕ will generally remain in the GEE. To complete the process, we replace ϕ by $\hat{\phi}(Y, \beta)$, a $\sqrt{N}$ consistent estimator when β is known. Let $U_i(\beta, \alpha) = D_i^T V_i^{-1} (Y_i - \mu_i)$, and with these estimators we have

$$\sum_{i=1}^{N} U_i(\beta, \hat{\alpha}(\beta, \hat{\phi}(\beta))) = 0. \tag{4}$$

The above estimation (4) can be performed iteratively where one alternates between estimating β with $(\hat{\alpha}, \hat{\phi})$ fixed, and estimating (α, ϕ) with $\hat{\beta}$ fixed. The solution of the above equation $\hat{\beta}$ has an asymptotic multivariate normal distribution of the form

$$\sqrt{N}\left(\hat{\beta} - \beta\right) \to N(0, \Sigma)$$

where $\Sigma = \lim_{N \to \infty} = N \Sigma_0^{-1} \Sigma_1 \Sigma_0^{-1}$, and

$$\Sigma_0 = \sum_{i=1}^{N} D_i^T V_i D_i, \qquad \Sigma_1 = \sum_{i=1}^{N} D_i^T V_i^{-1} \mathrm{Cov}(Y_i) V_i^{-1} D_i.$$

Replacing β, ϕ, and α by consistent estimates and the covariance matrix $\mathrm{Cov}(Y_i)$ by $(Y_i - \mu_i)(Y_i - \mu_i)^T$ in the above yields the sandwich estimate $\widehat{\Sigma}$ of Σ. The estimate $\widehat{\Sigma}$ is a consistent estimate of Σ even if the working correlation matrices $R_i(\alpha)$ are misspecified. For instance, it is often convenient to use a working independence model where $R_i(\alpha) = I$. Other popular choices include compound symmetry (i.e., exchangeable) with $R_{ij} = \alpha$ for any $i \neq j$ or first-order autoregressive with $R_{ij} = \alpha^{|i-j|}$, where R_{ij} denotes the (i,j)th element of R [Pan, 2001].

We will use the `geepack` package [Halekoh et al., 2006] to fit GEEs. We will reanalyze the stability dataset using generalized estimating equations.

```
library(geepack)
```

```
##
## Attaching package: 'geepack'
## The following object is masked from 'package:faraway':
##
##     ohio
```

```
modgeep = geeglm(stable ~ Sex + Age + Height + Weight + Surface + Vision,
                 id=Subject, corstr="exchangeable", scale.fix=TRUE,
                 data=ctsib, family=binomial)
```

We have specified the same fixed effects as in the corresponding GLMM earlier. Only simple groups are allowed while nested grouping variables cannot be accommodated easily in this function.

We are required to choose the correlation structure within each group. If we choose no correlation, then the problem reduces to a standard GLM. For this data, compound symmetry is selected as a covariance structure, since it seems reasonable that any pair of observations between subjects has the same correlation (ignoring a learning effect). Note that compound symmetry is referred to as exchangeable correlation in the `corstr` argument of the `geeglm` fitting function. Also note that we have chosen to fix ϕ at the default value of 1 to ensure that our analysis is comparable with the GLMM fit. Otherwise, there would not be a strong reason to fix this.

Here is the summary information:

```
summary(modgeep)
```

```
##
## Call:
## geeglm(formula = stable ~ Sex + Age + Height + Weight + Surface +
##     Vision, family = binomial, data = ctsib, id = Subject, corstr = "exchangeable",
##     scale.fix = TRUE)
##
##  Coefficients:
##             Estimate  Std.err   Wald Pr(>|W|)
## (Intercept) -6.16128  1.08020 32.534 1.17e-08 ***
## Sexmale      1.64487  0.90347  3.315   0.0687 .
## Age         -0.07659  0.30521  0.063   0.8019
## Height      -1.06930  0.44398  5.801   0.0160 *
## Weight       0.67199  0.52329  1.649   0.1991
## Surfacenorm  3.91631  0.56682 47.738 4.87e-12 ***
## Visiondome   0.35888  0.40403  0.789   0.3744
## Visionopen   3.17990  0.46063 47.657 5.08e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation structure = exchangeable
## Scale is fixed.
##
##   Link = identity
##
## Estimated Correlation Parameters:
##       Estimate Std.err
## alpha   0.2185 0.04467
## Number of clusters:   40  Maximum cluster size: 12
```

We can see that the estimated correlation between observations on the same subject is 0.22 with a standard error of 0.04. This suggests that there is correlation between responses within individuals. The standard errors are constructed using a sandwich estimator mentioned above. Further motivation for sandwich estimation is described in Section 8.5 of [Faraway, 2016]. Note that sandwich estimation typically, but not always, leads to standard errors larger than those obtained directly from likelihood calculations. There is no free lunch.

These standard errors can be used to construct Wald statistics. We see that the treatment factors, surface and vision, are significant. Height and possibly gender are marginally significant. This part of the conclusion is similar to our GLMM results.

There is one clear difference with the GLMM output: the estimates for the GEE are about half the size of the GLMM β. It is expected that the GEE estimates are smaller because GLMMs model the data at the subject or individual level. The correlation between the measurements on the individual is generated by the random effect. Thus the βs for the GLMM represent the effect on an individual. A GEE models the data at the population level. The βs for a GEE represent the effect of the predictors averaged across all individuals with the same predictor values. GEEs do not use random effects but model the correlation at the marginal or correlation level. This is a major distinction.

The testing for vision is not entirely satisfactory since it has three levels meaning two tests—one being highly significant and the other not at all. If we want a single test for the significance of vision, we need to refit the model without vision and make the standard anova-type comparison:

```
modgeep2 = geeglm(stable ~ Sex + Age + Height + Weight + Surface,
  id =Subject, corstr="exchangeable", scale.fix=TRUE, data=ctsib,
  family=binomial)
anova(modgeep2, modgeep)
```

```
## Analysis of 'Wald statistic' Table
##
## Model 1 stable ~ Sex + Age + Height + Weight + Surface + Vision
## Model 2 stable ~ Sex + Age + Height + Weight + Surface
##   Df   X2 P(>|Chi|)
## 1  2 58.4   2.1e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

As expected, we see that vision is strongly significant.

The `geepack` package provides the flexibility of modeling an ordinal response with clusters using the `ordgee()` function. This would be appropriate for the original form of this data where the response is actually measured on a four-point scale. Recall that we dichotomized stable to be 1 if completely stable, and 0 otherwise.

We will now model the epilepsy data using GEEs. We exclude the 49th case as before (all the same caveats apply). An autoregressive AR(1) model for the correlation structure seems to be the most natural since consecutive measurements will be more correlated than measurements separated in time. Note that this does require that the clusters be sorted in time order (they are in this case).

```
modgeep = geeglm(seizures ~offset(log(timeadj)) + expind + treat +
  I(expind*treat), id=id, family=poisson, corstr="ar1", data=epilepsy,
  subset=(id!=49))
summary(modgeep)
```

```
##
## Call:
## geeglm(formula = seizures ~ offset(log(timeadj)) + expind + treat +
##     I(expind * treat), family = poisson, data = epilepsy, subset = (id !=
##     49), id = id, corstr = "ar1")
##
```

```
##  Coefficients:
##                   Estimate Std.err  Wald Pr(>|W|)
## (Intercept)         1.3138  0.1616 66.10  4.4e-16 ***
## expind              0.1509  0.1108  1.86    0.173
## treat              -0.0797  0.1983  0.16    0.688
## I(expind * treat)  -0.3987  0.1745  5.22    0.022 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation structure = ar1
## Estimated Scale Parameters:
##
##             Estimate Std.err
## (Intercept)     10.6    2.35
##   Link = identity
##
## Estimated Correlation Parameters:
##       Estimate Std.err
## alpha    0.783  0.0519
## Number of clusters:   58  Maximum cluster size: 5
```

The drug effects, as measured by the interaction term, has a weakly significant effect. The dispersion parameter is estimated as 10.6. This means that if we did not account for the overdispersion, the standard errors would be much larger. The AR(1) correlation structure can be seen in the working correlation where adjacent measurements have 0.78 correlation.

Computing details for GEEs

Estimating β We now discuss some computing details of iterative algorithms used to obtain $(\hat{\beta}, \hat{\alpha}, \hat{\phi})$ in Liang and Zeger [1986]. In order to estimate $\hat{\beta}$ we alternate between a modified Fisher scoring algorithm for β with $(\hat{\alpha}, \hat{\phi})$ fixed, and moment based estimation methods for α and ϕ. Given current estimates of the nuisance parameters $\hat{\alpha}$ and $\hat{\phi}$ at iteration j, we obtain $\hat{\beta}_{j+1}$ as

$$\hat{\beta}_{j+1} = \hat{\beta}_j - \left[\sum_{i=1}^{N} D_i^T(\hat{\beta}_j)\widehat{V}_i(\hat{\beta}_j)D_i(\hat{\beta}_j)\right]^{-1} \left[\sum_{i=1}^{N} D_i(\hat{\beta}_j)\widehat{V}_i(\hat{\beta}_j)S_i(\hat{\beta}_j)\right], \tag{5}$$

where

$$\widehat{V}_i(\beta) = V_i(\beta, \hat{\alpha}(\beta, \hat{\phi}(\beta))),$$
$$S_i(\beta) = Y_i - \mu_i(\beta).$$

This procedure can be viewed as a modification of Fisher's scoring method in that the limiting value of the expectation of the derivative of $\sum U_i(\beta, \hat{\alpha}(\beta, \hat{\phi}(\beta)))$ is used for correction. Now, define $D = (D_1^T, \ldots, D_N^T)^T$, $S = (S_1^T, \ldots, S_N^T)$ and let $\widetilde{V}$ be a block diagonal matrix with $\widehat{V}_i$s as the diagonal elements. Define modified responses as

$$Z = D\beta - S,$$

then we see that the iterative procedure in (5) is equivalent to an IRLS of Z on D with weights $\widehat{V}_i^{-1}$.

Estimating α and ϕ At a given iteration j the correlation parameters α and scale parameter ϕ can be estimated from the current Pearson residuals defined by

$$r_{it} = \frac{y_{it} - \mu_{it}(\hat{\beta}_j)}{\sqrt{a(\mu_{it}(\hat{\beta}_j))}}$$

where $i = 1, \ldots, N$ and $t = 1, \ldots, n_i$. We can estimate

$$\hat{\phi}^{-1} = \frac{\sum_{i=1}^{N} \sum_{t=1}^{n_i} \hat{r}_{it}^2}{K - p}$$

where $K = \sum n_i$. To estimate α consistently we borrow strength over the N subjects. The specific estimator depends on the choice of $R(\alpha)$. The general approach is to estimate α through a simple function of

$$\widehat{R}_{uv} = \frac{\sum_{i=1} \hat{r}_{iu} \hat{r}_{iv}}{K - p}.$$

For example, let $\alpha_i = (\alpha_{i1}, \ldots, \alpha_{in_i - 1})$ where $\alpha_{it} = \text{corr}(Y_{it}, Y_{it+1})$. A natural estimator for α_i, given β and ϕ, is

$$\hat{\alpha_t} = \frac{\phi \sum_{i=1}^{N} \hat{r}_{it} \hat{r}_{it+1}}{N - p},$$

where this estimator is better utilized in the setting where $n_i = n$. Note that the asymptotic distribution of $\hat{\beta}$ does not depend on the specific choices of estimators for α and ϕ provided that these choices are $\sqrt{N}$ consistent. Remember that estimators $\hat{\beta}$ and $\widehat{V}$ will be consistent no matter the choice of $R(\alpha)$. However, choosing $R(\alpha)$ closer to the true correlation gives increased efficiency.

Acknowledgments

These notes are from Faraway [2016]. We also borrow materials from Pinheiro and Bates [2006], Agresti [2013], Knudson [2018], Knudson et al. [2021], Halekoh et al. [2006], and Liang and Zeger [1986].

References

Alan Agresti. *Categorical Data Analysis.* John Wiley & Sons, 2013.

Julian J Faraway. *Extending the linear model with R: generalized linear, mixed effects and nonparametric regression models.* CRC press, 2016.

Charles J Geyer. On the convergence of monte carlo maximum likelihood calculations. *Journal of the Royal Statistical Society: Series B (Methodological)*, 56(1):261–274, 1994.

Charles J Geyer and Elizabeth A Thompson. Constrained monte carlo maximum likelihood for dependent data. *Journal of the Royal Statistical Society: Series B (Methodological)*, 54(3):657–683, 1992.

Ulrich Halekoh, Søren Højsgaard, Jun Yan, et al. The r package geepack for generalized estimating equations. *Journal of statistical software*, 15(2):1–11, 2006.

Christina Knudson. glmm: generalized linear mixed models via monte carlo likelihood approximation. *R package version*, 1(3), 2018.

Christina Knudson, Sydney Benson, Charles Geyer, and Galin Jones. Likelihood-based inference for generalized linear mixed models: Inference with the r package glmm. *Stat*, 10(1):e339, 2021.

Kung-Yee Liang and Scott L Zeger. Longitudinal data analysis using generalized linear models. *Biometrika*, 73(1):13–22, 1986.

Wei Pan. Akaike's information criterion in generalized estimating equations. *Biometrics*, 57(1):120–125, 2001.

José Pinheiro and Douglas Bates. *Mixed-effects models in S and S-PLUS.* Springer Science & Business Media, 2006.