Chapter 13

Mixed Effect Models for Nonnormal Responses

13.1 Generalized Linear Mixed Models

Generalized linear mixed models (GLMM) combine the ideas of generalized linear models with the random effects modeling ideas of the previous two chapters. The response is a random variable, Y_i, taking observed values, y_i, for $i = 1,\ldots,n$, and follows an exponential family distribution as defined in Chapter 8:

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

Let $EY_i = \mu_i$ and let this be connected to the linear predictor η_i using the link function g by $\eta_i = g(\mu_i)$. Suppose for simplicity that we use the canonical link for g so that we may make the direct connection that $\theta_i = \mu_i$.

Now let the random effects, γ, have distribution $h(\gamma|V)$ for parameters V. The fixed effects are β. Conditional on the random effects, γ,

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

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

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

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

13.2 Inference

A variety of approaches are available for estimating and performing inference for these models. All have strengths and weaknesses so it is not possible to recommend a single method to use in all circumstances. We present an overview of the theory behind these approaches before demonstrating the implementation on two examples. Later in the chapter, we discuss a related method called generalized estimating equations (GEE).

Penalized Quasi-Likelihood (PQL): In Section 8.2, we described a method by

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 and we have the flexibility to modify the models to allow for nonstandard features. 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. Finally, the inferential conclusions are of a different form. This is either an advantage or disadvantage depending on your point of view. See Chapter 12 for an introduction to Bayes methods for LMMs. Extending these ideas to GLMMs is not difficult.

We now apply these methods to two examples. The first has a Bernoulli response and the second a Poisson response.

13.3 Binary Response

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. The data comes from Steele (1998) via the Australasian Data and Story Library (OzDASL).

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). We start by defining this response:

```
data(ctsib, package="faraway")
ctsib$stable <- ifelse(ctsib$CTSIB==1,1,0)
```

We can investigate the effects of the treatment variables on stability descriptively. Here is the mean response for the combined conditions:

```
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 have divided by 80 because `xtabs` sums the values for each combination and there are 40 subjects with each combination replicated twice. 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 in Figure 13.1.

```
library(dplyr)
subsum <- ctsib %>% group_by(Subject) %>% summarise(Height=Height[1],
    ↪  Weight=Weight[1], stable=mean(stable), Age=Age[1], Sex=Sex[1])
library(ggplot2)
ggplot(subsum, aes(x=Height,y=stable))+geom_point()
ggplot(subsum, aes(x=Weight,y=stable))+geom_point()
ggplot(subsum, aes(x=Age,y=stable))+geom_point()
```

```
ggplot(subsum, aes(x=Sex,y=stable))+geom_boxplot()
```

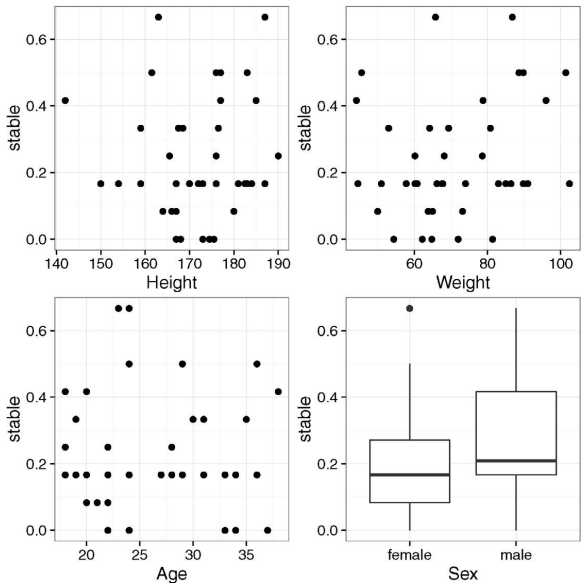

Figure 13.1 *Subject effects for the stability experiment. Response is proportion of stable over treatment conditions.*

We could fit a binomial GLM ignoring the subject information entirely:

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

```
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  7.27745    3.80399    1.91  0.05573
Sexmale      1.40158    0.51623    2.72  0.00663
Age          0.00252    0.02431    0.10  0.91739
Height      -0.09641    0.02684   -3.59  0.00033
Weight       0.04350    0.01800    2.42  0.01567
Surfacenorm  3.96752    0.44718    8.87  < 2e-16
Visiondome   0.36375    0.38322    0.95  0.34252
Visionopen   3.18750    0.41600    7.66  1.8e-14

n = 480 p = 8
Deviance = 295.203 Null Deviance = 526.254 (Difference = 231.051)
```

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. We could also try including a fixed subject factor:


```
gfs <- glm(stable ~ Sex + Age + Height + Weight + Surface + Vision +
    ↪ factor(Subject), binomial,data=ctsib)
```

However, when we examine the summary for this model, we see problems with identifiability. This is because the subject factors cannot be completely distinguished from the four subject-specific measures: sex, age, height and weight. Even if we could get around this problem, it would hardly be appropriate to treat the subject factor as a fixed effect. We do not care about the individual subjects but we are interested in how the four subject measures might affect stability. The experimental subjects are intended as a random sample from the target population. We'd like to know something about the inherent variability in that population that is not explained by measurable variables but we don't care about the specific individuals.

There are a variety of ways of fitting GLMMs in R. First we demonstrate the PQL method implemented in the `MASS` package:

```
library(MASS)
modpql <- glmmPQL(stable ~ Sex + Age + Height + Weight + Surface +
    ↪ Vision, random=~1|Subject, family=binomial,data=ctsib)
summary(modpql)
```

```
Random effects:
 Formula: ~1 | Subject
        (Intercept) Residual
StdDev:      3.0607  0.59062

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) 15.5715   13.4983 437  1.1536  0.2493
Sexmale      3.3553    1.7526  35  1.9145  0.0638
Age         -0.0066    0.0820  35 -0.0810  0.9359
Height      -0.1908    0.0920  35 -2.0736  0.0455
Weight       0.0695    0.0629  35  1.1052  0.2766
Surfacenorm  7.7241    0.5736 437 13.4665  0.0000
Visiondome   0.7265    0.3259 437  2.2289  0.0263
Visionopen   6.4853    0.5440 437 11.9219  0.0000
```

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. The residual SD is an artefact of the fitting process and does not exist in the statement of the model.

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, so these results cannot be relied upon. Furthermore, the Bernoulli response may lead to biased estimates of regression coefficients. Hence, it would be unwise to rely entirely on this analysis without investigating alternative methods of estimation.

The numerical integration-based methods are implemented in the `lme4` package. The default choice of method is the Laplace approximation.

```
library(lme4)
```

```
modlap <- glmer(stable ~ Sex + Age + Height + Weight + Surface +
    ↪ Vision + (1|Subject), family=binomial, data=ctsib)
```

Since the Laplace method is a special case of the Gauss-Hermite approximation which can only be more accurate, it is best to attempt this approach. Here we can use the maximum allowable number of quadrature points which is 25:

```
modgh <- glmer(stable ~ Sex + Age + Height + Weight + Surface +
    ↪ Vision + (1|Subject), nAGQ=25, family=binomial, data=ctsib)
```

We have a particularly simple random effects structure so we can easily afford to be profligate in the number of quadrature points (which is certainly more than we need). In more complex examples, we may need to specify much smaller numbers to allow computation in a reasonable time. Start small and increase until the estimates stop changing very much or the computation becomes infeasibly long. Now look at the output:

```
summary(modgh)
```

```
     AIC      BIC   logLik deviance df.resid
   247.9    285.5   -115.0    229.9      471

Scaled residuals:
   Min     1Q Median     3Q    Max
-4.884 -0.139 -0.020 -0.001  4.902

Random effects:
 Groups  Name        Variance Std.Dev.
 Subject (Intercept) 7.19     2.68
Number of obs: 480, groups:  Subject, 40

Fixed effects:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) 16.17166   12.72107    1.27    0.204
Sexmale      3.09679    1.69612    1.83    0.068
Age         -0.00668    0.07646   -0.09    0.930
Height      -0.19226    0.08895   -2.16    0.031
Weight       0.07515    0.05910    1.27    0.204
Surfacenorm  7.28541    1.05516    6.90  5.0e-12
Visiondome   0.67591    0.52737    1.28    0.200
Visionopen   6.08896    0.97241    6.26  3.8e-10
```

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)
```

```
       Df AIC BIC logLik deviance Chisq Chi Df Pr(>Chisq)
modgh2  5 247 268   -119      237
modgh   9 248 286   -115      230  7.37      4       0.12
```

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 scepticism. 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.

As with all such models, it is wise to check some diagnostics. These can be extracted using `residuals()` and `fitted()` functions. An alternative convenience is:

```
dd <- fortify(modgh2)
```

which extracts the residuals and fitted values and places them in a common data frame with the other variables. This makes the construction of some plots more convenient. For example, we might look at the QQ plots subsetted by the treatment variables:

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

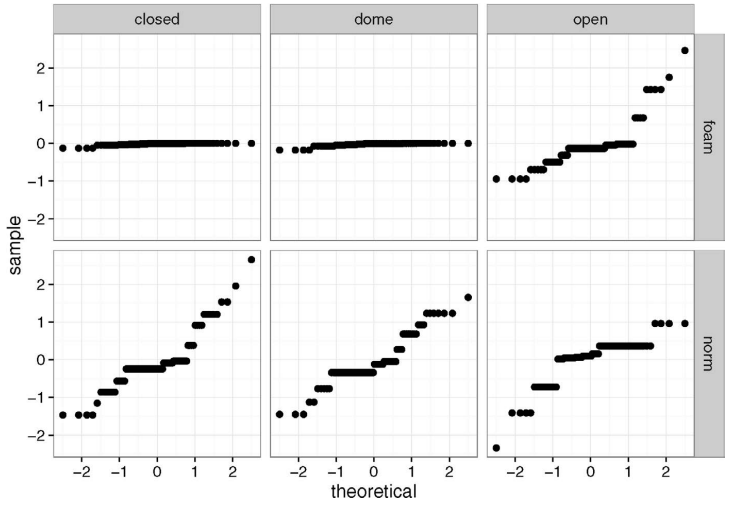

Figure 13.2 *QQ plots subsetted by treatment variables.*

In Figure 13.2, 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.

We can use INLA for a Bayesian approach to fitting these models. See Section 12.2 for an introduction. For ease of exposition, we use only the surface and vision as fixed effect predictors. The default, noninformative priors, are satisfactory:

```
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:

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

The posterior density for this SD is shown in the first panel of Figure 13.3:

```
x <- seq(0,7,length.out = 100)
sdf <- data.frame(yield = x, density=inla.dmarginal(x, sigmaalpha))
ggplot(sdf,aes(x=yield,y=density))+geom_line()
```

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:

```
restab <- sapply(result$marginals.fixed, function(x) inla.zmarginal(x,
    ↪ silent=TRUE))
restab <- cbind(restab, inla.zmarginal(sigmaalpha,silent=TRUE))
colnames(restab) = c("mu","norm","dome","open","alpha")
data.frame(restab)
```

```
                mu    norm     dome   open   alpha
mean       -10.298  7.3641  0.66618 6.1279  3.0248
sd          1.3507 0.92526  0.49873 0.8498 0.62416
quant0.025 -13.172  5.7182 -0.29877 4.6184  1.9838
quant0.25  -11.181  6.6971  0.32503  5.517  2.5756
quant0.5    -10.21  7.3038  0.65785 6.0704  2.9585
quant0.75   -9.334  7.9657  0.99581 6.6771  3.4007
quant0.975 -7.9392  9.3029   1.6579 7.9167   4.429
```

We could compute similar statistics on the subject random effects but there are too many to display them all. 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 in the second panel of Figure 13.3:

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

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:

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

```
[1] 0.17982
```

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. 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.

We can also use STAN for a Bayesian analysis as introduced in Section 12.1. Here is the STAN program we need:

```
data {
  int<lower=0> Nobs;
  int<lower=0> Nsubs;
  int<lower=0> Npreds;
  int<lower=0,upper=1> y[Nobs];
  int<lower=1,upper=Nsubs> subject[Nobs];
```


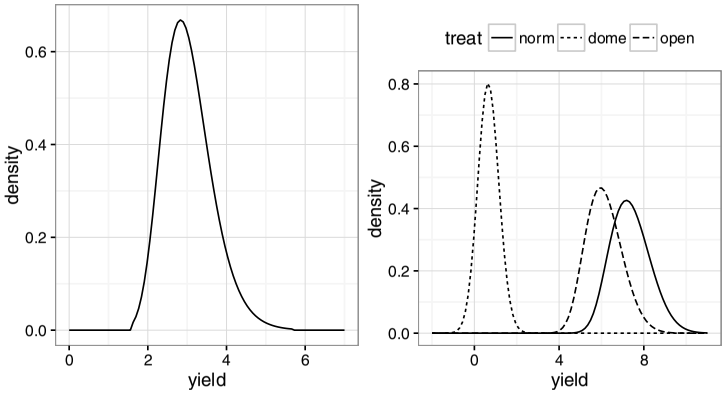

Figure 13.3 *Posterior density for the subject SD on the left and posterior densities for the treatment effects on the right.*

```
  matrix[Nobs,Npreds] x;
}
parameters {
  vector[Nsubs] subeff;
  real<lower=0> sigmasubj;
  vector[Npreds] beta;
}
model {
  subeff ~ normal(0,sigmasubj);
  sigmasubj ~ cauchy(0, 1);
  for(n in 1:Nobs) {
    y[n] ~ bernoulli_logit(x[n]*beta + subeff[subject[n]] );
  }
}
```

We have written this in a generic form so that you could use this for any grouped-by-subject data with a binary response. We use a half-Cauchy prior for the subject SD. This is somewhat more informative but seems justifiable in the context of this data. It also has the advantage of being more transparent.

We need to prepare the data in a format compatible with the `data` block in the code above. We form the model matrix of fixed effects, X, in advance:

```
xm <- model.matrix(~ Sex + Age + Height + Weight + Surface + Vision,
    ↪ ctsib)
stabledat <- with(ctsib, list(Nobs=nrow(ctsib),
  Nsubs=length(unique(ctsib$Subject)), Npreds=ncol(xm),
  y=stable, subject=Subject, x=xm))
```

We can now run the STAN model. We have broken the process into three steps. The first step translates the STAN code into C++, the second compiles that C++ code and the third runs the MCMC sampler. The advantage of doing it in three stages is that

one is likely only to do the first two once but the third might need to be repeated if the model or data is changed.

```
library(rstan)
rt <- stanc("glmmbin.stan")
sm <- stan_model(stanc_ret = rt, verbose=FALSE)
fit <- sampling(sm, data=stabledat)
```

This will take several minutes to run depending on the quality of your computer.

First we need to check the diagnostics of the MCMC sampling. We plot the chain for the subject SD as this is the parameter most likely to cause problems:

```
traceplot(fit,pars="sigmasubj", inc_warmup=FALSE)
```

The plot (not shown) is entirely satisfactory. We can display a summary for the parameters of interest:

```
print(fit,pars=c("sigmasubj","beta"))
```

```
          mean se_mean    sd   2.5%   25%   50%   75% 97.5% n_eff Rhat
sigmasubj 3.60    0.04  0.81   2.28  3.03  3.48  4.09  5.36   515 1.01
beta[1]  19.25    0.73 17.56 -13.12  7.66 18.52 30.11 55.08   585 1.01
beta[2]   3.83    0.10  2.24  -0.45  2.35  3.76  5.27  8.33   531 1.01
beta[3]  -0.01    0.00  0.11  -0.23 -0.08 -0.01  0.06  0.20   836 1.00
beta[4]  -0.23    0.00  0.12  -0.48 -0.30 -0.22 -0.15 -0.01   586 1.01
beta[5]   0.09    0.00  0.08  -0.06  0.03  0.08  0.14  0.24   860 1.00
beta[6]   8.56    0.06  1.33   6.28  7.58  8.43  9.39 11.33   560 1.01
beta[7]   0.75    0.01  0.56  -0.34  0.38  0.74  1.14  1.87  4000 1.00
beta[8]   7.24    0.05  1.24   5.17  6.34  7.13  8.01  9.92   563 1.01
```

The effective sample sizes are more than satisfactory.

Now we examine the posterior distributions. We extract the parameters of interest and restore the variable names for convenience. The `reshape2` package helps us arrange the data in a format for convenient plotting. We show the estimated densities in Figure 13.4 along with a vertical line at zero.

```
ipars <- data.frame(extract(fit, pars=c("sigmasubj","beta")))
colnames(ipars)[-1] <- colnames(xm)
library(reshape2)
rdf <- melt(ipars)
ggplot(rdf, aes(x=value))+geom_density() + facet_wrap(~ variable,
    ↪ scales="free")+geom_vline(xintercept=0)
```

We might also be interested in how the subjects in the experiment compare. We extract the subject random effects and sort the posterior means:

```
ppars <- data.frame(extract(fit, pars="subeff"))
sort(colMeans(ppars))
```

```
 subeff.3 subeff.38 subeff.37 subeff.14
-6.704126 -4.926872 -4.563769 -4.036449 ...
...edited...
subeff.17 subeff.29 subeff.25 subeff.27
 3.488328  5.906336  6.735636  6.924570
```

We see that subject 3 is the least stable and subject 27 is the most stable. Since we have access to the posterior distributions, we can readily investigate which difference might be notable.

13.4 Count Response

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

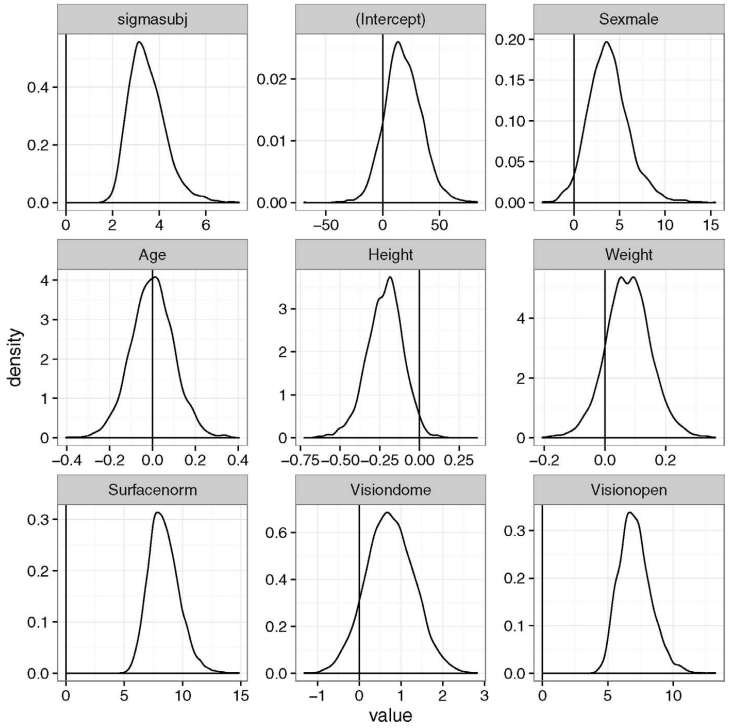

Figure 13.4 *Posterior distributions as produced by the STAN fit to the epilepsy data.*

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. The data have been analyzed by many authors including Thall and Vail (1990), Breslow and Clayton (1993) and Diggle et al. (2013). Does Progabide reduce the rate of seizures?

First we create some derived variables 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 were not treated (`treat=0`). The `expind` indicates the baseline phase by 0 and the treatment phase by 1. The length of these time phases is recorded in `timeadj`. We have created three new convenience variables: `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. The `dplyr` package is useful for these types of group summaries:

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

```
             drug
phase         placebo treatment
  baseline     3.8482    3.9556
  experiment  4.3036    3.9839
```

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 also 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")
```

We compare the two groups in the left panel of Figure 13.5 and find little to choose between them. The square-root transform is used to stabilize the variance; this is often used with count data. Now we compare the average seizure rate to the baseline for the two groups:

```
ratesum <- epilepsy %>%
 group_by(id, phase, drug) %>%
 summarise(rate=mean(seizures/timeadj))
library(tidyr)
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")
```

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)
```


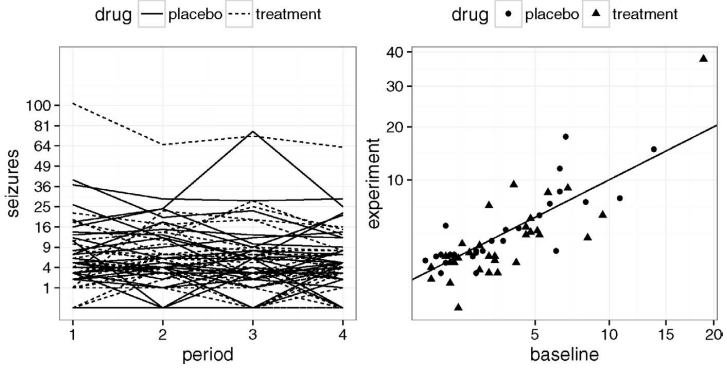

Figure 13.5 *Seizures per 2-week period on a square-root scale with treatment group shown as solid lines and the placebo group shown as dotted lines in the plot on the left. Mean seizures per week is shown on the right. We compare the baseline period with the experimental period, distinguishing those who receive treatment or control.*

Excluding a case should not be taken lightly. It is worth repeating the analysis with and without this subject. For projects where the analyst works with producers of the data, it will be possible to discuss substantive reasons for excluding cases. Exclusion of cases should always be reported and not concealed.

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* as explained in Section 5.3 to allow for the difference in lengths in the baseline and treatment periods.

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

```
                Estimate Std. Error z value Pr(>|z|)
(Intercept)       1.3476     0.0341   39.57  < 2e-16
expind            0.1118     0.0469    2.39    0.017
treat            -0.1068     0.0486   -2.20    0.028
I(expind * treat) -0.3024     0.0697   -4.34 0.000014

n = 290 p = 4
Deviance = 2411.550 Null Deviance = 2485.110 (Difference = 73.560)
```

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 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 as discussed in Section 9.4. However, this is a different consideration to the correlated response.

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

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

```
 Formula: ~1 | id
        (Intercept) Residual
StdDev:     0.68197   1.6054

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)         1.08079  0.143701 230  7.5211  0.0000
expind              0.11184  0.075767 230  1.4761  0.1413
treat              -0.00894  0.200244  56 -0.0446  0.9646
I(expind * treat) -0.30238  0.112689 230 -2.6834  0.0078
```

The parameter estimates are comparable to the GLM but the standard errors are larger as might be expected given that the correlated response has 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. A further concern is the problematic meaning of the residual SD as such a term does not appear in the statement of the model. Also we lack an AIC due to the quasi-ness of the likelihood. Even so, we do see a significant negative interaction effect indicating that the drug is effective.

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.

```
library(lme4)
modgh <- glmer(seizures ~offset(log(timeadj)) + expind + treat + I(
    ↪ expind*treat)+ (1|id), nAGQ=25, family=poisson, data=epilo)
summary(modgh)
```

```
Random effects:
 Groups Name        Variance Std.Dev.
 id     (Intercept) 0.515    0.718
Number of obs: 290, groups:  id, 58

Fixed effects:
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)        1.03600    0.14126    7.33  2.2e-13
expind             0.11184    0.04688    2.39    0.017
treat             -0.00815    0.19652   -0.04    0.967
I(expind * treat) -0.30239    0.06971   -4.34  1.4e-05
```

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] 0.73934
```

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.

We can also take a Bayesian approach, starting with `STAN`. We can use almost the same code as the binary response example except we need to add

```
  vector[Nobs] offset;
```

in the data block and replace the model line with

```
    y[n] ~ poisson_log(log(offset[n])+x[n]*beta + subeff[subject[n]] );
```

We prepare the data into the required format using:

```
epilo$id[epilo$id == 59] <- 49
xm <- model.matrix( ~ expind + treat + I(expind*treat), epilo)
epildat <- with(epilo,list(Nobs=nrow(epilo), Nsubs=length(unique(id)),
                           Npreds=ncol(xm),
                           y=seizures,
                           subject=id,
                           x=xm, offset=timeadj))
```

We've renumbered case 59 into the previously deleted case 49 slot. This is ugly but we need the subjects to be consecutively numbered.

Assuming that the code is placed in a file called `glmmpois.stan`, we translate, compile and run the sampler:

```
library(rstan)
rt <- stanc("glmmpois.stan")
sm <- stan_model(stanc_ret = rt, verbose=FALSE)
fit <- sampling(sm, data=epildat)
```

We can check the sampling properties of the chain by

```
traceplot(fit,pars="sigmasubj", inc_warmup=FALSE)
```

We've made the plot only for subject SD since this is the one most likely to cause problems. In this case, the plot (not shown) is satisfactory. We can review the posterior distributions:

```
ipars <- data.frame(rstan::extract(fit, pars=c("sigmasubj","beta")))
colnames(ipars) <- c("subject","intercept","expind","treat","
    ↪ interaction")
```

We plot the two most interesting posterior distributions as seen in Figure 13.6:

```
ggplot(ipars, aes(x=subject))+geom_density()
ggplot(ipars, aes(x=interaction))+geom_density() +geom_vline(
    ↪ xintercept=0)
```

We can see the subject SD is very clearly different from zero and that it is centered on about 0.7. The interaction effect (or drug effect) is negative centered on about -0.3. This looks clearly less than zero.

We can construct a convenient summary of the results including a sort of p-value.

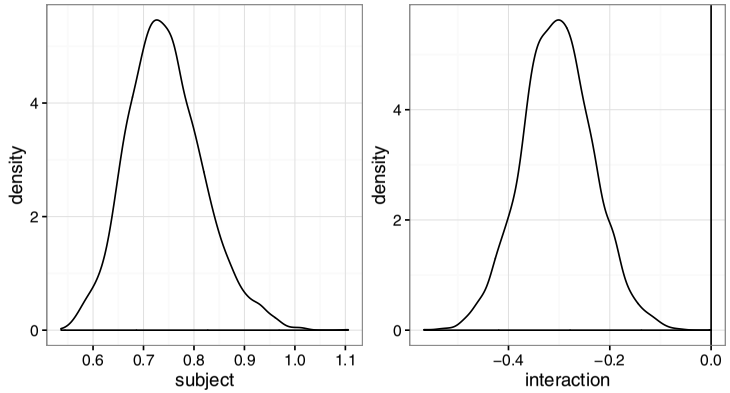

Figure 13.6 *Posterior distribution of the subject SD and drug effect for the epilepsy data.*

```
bayespval <- function(x) {p <- mean(x > 0); 2*min(p,1-p)}
smat <-  apply(ipars, 2, function(x) c(mean(x), quantile(x,c(0.025,
    ↪ 0.975)), bayespval(x)))
row.names(smat) <- c("mean","LCB","UCB","pvalue")
t(smat)
```

```
                mean       LCB      UCB pvalue
subject     0.743504  0.613355  0.90935 0.0000
intercept   1.021433  0.736583  1.30687 0.0000
expind      0.110912  0.017704  0.20310 0.0210
treat       0.005876 -0.386557  0.39178 0.9695
interaction -0.301699 -0.434090 -0.17111 0.0000
```

We see that the posterior means reported here are quite similar to the estimates computed earlier using likelihood methods. The Bayesian analysis seems rather more confident that there is a drug effect since this posterior distribution is well separated from zero.

The same model can also be fit by INLA in a straightforward way:

```
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        1.036  0.11178 -0.0081489    -0.30246  0.72583
sd        0.14196 0.046891    0.19753     0.06973 0.070948
```