Data Analysis of GLMM and Overdispersion

Moore's Teratology Data

This data example is from Agresti (2002). Female rats were put on iron-deficient diets and then randomized to receive placebo (group 1) and three different regents of iron supplements (groups 2, 3, and 4). They are sacrificed 3 weeks after pregnant.

n is the total number of fetuses in a litter and y is the number of dead fetuses.

```
setwd('d:/course/SKKU/Longitudinal_Data_Analysis/R-codes')

tera <- read.table ("tera.dat",col.names = c("litter", "group", "n", "y"))
tera$group <- factor (tera$group)
tera[1:5,]

tera.glm<- glm(cbind(y,n-y)~group,family=binomial,data=tera)
summary (tera.glm)

plot(tera$n,resid(tera.glm,type="pearson"),xlab="Litter Size",ylab = "Pearsons Resid
     main = "Moores Teratology Data")

pearson.residuals <- resid(tera.glm,"pearson")
phi <- sum(pearson.residuals^2)/tera.glm$df.residual
rho <- 0.188

## Calculate sandwich variance estimates
X <- model.matrix (tera.glm)
mu <- fitted (tera.glm)
V.mu <- tera$n * mu * (1 - mu)
```

```
## Matrix A^-1
Ainv <- solve (t(X) %*% diag (V.mu) %*% X)
## Model Based Standard Error Estiamtes
se.model <- sqrt (diag (Ainv))
## Sandwich Estimates with Scale Variance
se.scaled <- sqrt (phi) * se.model
## Sandwich Estimates With Beta-Binomial Variance
B <- t(X) %*% diag (V.mu * (1 + rho * (tera$n - 1))) %*% X
se.sandwich <- sqrt (diag (Ainv %*% B %*% Ainv))
## Empirical Sandwich Estimates
est.fnx <- X * (tera$y - tera$n * mu)
UUt <- t(est.fnx) %*% est.fnx
se.empirical <- sqrt (diag (Ainv %*% UUt %*% Ainv))
round (cbind (coef (tera.glm),se.model,se.scaled,se.sandwich,se.empirical),4)

## Quasi-likelihood Model 1
tera.quasi <- glm(cbind(y,n-y)~group,data=tera,family=quasibinomial)
summary (tera.quasi)

## GEE and Empirical Standard Errors
library(gee)
tera.gee <- gee(cbind(y,n-y)~group,data=tera,id = litter,family=binomial)
summary (tera.gee)

## Quasi-likelihood Model 2
#source ("bod_regn.R")
#x <- cbind(1,tera$g2,tera$g3,tera$g4)
#tera.bod <- bod.regn(y=tera$y,n=tera$n,x=x,dispersion="correlation",beta=c(1,-3,-4
#summary(tera.bod)

## Beta-Binomial Model: Maximum likelihood
library (rmutil) # You can download the library at http://www.commanster.eu/rcode.ht
library (gnlm)  # for 'gnlr' function
tera$g2 <- as.numeric(tera$group == 2)
tera$g3 <- as.numeric(tera$group == 3)
tera$g4 <- as.numeric(tera$group == 4)
library (boot)  # for 'inv.logit' function
```

```
attach (tera)
tera.gnlr2 <- gnlr(cbind(y,n - y),distribution="beta binomial",mu=function(beta){
 inv.logit(beta[1]+beta[2]*g2+beta[3]*g3+beta[4]*g4)
 },pmu=c(1, -3, -4, -4),pshape = 2)
tera.gnlr2

tera.gnlr <- gnlr(cbind(y,n-y),distribution="beta binomial",mu=function(beta){
 inv.logit (beta[1]+beta[2]*g2+beta[3]*g3+beta[4]*g4)
 },pmu = c(1, -3, -4, -4),shape=finterp(~ n),pshape=c(2,0.5))
tera.gnlr
```

Conditional Logistic Regression

The clogit function in package survival can fit conditional logistic regression models.

```
setwd('d:/course/SKKU/Longitudinal_Data_Analysis/2016Fall/R-codes')

# Conditional logistic regression
library(survival)
xover3 <- read.table("xover3.data",col.names=c("id","class","relief",
           "intercept","tx2","tx3","p2","p3","ptx1","ptx2","ptx3"))

xover3.cl <- clogit(relief~tx2+tx3+p2+p3+ptx2+ptx3+strata(id),data=xover3)
summary(xover3.cl)
```

2×2 Crossover Trial

```
# 2*2 Crossover trial

xover <- read.table("xover1.data",col.names=c("id","class","y",
 "intercept","trt","period","xover","BA"))
with(xover,ftable(BA,trt,y))
```

```
library(survival)
library(glmmML)
xover.cl<-clogit(y~trt+strata(id),data=xover)
xover.gee <-geese(y~trt,data=xover,corstr="exchangeable",id=id,family=binomial,scal

xover.glmm <-glmmML(y~trt,data=xover,cluster=id,family=binomial)


# 2*3 Crossover trial

library(MASS)
xover3 <- read.table("xover3.data",col.names=c("id","class","relief",
 "intercept","tx2","tx3","p2","p3","ptx1","ptx2","ptx3"))

xover3.glmm <-glmmPQL(fixed=relief~tx2+tx3+p2+p3+ptx2+ptx3,
                     random = ~ 1 | id,family = "binomial",data = xover3)
summary(xover3.glmm)

xover3.glmmML<- glmmML(relief~tx2+tx3+p2+p3+ptx2+ptx3,
                       cluster=xover3$id,family=binomial,
                       data=xover3,n.points = 20)
summary(xover3.glmmML)
```

Indonesian Children Health Study

```
# Indonesian Children Health Study

ICHS <- read.table("ICHS.dat", header = TRUE)
ichs.glmm <- glmmPQL(RESPONSE~VITA+AGE+I(AGE^2)+GENDER+TIME,
                     random= ~ TIME | ID,
                     data=ICHS,family="binomial")
summary(ichs.glmm)

truehist(ranef(ichs.glmm)[,1], xlab = "Intercept")
truehist(ranef(ichs.glmm)[,2], xlab = "Time")

# For GLMM, different algorithms and software may
# give quite different answers

# random intercept and slope
ichs.glmmPQL<-glmmPQL(RESPONSE~VITA+AGE+I(AGE^2)+GENDER+TIME,
                      random = ~ 1 | ID,data=ICHS,family="binomial")
summary(ichs.glmmPQL)


library(repeated) # download from my directory
ichs.glmm2 <- glmm(RESPONSE~VITA+AGE+I(AGE^2)+GENDER+TIME,nest=ID,
                   data=ICHS,family="binomial",points = 20)
summary (ichs.glmm2)


ichs.glmmML <-glmmML(RESPONSE~VITA+AGE+I(AGE^2)+GENDER+TIME,
                     cluster=ICHS$ID,data=ICHS,family=binomial,n.points=20)
summary(ichs.glmmML)
```

PQL in SAS

```
options ls = 72;
data teaching.ichs;
  infile 'ICHS.dat' firstobs=2;
  input id response time gender vita age;
run ;

proc glimmix data=teaching.ichs method=RSPL;
  class id;
  model response (event='1')=vita age age*age
  gender time/dist=binary solutions;
  nloptions maxit=100;
  random intercept/subject=id;
run;
```

Note that it is possible to fit a GEE model with glimmix

```
proc glimmix data=teaching.ichs method=RSPL empirical;
  class id;
  model response (event='1') = vita age age*age gender time/dist=binary solutions;
  nloptions maxit = 100;
  random _residual- / subject=id type=cs;
run ;
```

Random intercept (Gaussian quadrature with 10 points):

```
proc nlmixed data = teaching.ichs noad qpoints=10;
  parms beta0=-1.5 beta1=0.3 beta2=0.5 beta3=0 beta4=-0.8 beta5=0 tau=3;
  mu=beta0+beta1*vita+beta2*age+beta3*age*age+beta4*gender+beta5*time+b1;
  p=exp(mu)/(1+exp(mu));
  model response~binary(p);
  random b1~normal(0,tau**2) subject=id;
run;
```

Random intercept and slope (adaptive Quadrature with the number of points chosen automatically).

```
proc nlmixed data = ichs;
  parms beta0=-1.5 beta1=0.3 beta2=0.5 beta3=0 beta4=-0.8 beta5=0
        s1=3 c1=-0.5 s2=0.2;
  mu=beta0+beta1*vita+beta2*age+beta3*age*age+beta4*gender+beta5*time+b1+b2*time;
  p=exp(mu)/(1=exp(mu));
  model response~binary(p);
  random b1 b2~normal ([0,0],[s1,c1,s2])
  subject=id;
run;
```

Table 1: Comparison of fitting a random intercept model with binary data.

	glmmPQL	glmmML	glmm	SAS GLIMMIX	SAS NLMIXED
Intercept	-1.88 (0.76)	-2.35 (0.86)	-2.33 (0.37)	-1.56 (0.58)	-2.75 (0.78)
Vita	0.40 (0.35)	0.63 (0.42)	0.60 (0.17)	0.33 (0.30)	0.92 (0.41)
Age	0.56 (0.40)	0.83 (0.51)	0.80 (0.21)	0.47 (0.35)	1.12 (0.47)
Age^2	-0.089 (0.052)	-0.13 (0.066)	-0.13 (0.027)	-0.075 (0.045)	-0.18 (0.065)
Gender	-0.90 (0.34)	-1.25 (0.41)	-1.21 (0.17)	-0.78 (0.29)	-1.42 (0.39)
Time	0.033 (0.011)	0.034 (0.016)	0.034 (0.016)	0.027 (0.014)	0.034 (0.016)
τ	2.33	2.86	2.84 (0.16)	3.48 (0.47)	2.80 (0.25)

Seizure Data

```
# Seizure Data
setwd('d:/course/SKKU/Longitudinal_Data_Analysis/2016Fall/R-codes')
library(gee)
library(geepack)
library(repeated)

data(seize)
seizure

seiz.l <- reshape(seizure,varying=list(c("base","y1","y2","y3","y4")),
                  v.names="y", times=0:4, direction="long")
seiz.l <- seiz.l[order(seiz.l$id, seiz.l$time),]
seiz.l$t <- ifelse (seiz.l$time == 0, 8, 2)
seiz.l$x <- ifelse (seiz.l$time == 0, 0, 1)

# GEE
geese(y~offset(log(t))+x+trt+x:trt,id=id,
      data=seiz.l, subset=id!= 49,
      corstr = "exch", family=poisson)

# Poisson-Gaussian model: random intercept
glmmPQL(y ~ x + trt + x:trt + offset (log (t)),
        random = ~ 1 | id, family = poisson,
        data = seiz.l[seiz.l$id != 49,])

# Poisson-Gaussian model: random intercept and random slope for post
glmmPQL(y ~ x + trt + x:trt + offset (log (t)),
        random = ~ x | id,family = poisson,
        data = seiz.l[seiz.l$id != 49,])
```

SAS

```
data seizure ;
  infile 'seizlong.dat' firstobs=2;
  input trt age time y id t x;
  logt=log(t);
  if id=49 then delete;
run;

proc glimmix data=seizure method=RSPL empirical;
  class id;
  model y = x|trt / dist=poi offset=logt solutions;
  nloptions maxit = 100;
  random intercept / subject = id;
run;

proc nlmixed data=seizure;
  parms b0 = 1 b1 = 0 b2 = 0 b3 = 0 sig1 = 0.1;
  lambda=b0+b1*x+b2*trt+b3*x*trt+u0+log(t);
  mu=exp(lambda);
  model y ~ Poisson(mu);
  random u0 ~ Normal(0,sig1) subject=id;
run;

proc nlmixed data=seizure;
  parms b0 = 1 b1 = 1 b2 = 0 b3 = 0 s11 = 0.1
        s12 = 0.05 s22 = 0.1;
  lambda=b0+b1*x+b2*trt+b3*x*trt+u0+u1*x+log(t);
  mu=exp(lambda);
  model y ~ Poisson(mu);
  random u0 u1 ~ Normal([0,0],[s11,s12,s22])
  subject = id;
run;
```

Comparison of Results

	GEE	GLMM: Intercept		
	R	glmmPQL	GLIMMIX	NLMIXED
Intercept	1.34 (0.16)	1.08 (0.14)	1.04 (0.15)	1.04 (0.14)
x	0.11 (0.12)	0.11 (0.076)	0.11 (0.11)	0.11 (0.047)
Trt	-0.11 (0.19)	-0.009 (0.2)	-0.007 (0.19)	-0.008 (0.19)
$X \times$Trt	-0.30 (0.17)	-0.30 (0.11)	-0.30 (0.17)	-0.30 (0.070)
D_{11}		0.68	0.52 (0.1)	0.51 (0.1)
ρ	0.60 (0.08)			
ϕ	10.4 (2.3)			

	GLMM: Intercept and x	
	glmmPQL	NLMIXED
Intercept	1.12 (0.13)	1.07 (0.13)
x	-0.003 (0.10)	0.008 (0.11)
Trt	-0.023 (0.18)	-0.008 (0.19)
$X \times$Trt	-0.30 (0.15)	-0.35 (0.15)
D_{11}	0.64	0.45 (0.09)
D_{22}	0.38	0.22 (0.06)
D_{12}	0.16	0.015 (0.05)

- The two functions (glmmML and glmm) in R implementing G-H quadrature methods seem to give erroneous answers.

- The point estimates from the GEE and GLMM random intercept models agree well with each other, as expected.

- From AIC, GLMM with random slope is a better model (644 vs 657).

- The PQL method is much faster than the quadrature method.

- PQL is also more robust to model misspecification.