# BIOSTAT 571 Homework 2

Wenxiao Gu

January 22, 2014

## Contents

# 1 Pr1

## 1.1 Part(a)

Since the mean-variance relationship of Bernoulli is fixed, then the theretical dispersion parameter would be 1. However, the estimated dispersion parameter from **glm()** function is 1.000836. The difference between the theoretical and the estimated values of dispersion parameter came from the calculation[1] in **glm()** function.

The dispersion of **glm()** is taken as 1 for the binomial and Poisson families, and otherwise estimated by the residual Chisquared statistic divided by the residual degrees of freedom. The relationship is stated as below,

$$\begin{aligned}
\hat{\phi}_{glm} &= \frac{1}{n-1}\hat{\chi}^2 = 1.000836 \neq 1 \\
\hat{\phi}_{theory} &= \frac{1}{n}\hat{\chi}^2 \\
\hat{\chi}^2 &= \sum_i \frac{(y_i - \hat{\mu}_i)^2}{\hat{\mu}_i(1-\hat{\mu}_i)}.
\end{aligned}$$

Therefore, in our example of data with sample size of $n = 1200 - 2 = 1198$, we have

$$\begin{aligned}
\hat{\phi}_{theory} &= \frac{n-1}{n}\hat{\phi}_{glm} \\
&= \frac{1197}{1198} \cdot 1.000836 \\
&= 1.
\end{aligned}$$

## 1.2 Part(b)

$$\begin{aligned}
Y_i | p_i &\sim binomial(m_i, p_i) \\
p_i &\sim beta(\alpha, \beta)
\end{aligned}$$

There are three ways in R to compute the log-likelihood, **dbetabinom()** function, **dbetabinom.ab()** function and computing by hand, as stated briefly below (and in the **Appendix**),

$$\begin{aligned}
P(Y_i = y) &= \mathbf{dbetabinom}\left[x = y, size = m_i, prob \equiv \mu_{bern.} = E(Y_i)/m_i = \frac{\alpha}{\alpha+\beta}, \rho = \frac{1}{\alpha+\beta+1}\right] \\
&= \mathbf{dbetabinom.ab}\left[x = y, size = m_i, shape1 = \alpha, shape2 = \beta\right] \\
&= \binom{m_i}{y} \cdot \frac{beta(\alpha+y, m_i+\beta-y)}{beta(\alpha,\beta)} \\
log-likelihood &\sim -\sum log\left\{\mathbf{dbetabinom}\left[\ldots\right]\right\} \\
&\sim -\sum log\binom{m_i}{y} - \sum log\left[\frac{beta(\alpha+y, m_i+\beta-y)}{beta(\alpha,\beta)}\right]
\end{aligned}$$

I use all of the three ways to do in my code. Note that the Hessian matrix of **optim()** function is valid for only the unconstrained problem even if the box constraints are active[2]. Thus, we use the **expit()** function to transform the parameters in order to drop the constraints before optimization. Also, I use **nlm()** function to double check the answers. The results are stated as below,

[1] http://stat.ethz.ch/R-manual/R-patched/library/stats/html/summary.glm.html

[2] **optim()** help documentation

| Model | Estimate | 95% CI | Prob.$\hat{p}$ |
|---|---|---|---|
| Binomial | - | - | 0.0893 |
| Quasi-binomial | $\hat{\phi}$ = 1.34 | - | 0.0893 |
| VGLM beta-binomial | $\hat{\rho}$ = 0.0818 | (0.0376, 0.169) | 0.0903 |
| Optimization beta-binomial<br>(logit link on both $\rho$ and $P$) | $\hat{\rho}$= 0.0818 | (0.0367, 0.169) | 0.0903 |
| Optimization beta-binomial<br>(exp link on both $\alpha$ and $\beta$ with no Delta Method) | $\hat{\rho}$= 0.0818 | (0.0366, 0.173) | 0.0903 |
| Optimization beta-binomial<br>(no link on $\rho$ and $P$ ) | $\hat{\rho}$= 0.0818 | (0.0179, 0.146) | 0.0903 |

## 1.3 Part(c)

$$\begin{aligned} Y_i|p_i, baseage &\sim binomial(m_i, p_i) \\ p_i &\sim beta(\alpha, \beta) \\ \mu_{bern.} = E(p_i) &= expit\,(intercept + slope \times baseline) \end{aligned}$$

Thus, we can calculate a few variables to pass to our optimization scheme.

$$\begin{aligned} \mu_{bern.} &= expit(intercept + slope \times baseage) \\ \alpha &= \mu_{bern.}(\frac{1}{\rho} - 1) \\ \beta &= (1 - \mu_{bern.})(\frac{1}{\rho} - 1) \end{aligned}$$

Similar to part(b), we have three options to calculate the log-likelihood, as below

$$\begin{aligned} P(Y_i = y) &= \textbf{dbetabinom}\,[x = y, size = m_i, prob \equiv \mu_{bern.} = expit(intercept + slope \times baseage), \rho = \rho] \\ &= \textbf{dbetabinom.ab}\,[x = y, size = m_i, shape1 = \alpha, shape2 = \beta] \\ &= \binom{m_i}{y} \cdot \frac{beta(\alpha + y, m_i + \beta - y)}{beta(\alpha, \beta)} \\ log - likelihood &\sim -\sum log\,\{\textbf{dbetabinom}\,[\ldots]\} \\ &\sim -\sum log \binom{m_i}{y} - \sum log \left[\frac{beta(\alpha + y, m_i + \beta - y)}{beta(\alpha, \beta)}\right] \end{aligned}$$

In this part, I use both **optim()** and **nlm()** functions for minimizing the log-likelihood. They return the very similar values. The results are stated as below,

| Model | Estimate | 95% CI | Prob.$\hat{p}$ at baseage |
|---|---|---|---|
| Quasi-binomial | $\hat{\phi}$ = 1.194 | - | expit(-2.55-0.0274*baseage) |
| VGLM beta-binomial | $\hat{\rho}$ = 0.0622 | (0.0239, 0.153) | expit(-2.52-0.0264*baseage) |
| Optimization beta-binomial, logit link on both $\rho$ and $mu$ | $\hat{\rho}$= 0.0622 | (0.0222, 0.162) | expit(-2.52-0.0264*baseage) |

## 1.4 Part(d)

- The estimated dispersion parameter of the quasi-likelihood models decreases from 1.34 to 1.194 if the baseline age is included in our model. The estimate is closer to 1 thereofore. I think that the baseline age explains some information on the variance not explained by the binomial model.
- Also, the estimated within-child correlation decreases from 0.0818 to 0.0622, if the baseline age is included in our model.

### 1.5 Part(e)

$$\begin{aligned} Y_i|p_i, baseage, xerophthalmia &\sim binomial(m_i, p_i) \\ p_i &\sim beta(\alpha, \beta) \end{aligned}$$

Xerophthalmia status can not be put into the model because it is not constant in different visits of the same ID. For each ID, when combining the data to a binomial structure, I relabeled the xero to 1 if there is at least one 1 in each visit. The estimated dispersion factor from the quasi-binomial is 1.19439. This number is very close to 1.194 in our quasi-binomial model in part(c) without xero. So there is not evidence that xero is strongly associated with respiratory infections in our data.

## 2 Pr2

My previous work is based on an Native American survey data from several reservations of North Dakota. The spatial correlation is very obvious in the data.

### 2.1 Part(a)

We were interested in the socioeconomic status of the residents in those regions. The data frame includes their house location, householder's age, number of children, income, expense, and many other variables. Our primary interest is to find the cluster/region of the low income residents.

### 2.2 Part(b)

The survey were conducted in person in order to maintain a high response rate. The mechanism is based on the stratified sampling, but with a few improvements and modifications.

### 2.3 Part(c)

Since the randomness is guaranteed in the stratified mechanism, we can say that the data is valid for our research. However, there may be some correlation between house neighbors. Although the sampling is stratified on a basis of buildings to reduce some spatial correlation within the same building, there could still be a bunch of households living very closely to each other.

### 2.4 Part(d)

I would say that there may be evidence that people on average tend to have a higher socioeconomic status in rich neighborhood, and that people live in remote places tend to have a lower socioeconomic status. From my discovery, the road status is crucial as well as the coordinates. The households connected to a road with a better condition tend to have a higher socioeconomic status. So the street/road condition can be a good substitute of coordinates.

## 3 Pr3

### 3.1 Part(a)

$$\hat{\beta} = (X^T X)^{-1} X^T Y$$

$$\begin{aligned} Var(\hat{\beta}) &= Var\left[(X^T X)^{-1} X^T Y\right] \\ &= (X^T X)^{-1} X^T Var(Y) X (X^T X)^{-1} \\ &= (X^T X)^{-1} X^T \Sigma X (X^T X)^{-1} \end{aligned}$$

## 3.2 Part(b)

$$\begin{aligned}
S^2 &= \frac{1}{n-p}\sum_{i=1}^{n}\left(Y_i - x_i^T\hat{\beta}\right)^2 \\
E\left(S^2\right) &= E\left[\frac{1}{n-p}\sum_{i=1}^{n}\left(Y_i - x_i^T\hat{\beta}\right)^2\right] \\
&= \frac{1}{n-p}\cdot E\left[\sum_{i=1}^{n}\left(Y_i - x_i^T\hat{\beta}\right)^2\right] \\
&= \frac{1}{n-p}\cdot E\left[\left(Y - X\hat{\beta}\right)^T\left(Y - X\hat{\beta}\right)\right] \\
&= \frac{1}{n-p}\cdot E\left[\left(Y - X(X^TX)^{-1}X^TY\right)^T\left(Y - X(X^TX)^{-1}X^TY\right)\right] \\
&= \frac{1}{n-p}\cdot E\left[Y^T\left(I - X(X^TX)^{-1}X^T\right)Y\right]
\end{aligned}$$

Recell that the projection matrix P of X onto the plane spanned by X is X, and then,

$$\begin{aligned}
P &\equiv X(X^TX)^{-1}X^T \\
X(X^TX)^{-1}X^T\cdot X &= X \\
\left(I - X(X^TX)^{-1}X^T\right)\cdot X &= X - X = 0
\end{aligned}$$

Hence,

$$\begin{aligned}
E\left(S^2\right) &= \frac{1}{n-p}\cdot E\left[Y^T\left(I - X(X^TX)^{-1}X^T\right)Y\right] \\
&= \frac{1}{n-p}\cdot E\left[Y^T\right]\cdot\left(I - X(X^TX)^{-1}X^T\right)\cdot E\left[Y\right] + \frac{1}{n-p}trace\left[\left(I - X(X^TX)^{-1}X^T\right)Var(Y)\right] \\
&= \frac{1}{n-p}\cdot E\left[Y^T\right]\cdot\left(I - X(X^TX)^{-1}X^T\right)\cdot X\beta + \frac{1}{n-p}trace\left[\left(I - X(X^TX)^{-1}X^T\right)\Sigma\right] \\
&= \frac{1}{n-p}\cdot E\left[Y^T\right]\cdot\left[\left(I - X(X^TX)^{-1}X^T\right)\cdot X\right]\beta + \frac{1}{n-p}trace\left[\left(I - X(X^TX)^{-1}X^T\right)\Sigma\right] \\
&= 0 + \frac{1}{n-p}trace\left[\left(I - X(X^TX)^{-1}X^T\right)\Sigma\right] \\
&= \frac{\sigma^2}{n-p}trace\left[\left(I - X(X^TX)^{-1}X^T\right)R\right]
\end{aligned}$$

## 3.3 Part(c)

$$E\left(S^2\right) = \begin{cases}\frac{\sigma^2}{n-p}trace\left[\left(I - X(X^TX)^{-1}X^T\right)R\right] & if\ not\ homoskedasticity \\ \frac{\sigma^2}{n-p}trace\left[\left(I - X(X^TX)^{-1}X^T\right)I\right] = \sigma^2 & if\ homoskedasticity\end{cases}$$

The trace of the $\left(I - X(X^TX)^{-1}X^T\right)R$ will determine whether $E\left(S^2\right)$ underestimates or overestimates $\sigma^2$. If the errors are independent and homoskedasticity, the $R = I$ and thus $E\left(S^2\right) = \sigma^2$.

Otherwise, things will be a little complicated, as following,

$$E\left(S^2\right) = \begin{cases} underestimates\ \sigma^2 & if\ trace\left[\left(I - X(X^TX)^{-1}X^T\right)R\right] < n-p \\ overestimates\ \sigma^2 & if\ trace\left[\left(I - X(X^TX)^{-1}X^T\right)R\right] > n-p. \end{cases}$$

### 3.4 Part(d)

Two sets of simulation are done with 500 times of simulations. We set the significant level is 5%.

If the correlation is positive, the coverage for both $\beta_0$ and $\beta_1$ are above 95%. This is not good, because it over-estimate the true standard error.

If the correlation is negative, the coverage for both $\beta_0$ and $\beta_1$ are below 95%. This is not good, because it under-estimate the true standard error.

If the correlation is zero, the coverage for both $\beta_0$ and $\beta_1$ are around 95%. This is accurate.

The graphs for the simulations are below,

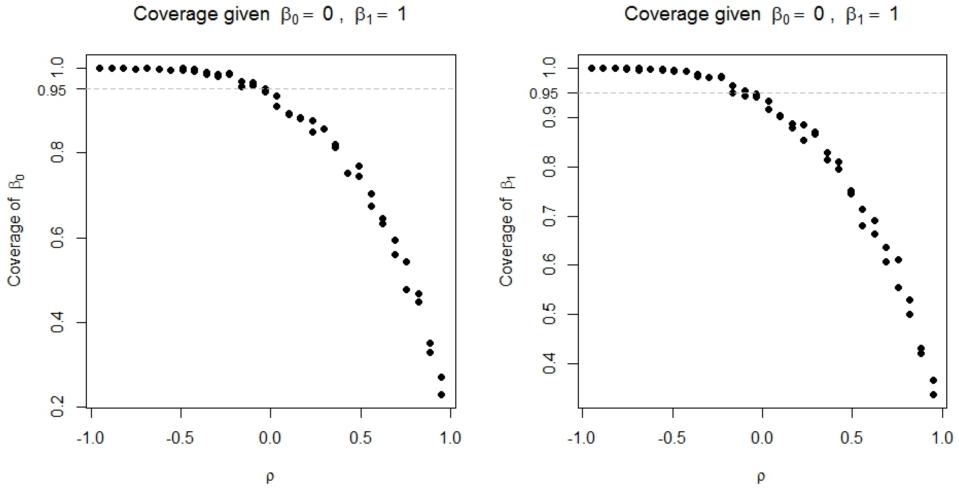

Figure 1: $\beta_0 = 0$ and $\beta_1 = 1$

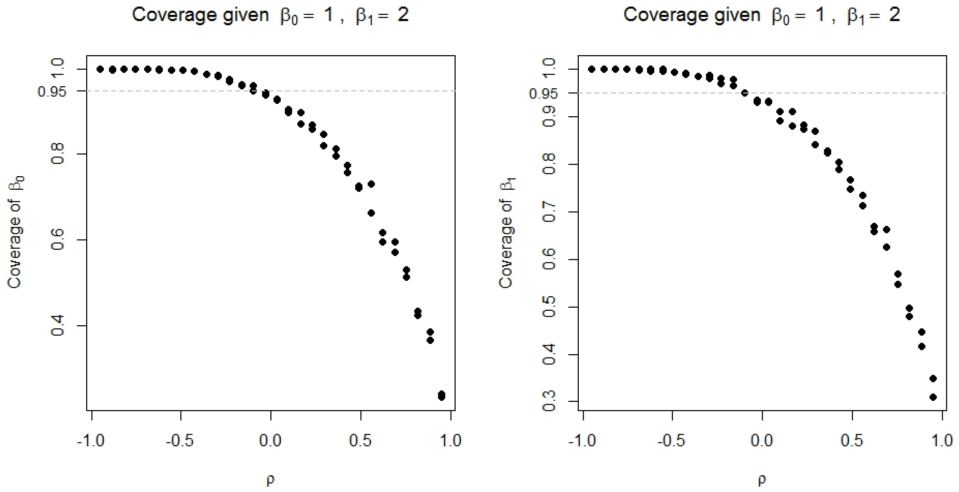

Figure 2: $\beta_0 = 1$ and $\beta_1 = 2$

# 4 Pr4

### 4.1 Part(a)

There are three types of variances of $\bar{y}_t$. $\rho$ is fixed.

- The first type. The boundaries of time interval is fixed. The interval is divided into equal spaces. As $n$ increases,
  - the length for each segment of the time diveded decreases;
  - the first type of variance first decreases and then increases.
- The second type. The boundaries of time interval is not fixed. The interval is divided into equal spaces. As $n$ increases,
  - the total length of the time increases;
  - the second type of variance decreases to zero.
- The third type. The boundaries of time interval is not fixed. The interval is divided into equal spaces. As $n$ increases,
  - the total length of the time increases;
  - the third type of variance increases to a limit of $\sigma^2$.

## 4.2 Part(b)

## Appendix

```
##Problem 1
setwd("d:/Dropbox/Dropbox/571/GWXHW2/")
ichs = read.table("ICHS.txt", stringsAsFactors=F)
ichs = ichs[-which(ichs$id==161013 & ichs$baseage==11),]
nrow(ichs)
# Overdispersed
fit.1 =  glm(infect ~ 1, family=quasibinomial, data=ichs)
# names(summary(fit.1))
summary(fit.1)$dispersion
length(ichs$infect)
1197/1198*summary(fit.1)$dispersion

head(ichs)
## df: bernoulli ---> binomial
n.bin = length(unique(ichs$id)) #275 children
ichs.bin = data.frame(id=unique(ichs$id))
for (i in 1:n.bin){
  ichs.bin$baseage[i]=mean(subset(ichs, id == unique(ichs$id)[i])$baseage)
  ichs.bin$xero[i] = max(subset(ichs, id == unique(ichs$id)[i])$xero)
  ichs.bin$n.obs[i] = length(subset(ichs, id == unique(ichs$id)[i])$infect)
  ichs.bin$n.infect[i] = sum(subset(ichs, id == unique(ichs$id)[i])$infect)
}
head(ichs.bin)
## b)
expit(coef(glm(cbind(n.infect, n.obs-n.infect)~1, family=binomial, data=ichs.bin)))
# 1. quasi-likelihood
fit.quasi=glm(cbind(n.infect, n.obs-n.infect)~1, family=quasibinomial, data=ichs.bin)
summary(fit.quasi)
expit(coef(fit.quasi))
vcov(fit.quasi)
# 2. Beta-Binomial using vglm()
fit.vglm =vglm(cbind(n.infect, n.obs-n.infect)~1, family=betabinomial, data=ichs.bin)
summary(fit.vglm)
expit(coef(fit.vglm))
hessian = solve(vcov(fit.vglm))
# vcov(summary(fit.vglm))
expit(coef(fit.vglm))[2]
rho.confidence.vglm = expit(coef(fit.vglm)[2] + c(0,-1, 1)*1.96*sqrt(vcov(fit.vglm)[2, 2]))
rho.confidence.vglm

# 3. Beta-Binomial using direct optimization of the log-likelihood
log.like =  function (x,size,exp.param){
  alpha = exp(exp.param[1])
  beta = exp(exp.param[2])
  rho = 1/(alpha +beta +1)
  mu =  alpha/(alpha+beta)

  # o1: by alpha, beta

  # o2: or by formula of posterior

  # o3: or directly by prob and rho
  o1 =  log(dbetabinom.ab(x  = x, size = size, shape1 = alpha, shape2 = beta))
  o2 =  lchoose(size,x) +lbeta(alpha+x, size+beta-x) - lbeta(alpha, beta)
  o3 =  log(dbetabinom(x=x,size=size,prob=mu,rho=rho))
  # o1 = o2 = o3
  # cbind(o1,o2,o3,o1==o2)
  out =  -sum(o3)
  return(out)
}
# way 1 use optim() without constraints.
# way 2 use nlm()
hand.way1 = optim(par=c(0.1,  2.32),
                log.like,
                method="BFGS",
                x=ichs.bin$n.infect,
                size=ichs.bin$n.obs,
                hessian=TRUE)
hand.way2 = nlm(f=log.like, p=c(1,1),
            x=ichs.bin$n.infect,
            size=ichs.bin$n.obs,
            hessian=TRUE)
exp(rbind(hand.way1$par,hand.way2$estimate))
exp.alpha.est = hand.way1$par[1]
exp.beta.est = hand.way1$par[2]
std.err1 = sqrt(diag(solve(hand.way1$hessian)))
std.err2 = sqrt(diag(solve(hand.way2$hessian)))
std.err = 1/2*(std.err1+std.err2)
exp.alpha.confidence = exp.alpha.est+c(0,-1, 1)*1.96*std.err[1]
exp.beta.confidence  = exp.beta.est+c(0,-1, 1)*1.96*std.err[2]
alpha.confidence =  exp(exp.alpha.confidence)
beta.confidence =  exp(exp.beta.confidence)
# with no delta method: approximate
rho.confidence = 1/(1+alpha.confidence+beta.confidence)
rho.confidence
alpha.est = exp(exp.alpha.est )
beta.est = exp(exp.beta.est)
mu.est = alpha.est/(beta.est+alpha.est)
rho.est = 1/(beta.est+alpha.est+1)
rho.est

## Way3 optim() with constraints
log.like2 =  function (x,size,param){
  mu = param[1]
  rho = param[2]
  o3 =  log(dbetabinom(x=x,size=size,prob=mu,rho=rho))
  out =  -sum(o3)
  return(out)
}
hand.way3 = optim(par=c(0.5,  0.5),
                  log.like2,
                  method="L-BFGS-B",
                  lower = 1e-9, upper = 1 -1e-9 ,
                  x=ichs.bin$n.infect,
                  size=ichs.bin$n.obs,
                  hessian=TRUE)
solve(hand.way3$hessian) #hessian not valid for constrained problem
hand.way3$par[2]+c(0,-1,1)*1.96*sqrt(solve(hand.way3$hessian))[2,2]


## Way4: optim without constraints
log.like3 =  function (x,size,param){
  mu = expit(param[1])
  rho = expit(param[2])
  o3 =  log(dbetabinom(x=x,size=size,prob=mu,rho=rho))
  out =  -sum(o3)
  return(out)
}
hand.way4 = optim(par=c(0.5,  0.5),
                  log.like3,
                  method="L-BFGS-B",
#                   lower = 1e-9, upper = 1 -1e-9 ,
                  x=ichs.bin$n.infect,
                  size=ichs.bin$n.obs,
                  hessian=TRUE)
hand.way4$par
hand.way4$hessian
solve(hand.way4$hessian) # var
# recall in way 2.
expit(coef(fit.vglm))
vcov(fit.vglm) #var using way 2.

rho.confidence4 = expit(hand.way4$par[2]+c(0,-1,1)*1.96*sqrt(vcov(fit.vglm))[2,2])
rho.confidence4
hand.way5 = nlm(f = log.like3,
                p = c(0.5,0.5),
                x=ichs.bin$n.infect,
                size=ichs.bin$n.obs,
                hessian=TRUE)

## Compare the four hessian matrices by viewing its inverse
vcov(fit.vglm) # The reference Cov(prob,rho) matrix
solve(hand.way3$hessian) # optim() with constaints
solve(hand.way4$hessian) # optim() without constaints
solve(hand.way5$hessian) # nlm()

## c)
# 1. quasi-likelihood
fitc.quasi=glm(cbind(n.infect, n.obs-n.infect)~ baseage, family=quasibinomial, data=ichs.bin)
expit(coef(fitc.quasi))
summary(fitc.quasi)
vcov(fitc.quasi)
# 2. Beta-Binomial using vglm()
fitc.vglm =vglm(cbind(n.infect, n.obs-n.infect)~baseage, family=betabinomial, data=ichs.bin)
summary(fitc.vglm)
solve(vcov(fitc.vglm))
# vcov(summary(fit.vglm))
expit(coef(fitc.vglm))
```

```
rho.c.confidence.vglm = expit(coef(fitc.vglm)[2] + c(0,-1, 1)*1.96*sqrt(vcov(fitc.vglm)[2,2]))
rho.c.confidence.vglm

# 3. Beta-Binomial using direct optimization of the log-likelihood
log.like.c =  function (x,size,param,baseage){
  intercept = param[1]
  slope = param[3]
  rho = expit(param[2])
  mu = expit(intercept + slope*baseage)
  alpha = mu*(1/rho-1)
  beta = (1-mu)*(1/rho-1)
  # by alpha, beta
  o1 =  log(dbetabinom.ab(x  = x, size = size, shape1 = alpha, shape2 = beta))
  # or by formula of posterior
  o2 =  lchoose(size,x) +lbeta(alpha+x, size+beta-x) - lbeta(alpha, beta)
  # or directly by prob and rho
  o3 =  log(dbetabinom(x=x, size=size, prob= expit(intercept + slope*baseage), rho=rho))
  # o1==o2==o3
  out =  -sum(o3)
  return(out)
}
# way 1 and way 2
hand.way1.c = optim(par=c(0.5,0.5,0.5),
                  log.like.c,
                  method="BFGS",
                  x=ichs.bin$n.infect,
                  size=ichs.bin$n.obs,
                  baseage = ichs.bin$baseage,
                  hessian=TRUE)
hand.way2.c = nlm(f=log.like.c,
                  p=c(0.5,0.5,0.5),
                  x=ichs.bin$n.infect,
                  size=ichs.bin$n.obs,
                  baseage = ichs.bin$baseage,
                  hessian=TRUE)
expit(rbind(hand.way1.c$par,hand.way2.c$estimate))
hand.way1.c$hessian
hand.way2.c$hessian
expit(hand.way1.c$par)
hand.way1.c$par
hand.way2.c
se.rho.c = sqrt(solve(hand.way2.c$hessian))[2,2]
rho.est.c = expit(rbind(hand.way1.c$par,hand.way2.c$estimate))[2]
rho.confidence.c = expit(hand.way2.c$estimate[2]+c(0,-1,1)*1.96*se.rho.c)
rho.confidence.c

## part(d)
table.d=rbind(rho.confidence.vglm,rho.confidence4,rho.c.confidence.vglm,rho.confidence.c)
colnames(table.d)=c("Est.","lower ci.","upper ci.")
table.d


#e)
```

```r
## xero at least one 1, we take it as one!

# 1. quasi-likelihood
fit.d.quasi=glm(cbind(n.infect, n.obs-n.infect)~baseage + xero, family=quasibinomial, data=ichs.bin)
summary(fit.d.quasi)
vcov(fit.d.quasi)
# 2. Beta-Binomial using vglm()
fit.d.vglm =vglm(cbind(n.infect, n.obs-n.infect)~baseage + xero, family=betabinomial, data=ichs.bin)
summary(fit.d.vglm)
```