Web-Based Supplemental Code Materials for
“A Novel Modeling Framework for Ordinal Data Defined by Collapsed Counts”
by James S. McGinley, Patrick J. Curran, and Donald Hedeker

Below is R code that simulates ordinal data consisting of underlying counts and fits the proposed ONB and OZINB models. In terms of the variables, x1 and x2 are predictors, y.onb is the ordinal response with underlying counts following a negative binomial (NB) distribution, and y.ozinb is the ordinal response with underlying counts following a zero-inflated negative binomial (ZINB) distribution. For both ordinal outcomes, the counts are collapsed as 0=”0”, 1=”1-2”, 2=”3-5”, 3=”6-9”, 4=”10-19”, 5=”20-39”, and 6=”40+”.

Simulating Example Data

```r
# SAMPLE SIZE
n <- 1000

# POPULATION VALUES
b0 <- 1.5
b1 <- -.5
b2 <- .5
g0 <- .2
g1 <- .2
g2 <- -.3
alpha <- 5

# SIMULATING COUNT DATA
x1 <- rnorm(n, mean=0, sd=1)
x2 <- rnorm(n, mean=0, sd=1)
x <- cbind(1,x1,x2) # NEED THIS TO FIT MODEL LATER
w <- x # NEED THIS TO FIT MODEL LATER
mu <- exp(b0 + b1*x1 + b2*x2);
y.nb <- rnbinom(n, mu=mu, size=1/alpha)
y.zero <- plogis(g0 + g1*x1 + g2*x2)
y.zinb <- ifelse((runif(n, min=0, max=1)) > y.zero,y.nb,0)
table(y.nb)
table(y.zinb)

# COLLAPSING COUNTS INTO ORDINAL RESPONSES
cutpoints <- c(-.01,0,2.5,9,19,39,Inf) # -.01 is an arbitrary low end value needed to collapse counts with cut() fn
y.onb <- cut(y.nb, breaks=cutpoints, labels=F)-1
y.ozinb <- cut(y.zinb, breaks=cutpoints, labels=F)-1
table(y.onb)
table(y.ozinb)
```
Function to Fit ONB Model

```r
onb.lf<-function(theta,y.ordinal,X){
  n<-nrow(X)
  k<-ncol(X)
  # below are parms to be estimated
  beta<-theta[1:k]
  alpha<-theta[k+1]
  # below is the model that is fitted
  mu<-exp(X%*%beta)
  # below is the difference between the cdfs at the various cutpoints
  prob0 = pnbinom(0, size = 1/alpha, mu = mu, lower.tail = TRUE, log.p = FALSE)
  prob1 = pnbinom(2, size = 1/alpha, mu = mu, lower.tail = TRUE, log.p = FALSE)-
           (pnbinom(0, size = 1/alpha, mu = mu, lower.tail = TRUE, log.p = FALSE) )
  prob2 = pnbinom(5, size = 1/alpha, mu = mu, lower.tail = TRUE, log.p = FALSE)-
           (pnbinom(2, size = 1/alpha, mu = mu, lower.tail = TRUE, log.p = FALSE) )
  prob3 = pnbinom(9, size = 1/alpha, mu = mu, lower.tail = TRUE, log.p = FALSE)-
           (pnbinom(5, size = 1/alpha, mu = mu, lower.tail = TRUE, log.p = FALSE) )
  prob4 = pnbinom(19, size = 1/alpha, mu = mu, lower.tail = TRUE, log.p = FALSE)-
           (pnbinom(9, size = 1/alpha, mu = mu, lower.tail = TRUE, log.p = FALSE) )
  prob5 = pnbinom(39, size = 1/alpha, mu = mu, lower.tail = TRUE, log.p = FALSE)-
           (pnbinom(19, size = 1/alpha, mu = mu, lower.tail = TRUE, log.p = FALSE) )
  prob6 = 1- pnbinom(39, size = 1/alpha, mu = mu, lower.tail = TRUE, log.p = FALSE)
  p <- ifelse (y.ordinal == 0, prob0, 
               ifelse (y.ordinal == 1, prob1, 
               ifelse(y.ordinal == 2, prob2, 
               ifelse(y.ordinal == 3, prob3, 
               ifelse(y.ordinal == 4, prob4, 
               ifelse(y.ordinal == 5, prob5, 
               prob6))))))
  ll = sum(log(p))
  return(-ll)
}
```
Function to Fit OZINB Model

```r
ozinb.lf<-function(theta,y.ordinal,X,W){
n<-nrow(X)
k<-ncol(X)
t<-ncol(W)
l<-k+1
h<-k+t
# below are parms to be estimated
beta<-theta[1:k]
gamma<-theta[1:h]
alpha<-theta[h+1]
# below is the model that is fitted
## zero process
logit.pi=W%*%gamma;
inf.pi=1/(1+exp(-logit.pi));
## count process
mu<-exp(X%*%beta)
# below is the difference between the cdfs at the various cutpoints
prob0 = pnbinom(0, size = 1/alpha, mu = mu, lower.tail = TRUE, log.p = FALSE)
prob1 = pnbinom(2, size = 1/alpha, mu = mu, lower.tail = TRUE, log.p = FALSE)-
(pnbinom(0, size = 1/alpha, mu = mu, lower.tail = TRUE, log.p = FALSE) )
prob2 = pnbinom(5, size = 1/alpha, mu = mu, lower.tail = TRUE, log.p = FALSE)-
(pnbinom(2, size = 1/alpha, mu = mu, lower.tail = TRUE, log.p = FALSE) )
prob3 = pnbinom(9, size = 1/alpha, mu = mu, lower.tail = TRUE, log.p = FALSE)-
(pnbinom(5, size = 1/alpha, mu = mu, lower.tail = TRUE, log.p = FALSE) )
prob4 = pnbinom(19, size = 1/alpha, mu = mu, lower.tail = TRUE, log.p = FALSE)-
(pnbinom(9, size = 1/alpha, mu = mu, lower.tail = TRUE, log.p = FALSE) )
prob5 = pnbinom(39, size = 1/alpha, mu = mu, lower.tail = TRUE, log.p = FALSE) -
pnbinom(19, size = 1/alpha, mu = mu, lower.tail = TRUE, log.p = FALSE)
prob6 = 1- pnbinom(39, size = 1/alpha, mu = mu, lower.tail = TRUE, log.p = FALSE)
p <- ifelse (y.ordinal == 0, prob0,
  ifelse (y.ordinal == 1, prob1,
    ifelse (y.ordinal == 2, prob2,
      ifelse (y.ordinal == 3, prob3,
        ifelse (y.ordinal == 4, prob4,
          ifelse (y.ordinal == 5, prob5,
            prob6))))))
logl = ifelse(y.ordinal==0, log((1- inf.pi)*p + inf.pi), log((1- inf.pi)*p) )
ll = sum(logl)
return(-ll)
}
```
Fitting the ONB and OZINB Models

```
# ONB: FIRST 4 NUMBERS ARE START VALUES
fit.onb <- optim(c(1.5, -0.5, 0.5, 5), onb.lf, method = "BFGS", hessian = T, y.ordinal = y.onb, X = x)
fit.onb$convergence # a 0 means it converged
fit.onb$par # estimates they come out in order - b0, b1, b2, alpha
sqrt(diag(solve(fit.onb$hessian))) # stderrs they come out in order - b0, b1, b2, alpha

# OZINB: FIRST 7 NUMBERS ARE START VALUES
fit.ozinb <- optim(c(1.5, -0.5, 0.5, 2, 2, -0.3, 5), ozinb.lf, method = "BFGS", hessian = T, y.ordinal = y.ozinb, X = x, W = w)
fit.ozinb$convergence # a 0 means it converged
fit.ozinb$par # estimates they come out in order - b0, b1, b2, g0, g1, g2, alpha
sqrt(diag(solve(fit.ozinb$hessian))) # stderrs they come out in order - b0, b1, b2, g0, g1, g2, alpha
```
Below is SAS code that simulates ordinal data with underlying counts and fits the proposed ONB and OZINB models. In terms of the variables, x1 and x2 are predictors, y_onb is the ordinal response with underlying counts following a negative binomial (NB) distribution, and y_ozinb is the ordinal response with underlying counts following a zero-inflated negative binomial (ZINB) distribution. The counts are collapsed as 0=“0”, 1=“1-2”, 2=“3-5”, 3=“6-9”, 4=“10-19”, 5=“20-39”, and 6=“40+”.

Simulating Example Data

```sas
data countsim;
  do id=1 to 1000; * sample size of 1,000;
    * POPULATION VALUES;
    b0 = 1.5;
    b1 = -.5;
    b2 = .5;
    g0 = .2;
    g1 = .2;
    g2 = -.3;
    alpha = 5;
    * SIMULATING COUNT DATA;
    x1 = rannor(92983);
    x2 = rannor(55432);
    mu = exp(b0 + b1*x1 + b2*x2);
    parm1 = 1/(1+mu*alpha);
    y_nb = rand('NEGB',parm1,1/alpha);
    pzero = cdf('LOGISTIC',g0 + g1*x1 +g2*x2);
    if ranuni(3288)>pzero then do;
      y_zinb = y_nb;
    end;
    else do;
      y_zinb = 0;
    end;
  output;end;
run;
* COLLAPSING COUNTS INTO ORDINAL RESPONSES;
data ordinal;set countsim;
  if y_nb=0 then y_onb=0;
  if 1 le y_nb le 2 then y_onb=1;
  if 3 le y_nb le 5 then y_onb=2;
  if 6 le y_nb le 9 then y_onb=3;
  if 10 le y_nb le 19 then y_onb=4;
  if 20 le y_nb le 39 then y_onb=5;
  if y_nb ge 40 then y_onb=6;
  if y_zinb=0 then y_ozinb=0;
  if 1 le y_zinb le 2 then y_ozinb=1;
  if 3 le y_zinb le 5 then y_ozinb=2;
  if 6 le y_zinb le 9 then y_ozinb=3;
  if 10 le y_zinb le 19 then y_ozinb=4;
  if 20 le y_zinb le 39 then y_ozinb=5;
  if y_zinb ge 40 then y_ozinb=6;
  drop b0 b1 b2 g0 g1 g2 alpha;
run;
proc freq data=ordinal;
table y_nb y_zinb y_onb y_ozinb;run;
```
**Fitting ONB Model in NLMIXED**

```plaintext
PROC NLMIXED DATA=ordinal gconv=0;
   PARMS b0=1.5 b1=-.5 b2=.5 alpha=5;

   * THE MODEL FITTED;
   mu = exp(b0 + b1*x1 + b2*x2);

   * SETTING UP THE CDFS TO EXPRESS LIKELIHOOD;
   ARRAY pdf[40] pdf0-pdfl39;
   do v1=1 to 40;
      v=v1-1;
      pdf[v1] = gamma(v+alpha**-1) / (gamma(alpha**-1)*gamma(v+1)) * (alpha*mu)**v * 
                 (1+alpha*mu)**-(v+alpha**-1);
   end;
   CDF_CPl=sum(of pdf0);
   CDF_CP2=sum(of pdf0-pdfl2);
   CDF_CP3=sum(of pdf0-pdfl5);
   CDF_CP4=sum(of pdf0-pdfl9);
   CDF_CP5=sum(of pdf0-pdfl19);
   CDF_CP6=sum(of pdf0-pdfl39);

   * EXPRESSING PROBABILITY OF BEING IN A GIVEN CATEGORICAL AS A FUNCTION OF CDFS;
   IF (y_onb=0) THEN p=CDF_CP1;
   ELSE IF (y_onb=1) THEN p = CDF_CP2 - CDF_CP1;
   ELSE IF (y_onb=2) THEN p = CDF_CP3 - CDF_CP2;
   ELSE IF (y_onb=3) THEN p = CDF_CP4 - CDF_CP3;
   ELSE IF (y_onb=4) THEN p = CDF_CP5 - CDF_CP4;
   ELSE IF (y_onb=5) THEN p = CDF_CP6 - CDF_CP5;
   ELSE IF (y_onb=6) THEN p = 1 - CDF_CP6;

   ll = LOG(p);

   MODEL y_onb ~ GENERAL(ll);
RUN;
```
Fitting OZINB Model in NLMIXED

PROC NLMIXED DATA=ordinal gconv=0;
    PARMS b0=1.5 b1=-.5 b2=.5 g0=.5 g1=.2 g2=-.3 alpha=5;

    * THE MODEL FITTED;
    *zero process;
    logit_pi=g0 + g1*x1 + g2*x2;
    inf_pi= 1/(1+exp(-logit_pi));
    *count process;
    mu=exp(b0 + b1*x1 + b2*x2);

    * SETTING UP THE CDFS TO EXPRESS LIKELIHOOD;
    ARRAY pdf[40] pdf0-pdf39;
    do v1=1 to 40;
        v=v1-1;
        pdf[v1] = gamma(v+alpha**-1) / (gamma(alpha**-1)*gamma(v+1)) * (alpha*mu)**v *
                   (1+alpha*mu)**-(v+alpha**-1);
    end;
    CDF_CP1=sum(of pdf0);
    CDF_CP2=sum(of pdf0-pdf2);
    CDF_CP3=sum(of pdf0-pdf5);
    CDF_CP4=sum(of pdf0-pdf9);
    CDF_CP5=sum(of pdf0-pdf19);
    CDF_CP6=sum(of pdf0-pdf39);

    * EXPRESSING PROBABILITY OF BEING IN A GIVEN CATEGORICAL AS A FUNCTION OF CDFS;
    IF (y_ozinb=0) THEN p=CDF_CP1;
    ELSE IF (y_ozinb=1) THEN p = CDF_CP2 - CDF_CP1;
    ELSE IF (y_ozinb=2) THEN p = CDF_CP3 - CDF_CP2;
    ELSE IF (y_ozinb=3) THEN p = CDF_CP4 - CDF_CP3;
    ELSE IF (y_ozinb=4) THEN p = CDF_CP5 - CDF_CP4;
    ELSE IF (y_ozinb=5) THEN p = CDF_CP6 - CDF_CP5;
    ELSE IF (y_ozinb=6) THEN p = 1 - CDF_CP6;

    IF (y_ozinb=0) THEN ll=LOG((1-inf_pi)*p + inf_pi);
    ELSE IF (y_ozinb gt 0) THEN ll=LOG((1-inf_pi)*p);
    MODEL y_ozinb ~ GENERAL(ll);
RUN;