Package 'lmeNB'

February 20, 2015

Type Package

Title Compute the Personalized Activity Index Based on a Negative Binomial Model

Version 1.3

Date 2015-01-14

Author Yinshan Zhao and Yumi Kondo (with contributions from Steven G. Johnson, Rudolf Schuerer and Brian Gough on the integration subroutines)

Maintainer Yumi Kondo <y.kondo@stat.ubc.ca>

Description The functions in this package implement the safety monitoring procedures proposed in the paper titled ``Detection of unusual increases in MRI lesion counts in individual multiple sclerosis patients" by Zhao, Y., Li, D.K.B., Petkau, A.J., Riddehough, A., Traboulsee, A., published in Journal of the American Statistical Association in 2013. The procedure first models longitudinally collected count variables with a negative binomial mixed-effect regression model. To account for the correlation among repeated measures from the same patient, the model has subject-specific random intercept, which can be modelled with a gamma or log-normal distributions. One can also choose the semi-parametric option which does not assume any distribution for the random effect. These mixed-effect models could be useful beyond the application of the safety monitoring. The maximum likelihood methods are used to estimate the unknown fixed effect parameters of the model. Based on the fitted model, the personalized activity index is computed for each patient. Lastly, this package is companion to R package lmeNBBayes, which contains the functions to compute the Personalized Activity Index in Bayesian framework.

License GPL (>= 2)

Copyright The portion of numerical integration is based on Cubature, Copyright (c) 2005-2013 Steven G. Johnson (http://ab-initio.mit.edu/wiki/index.php/Cubature), which in turn is based on HIntLib (also distributed under * the GNU GPL, v2 or later), copyright (c) 2002-2005 Rudolf Schuerer and GNU GSL (also distributed under the GNU GPL, v2 or later), copyright (c) 1996-2000 Brian Gough.(http://www.gnu.org/software/gsl/).

LazyData true

Depends numDeriv,statmod,lmeNBBayes

NeedsCompilation yes

Repository CRAN

Date/Publication 2015-02-02 22:40:23

R topics documented:

`CP.ar1.se` *Compute a conditional probability of observing a set of counts as extreme as the new observations of a subjectvisit given the previous observations of the same subject based on the negative binomial mixed-effect AR(1) model.*

Description

Given the parameter estimates of $\alpha, \theta, \delta, \beta_0, \beta_1, \cdots$ of the negative binomial mixed effect AR(1) model, these functions compute the following conditional probability:

$Pr(q(\boldsymbol{Y}_{i,new}) \geq q(\boldsymbol{y}_{i,new}) | \boldsymbol{Y}_{i,pre} = \boldsymbol{y}_{i,pre})$,

where $\boldsymbol{y}_{i,new}$ and $\boldsymbol{y}_{i,pre}$ are vectors of previous and new observations from subject i and $q()$ is a function which provides a scalar summary of the new observations.

These functions are subroutines of `index.batch`.

`CP.ar1.se` returns the estimate of the conditional probability and its asymptotic standard error of a subject based on AR(1) model. The evaluations for the probability is done by its subroutine `jCP.ar1`, which, in turn, has two subroutines `CP1.ar1` and `MCCP.ar1`. `CP1.ar1` computes the probability via the adaptive quadrature while `MCCP.ar1` computes the probability via the Monte Carlo integration.

Usage

```
CP.ar1.se(tpar, ypre, ynew, y2m = NULL, XM, stp,
  RE = "G", V, MC = FALSE, qfun = "sum",i.tol=1E-75)
```

```
jCP.ar1(tpar, ypre, ynew, y2m = NULL, XM, stp, RE = "G", LG =FALSE,
      MC = FALSE, N.MC = 40000, qfun = "sum", oth =NULL,i.tol=1E-75)

CP1.ar1(ypre, ynew, y2m=NULL, stp, u, th, a, dt, RE = "G", oth,qfun,i.tol=1E-75)

MCCP.ar1(ypre, ynew, stp, u, th, a, dt, RE = "G", N.MC = 1000, oth, qfun = "sum")
```

Arguments

tpar	A vector of length 4 + # covariates, containing the estimates of the model in the order that $\log(\alpha), \log(\theta), logit(\delta), \beta_0, \beta_1, \cdots$. If the semi-parametric approach is taken, then theta is a place holder and can be any value.
ypre	A vector of the length the number of previous observations, containing counts on pre-scans.
y2m	Internal use only. Set as y2m=NULL.
ynew	A vector of length the number of new observations, containing counts on new scans.
XM	See CP.se.
stp	A vector of length n_i, containing index to indicates missing scans. The first entry must be zero. For example, if there is no missing scans and there are five repeated measures, then stp=c(0,1,1,1,1). If the third scan is missing and there are four repeated measures, then stp=c(0,1,2,1).
RE	See lmeNB. Note that this option is NOT accepted in CP.ar1.se.
LG	See CP.se.
MC	If TRUE then the function MCCP.ar1 is called and the Monte carlo integration is performed to integrate out the random effect. Fast but could be unreliable; not recommended for computing the confidence intervals. If FALSE then the function CP1.ar1 is called and the adaptive quadrature is performed. Slow but reliable.
N.MC	The number of the Monte Carlo integration. Necessary if MC=TRUE.
qfun	See index.batch.
oth	See CP.se. If RE="NoN", othr must be the frequency table of the random effects, which can be obtained based on dist=obj$gi where obj is the output of fitSemiAR1.
V	See CP.se.
th	The estimated *theta*.
a	The estimated *alpha*.
dt	The estimated *delta*.
u	A vector of length the number of repeated measures, containing the estimated mean counts $(\mu_{i1}, \cdots, \mu_{in_i})$. If the mean of *Y_ij* is modeled linearly on beta with the log-link function, then u=exp(beta0+ XM[,1]*beta1+XM[,2]*beta2+...)).
i.tol	See lmeNB

Author(s)

Zhao, Y. and Kondo, Y.

References

Detection of unusual increases in MRI lesion counts in individual multiple sclerosis patients. (2013) Zhao, Y., Li, D.K.B., Petkau, A.J., Riddehough, A., Traboulsee, A., Journal of the American Statistical Association.

See Also

The main function to fit the Negative Binomial mixed-effect model: lmeNB,

The internal functions of lmeNB for fitting relevant models: fitParaIND, fitParaAR1, fitSemiIND, fitSemiAR1,

The other subroutines of index.batch to compute the conditional probability index:

CP.se, jCP,

The functions to generate simulated datasets: rNBME.R.

Examples

```
## Not run:
ilgt <- function (x)
{
    tem = exp(x)
    res = tem/(1 + tem)
    return(res)
}
lgt <- function (p)
{
    log(p/(1 - p))
}
## the vector of a parameter estimates if log(a),log(theta),logit(delta),beta0.
tpar  <- c(log(2),log(0.5),lgt(0.5),2)
ypre <- c(0, 1)
ynew <- c(1, 0, 0)
## No covariate
XM <- NULL
## no missing visit
stp <- c(0,1,1,1,1)
RE <- "G"
## The estimate of the variance covariance matrix
V <-
matrix(
c( 0.17720309, -0.240418504,  0.093562548,  0.009141980,
  -0.24041850,  0.605132808, -0.160454773, -0.003978118,
   0.09356255, -0.160454773,  0.095101658,  0.005661923,
   0.00914198, -0.003978118,  0.005661923,  0.007574769),
nrow=4)
```

```
## the estimate of the conditional probability based on the sum summary statistics and its SE
CP.ar1.se(tpar = tpar, ypre = ypre, ynew = ynew,
  XM =XM, stp = stp,
  RE = RE, V = V, MC = FALSE, qfun = "sum")

## the estimate of the conditional probability based on the max summary statistics and its SE
CP.ar1.se(tpar = tpar, ypre = ypre, ynew = ynew,
  XM =XM, stp = stp,
  RE = RE, V = V, MC = FALSE, qfun = "max")


## CP.ar1.se calls for jCP.ar1 to compute the estimate of the conditional probability
## the estimate of the conditional probability based on the sum summary statistics
jCP.ar1(tpar = tpar, ypre = ypre, ynew = ynew,
y2m=NULL,  XM =XM, stp = stp,
        RE = RE, LG = FALSE, MC = FALSE, N.MC = 40000, qfun = "sum", oth = NULL)

## jCP.ar1 calls for CP.ar1 to compute the estimate of the conditional probability
## via the adaptive quadrature (MC=F)
## the estimate of the conditional probability

u <- rep(exp(tpar[4]),length(ypre)+length(ynew))

CP1.ar1(ypre = ypre, ynew =ynew,
stp =stp, u = u, th = exp(tpar[2]), a = exp(tpar[1]),
dt= ilgt(tpar[3]), RE = RE, qfun = "sum")


## jCP.ar1 calls for CP.ar1 to compute the estimate of the conditional probability
## via the Monte Carlo method (MC=T)
## the estimate of the conditional probability
MCCP.ar1(ypre = ypre, ynew =ynew, stp = stp,
 u = u, th = exp(tpar[2]), a = exp(tpar[1]),  dt = ilgt(tpar[3]),
 RE = RE, N.MC = 1000, qfun = "sum")

## End(Not run)
```

CP.se — *Compute a conditional probability of observing a set of counts as extreme as the new observations of a subject given the previous observations from the same subject based on the negative binomial mixed effect independent model.*

Description

Given the parameter estimates of $\alpha, \theta, \beta_0, \beta_1, \cdots$ of the negative binomial mixed effect AR(1) model, these functions compute the following conditional probability:

$Pr(q(\boldsymbol{Y}_{i,new}) \geq q(\boldsymbol{y}_{i,new}) | \boldsymbol{Y}_{i,pre} = \boldsymbol{y}_{i,pre})$

where $\boldsymbol{y}_{i,new}$ and $y_{i,pre}$ are vectors of previous and new observations from subject i and $q()$ is a function which provides a scalar summary of the new observations. These functions are subroutines of index.batch. CP.se returns the estimate of the conditional probability of single subject and the asymptotic standard error of the logit of the estimate of the conditional probability based on the independent model. The computation for of the probability is done by its subroutine jCP.

Usage

```
CP.se(tpar, Y1, Y2, sn1, sn2, XM = NULL,
      RE = "G", V, qfun ="sum",i.tol)
jCP(tpar, Y1, Y2, sn1, sn2, XM = NULL,
    RE = "G", LG =FALSE, oth = NULL, qfun = "sum",i.tol)
```

Arguments

tpar	A vector of length 3 + # covariates, containing the estimates of the model in the order of $\log(\alpha), \log(\theta), \beta_0, \beta_1, \cdots$. Note that α is the dispersion parameter and the θ is a variance estimate of the random effect. If the semi-parametric approach is taken, then $\log(\theta)$ is a place holder and can be any number.
Y1	A scalar containing the sum of the previously observed response counts of a subject.
Y2	A scalar containing the summary statistics of the newly observed response counts of a subject $q(y_{i,new})$
sn1	The number of previous observations.
sn2	The number of new observations.
XM	A n_i by # covariates matrix containing the covariate values of subject i, where whre n_i is the total number of previous and new observations. If there is no covariate, i.e., the model only has an intercept term, then set XM=NULL.
RE	See lmeNB. Note that the semiparametric model is NOT accepted in CP.se. For jCP, if dist="NoN", then the conditional probability is computed by assuming the random effect is from a distribution represented by the argument oth.
V	The variance covariance matrix of the parameters tpar.
qfun	See index.batch.
oth	If RE= "G" or "N", then oth should be NULL. If RE="NoN", oth must be the frequency table of the random effects. i.e.,RE=obj$gtb where obj is the output of fitSemiIND.
LG	If LG=TRUE, then the logit of the conditional probability is returned.
i.tol	See lmeNB

Author(s)

Zhao, Y. and Kondo, Y.

References

Detection of unusual increases in MRI lesion counts in individual multiple sclerosis patients. (2013) Zhao, Y., Li, D.K.B., Petkau, A.J., Riddehough, A., Traboulsee, A., Journal of the American Statistical Association.

See Also

The main function to fit the model is: lmeNB,

The internal functions of lmeNB for fitting relevant models:

fitParaIND, fitParaAR1, fitSemiIND, fitSemiAR1,

The other subroutines of index.batch to compute the conditional probability index:

MCCP.ar1, CP.ar1.se, CP.se, jCP,

The functions to generate simulated datasets: rNBME.R.

Examples

```
## Not run:
## tpar contains: log(a),log(theta),beta0
tpar <-  c(0.5, -0.5, 1.3)
## A scalar containing the sum of the response counts in pre scans
Y1 <- 0
## A scalar containing the summary statistics of the response counts in new scans q(y_new)
Y2 <- 1
## The number of scans in the pre scans.
sn1 <- 2
## The number of scans in the new scans.
sn2 <- 3
## the covariate matrix
XM <- NULL
RE <- "G"
## the variance covariance matrix:
V <- matrix(
  c(0.0490673003, -0.0004481864, 0.013279476,
   -0.0004481864,  0.0245814022, 0.001231522,
    0.0132794760,  0.0012315221, 0.023888065),nrow=3)

## the estimate of the conditional probability based on the sum summary statistics and its SE
CP.se(tpar = tpar, Y1=Y1 ,Y2= Y2, sn1 = sn1, sn2 = sn2, XM = XM, RE = RE, V = V, qfun = "sum")

## the estimate of the conditional probability based on the max summary statistics and its SE
CP.se(tpar = tpar, Y1=Y1 ,Y2= Y2, sn1 = sn1, sn2 = sn2, XM = XM, RE = RE, V = V, qfun = "max")

## jCP calls for CP.se to compute the estimate of the conditional probability
jCP(tpar = tpar, Y1 = Y1, Y2 = Y2, sn1 = sn1, sn2 = sn2,
    XM = XM, RE = RE, LG = FALSE, oth = NULL, qfun = "sum")

## End(Not run)
```

`fitParaAR1` *Performs the maximum likelihood estimation for the negative binomial mixed-effect AR(1) model*

Description

This function fits a negative binomial mixed-effect AR(1) model in the formulation described Zhao et al. (2013). The conditional distribution of response counts given random effect is modelled by Negative Binomial as described in description of lmeNB. The conditional dependence among the response counts of a subject is modeled with AR(1) structure. The random effects are modelled with either gamma or log-normal distributions. See descriptions of lmeNB.

Usage

```
fitParaAR1(formula, data, ID, Vcode, p.ini = NULL, IPRT = FALSE,
           RE = "G", i.tol = 1e-75, o.tol = 0.001)
```

Arguments

formula	See lmeNB.
data	See lmeNB.
ID	See lmeNB.
Vcode	See lmeNB.
p.ini	A vector of length 4 + # covariates, containing the initial values for the parameters ($\log(\alpha)$, $\log(\theta)$, $logit(\delta)$, β_0, β_1...). NULL is accepted.
IPRT	See lmeNB.
RE	See fitParaIND.
i.tol	See lmeNB.
o.tol	See lmeNB.

Details

fitParaAR1 calls optim to minimize the negative log-likelihood of the negative binomial model with respect to the model parameters: ($\log(\alpha)$, $\log(\theta)$, $logit(\delta)$, β_0, β_1...). The Nelder-Mead algorithm is employed. The log-likelihood is obtained by marginalizing out the random effects. The numerical integration is carried out using adaptive quadrature. When missing visits are present, an approximation of the likelihood is used (see Zhao et al. (2013) for details.) All the computations are done in R.

Value

opt	See lmeNB.
nlk	See lmeNB.
V	See lmeNB.
est	See lmeNB.
AR	TRUE

Author(s)

Zhao, Y. and Kondo, Y.

References

Detection of unusual increases in MRI lesion counts in individual multiple sclerosis patients. (2013) Zhao, Y., Li, D.K.B., Petkau, A.J., Riddehough, A., Traboulsee, A., Journal of the American Statistical Association.

See Also

The main function to fit the Negative Binomial mixed-effect model: lmeNB,

The functions to fit the other models: fitParaIND,

fitSemiIND, fitSemiAR1,

The subroutines of index.batch to compute the conditional probability index: jCP.ar1, CP1.ar1, MCCP.ar1, CP.ar1.se, CP.se, jCP,

The functions to generate simulated datasets: rNBME.R.

Examples

```
## Not run:

## ============================================================================
## generate a data based on the negative binomial mixed-effect AR(1) model.
## Under this model, the response counts follows the negative binomial:
## Y_ij | G_i = g_i ~ NB(r_ij,p_i) where r_ij = exp(X^T beta)/a , p_i =1/(a*g_i+1)
## with G_i ~ Gamma(scale=th,shape=1/th)
##
## The adjacent repeated measures of the same subject are correlated
## with correlation structure:
## cov(Y_ij,Y_ij'|G_i=g_i)=d^{j-j'} E(Y_ij')*(a*g_i^2+g_i)

loga <- -0.5
logtheta<- 1.3
logitd <- -0.2
b0 <- 0.5 ## no covariates;
## 80 subjects each with 5 scans
n <- 80
sn <- 5

set.seed(1)
DT2 <-  rNBME.R(gdist = "G",
                n = n, ##  the total number of subjectss
       sn = sn,
       th=exp(logtheta),
                u1 = rep(exp(b0),sn),
       u2 = rep(exp(b0),sn),
       a = exp(loga),
       d = exp(logitd)/(1+exp(logitd))
      )
Vcode <- rep(-1:(sn-2),n) # scan number -1, 0, 1, 2, 3
ID <- DT2$id
new <- Vcode > 0
dt2 <- data.frame(CEL=DT2$y)
```

```
## ============================================================================

## 1) Fit the negative binomial mixed-effect AR(1) model
## where the random effects are from the gamma distribution
## This is the true model

re.gamma.ar1 <- fitParaAR1(formula=CEL~1,data=dt2,ID=ID,
           Vcode=Vcode,
           p.ini=c(loga,logtheta,logitd,b0),
           ## log(a), log(theta), logit(d), b0
           RE="G",
           IPRT=TRUE)



## compute the estimates of the conditional probabilities
## with sum of the new repeated measure as a summary statistics
## Note C=TRUE with i.tol=1E-3 options compute the index faster
## i.se=TRUE requires more time
Psum <- index.batch(olmeNB=re.gamma.ar1,data=dt2,ID=ID,Vcode=Vcode,
     labelnp=new,qfun="sum", IPRT=TRUE,i.se=FALSE,C=TRUE,i.tol=1E-3)


## compute the estimates of the conditional probabilities
## with max of the new repeated measure as a summary statistics
Pmax <-index.batch(olmeNB=re.gamma.ar1,data=dt2,ID=ID,Vcode=Vcode,
                  labelnp=new,qfun="max", IPRT=TRUE,i.se=FALSE,C=TRUE,i.tol=1E-3)

## Which patient's estimated probabilities based on the sum and max
## statistics disagrees the most?
( IDBigDif <- which(rank(abs(Pmax$condProbSummary[,1]-Psum$condProbSummary[,1]))==80) )
## Show the patient's CEL counts
dt2$CEL[ID==IDBigDif]
## Show the estimated conditional probabilities based on the sum summary statistics
Psum$condProbSummary[IDBigDif,]
## Show the estimated conditional probabilities based on the max summary statistics
Pmax$condProbSummary[IDBigDif,]


## 2) Fit the negative binomial mixed-effect AR(1) model
## where random effects is from the log-normal distribution

re.logn.ar1 <- fitParaAR1(formula=CEL~1,data=dt2,ID=ID,
          Vcode=Vcode,
          p.ini=c(loga,logtheta,logitd,b0), ## log(a), log(theta), logit(d), b0
              RE="N",IPRT=TRUE)

Psum <- index.batch(olmeNB=re.logn.ar1,data=dt2,ID=ID,Vcode=Vcode,
                    labelnp=new,qfun="sum", IPRT=TRUE,i.se=FALSE,C=TRUE,i.tol=1E-3)
re.logn.ar1$Psum <- Psum
```

```
## 3) Fit the negative binomial independent model
## where random effects are from the gamma distribution
re.gamma.ind <- fitParaIND(formula=CEL~1,data=dt2,ID=ID,
                           RE="G",
             p.ini=c(loga,logtheta,b0),
            IPRT=TRUE)

Psum <- index.batch(olmeNB=re.gamma.ind,data=dt2,ID=ID,
                    labelnp=new,qfun="sum", IPRT=TRUE,i.se=TRUE)



## 4) Fit the negative binomial independent model
## where random effects are from the lognormal distribution
re.logn.ind <- fitParaIND(formula=CEL~1,data=dt2,ID=ID,
                          RE="N",
           p.ini=c(loga,logtheta,b0),
           IPRT=TRUE)

Psum <- index.batch(olmeNB=re.logn.ind, data=dt2,ID=ID,
                   labelnp=new,qfun="sum", IPRT=TRUE,i.se=TRUE)


## 5) Fit the semi-parametric negative binomial AR(1) model

logvarG <- -0.5
re.semi.ar1 <- fitSemiAR1(formula=CEL~1,data=dt2,ID=ID,
                          p.ini=c(loga, logvarG, logitd,b0),Vcode=Vcode)
Psum <- index.batch(olmeNB=re.semi.ar1,data=dt2,ID=ID, Vcode=Vcode,
             labelnp=new,qfun="sum", IPRT=TRUE,i.se=FALSE)


## 6) Fit the semi-parametric negative binomial independent model
re.semi.ind <- fitSemiIND(formula=CEL~1,data=dt2,ID=ID, p.ini=c(loga, logvarG, b0))
Psum <- index.batch(olmeNB=re.semi.ind,data=dt2,ID=ID,
                   labelnp=new, qfun="sum", IPRT=TRUE,i.se=FALSE)


## ======================================================================== ##
## == Which model performed the best in terms of the estimation of beta0 == ##
## ======================================================================== ##

getpoints <- function(y,estb0,sdb0=NULL,crit=qnorm(0.975))
{
points(estb0,y,col="blue",pch=16)
if (!is.null(sdb0))
{
points(c(estb0-crit*sdb0,estb0+crit*sdb0),rep(y,2),col="red",type="l")
}
}
ordermethod <- c("gamma.ar1","logn.ar1","gamma.ind","logn.ind","semi.ar1","semi.ind")

estb0s <- c(
```

```
re.gamma.ar1$est[4,1],
re.logn.ar1$est[4,1],
re.gamma.ind$est[3,1],
re.logn.ind$est[3,1],
re.semi.ar1$est[4],
re.semi.ind$est[3]
)

## The true beta0 is:
b0
c <- 1.1
plot(0,0,type="n",xlim=c(min(estb0s)-0.5,max(estb0s)*c),ylim=c(0,7),yaxt="n",
main="Simulated from the AR(1) model \n with random effect ~ gamma")

legend("topright",
legend=ordermethod)
abline(v=b0,lty=3)

## 1) gamma.ar1
sdb0 <- re.gamma.ar1$est[4,2]
getpoints(6,estb0s[1],sdb0)

## 2)logn.ar1
sdb0 <- re.logn.ar1$est[4,2]
getpoints(5,estb0s[2],sdb0)

## 3) gamma.ind
sdb0 <- re.gamma.ind$est[3,2]
getpoints(4,estb0s[3],sdb0)

## 4) logn.ind
sdb0 <- re.logn.ind$est[3,2]
getpoints(3,estb0s[4],sdb0)

## 5) semi.ar1
getpoints(2,estb0s[5])

## 6) semi.ind
getpoints(1,estb0s[6])



## End(Not run)
```

`fitParaIND` *Performs the maximum likelihood estimation for the negative binomial mixed-effect independent model*

Description

This function fits the parametric negative binomial mixed-effect independent model in the formulation described Zhao et al (2013). The conditional distribution of response count given random effect is modelled by Negative Binomial as described in description of lmeNB. The conditional dependence among the response counts of a subject is assumed independent. The random effects are modelled with either gamma or log-normal distributions. See descriptions of lmeNB.

Usage

```
fitParaIND(formula, data, ID, p.ini = NULL, IPRT = FALSE, RE = "G",
           i.tol = 1e-75, o.tol = 0.001, COV = TRUE)
```

Arguments

formula	See lmeNB.
data	See lmeNB.
ID	See lmeNB.
p.ini	The initial values of the parameters $\log(\alpha), \log(\theta), \beta_0, \beta_1$ NULL is accepted.
IPRT	See lmeNB.
RE	The distribution of random effects G_i. If RE="G" then the random effects are assumed to be from the gamma distribution. If RE="N" then they are assumed to be form the log-normal distribution.
i.tol	See lmeNB.
o.tol	See lmeNB.
COV	Internal use only.

Details

fitParaIND calls optim to minimize the negative log-likelihood of the negative binomial model with respect to the model parameters: $\log(\alpha), \log(\theta), \beta_0, \beta_1, \cdots$. The Nelder-Mead algorithm is employed. The log-likelihood is obtained by marginalizing out the random effects. The numerical integration is carried out using adaptive quadrature.

The missing count responses, if assumed to be missing at random, can be igored.Other types of missing data are currently not accepted.

All the computations are done in R.

Value

call	See lmeNB.
opt	See lmeNB.
nlk	See lmeNB.
V	See lmeNB.
est	See lmeNB.
AR	FALSE

Author(s)

Zhao, Y. and Kondo, Y.

References

Detection of unusual increases in MRI lesion counts in individual multiple sclerosis patients. (2013) Zhao, Y., Li, D.K.B., Petkau, A.J., Riddehough, A., Traboulsee, A., Journal of the American Statistical Association.

See Also

The wrapper function for all the negative binomial mixed effect regression: lmeNB.

The functions to fit the other negative binomial mixed effect models:

fitParaAR1, fitSemiIND, fitSemiAR1,

The subroutines of index.batch to compute the conditional probability index: jCP.ar1, CP1.ar1, MCCP.ar1, CP.ar1.se, CP.se, jCP,

The functions to generate simulated datasets: rNBME.R.

Examples

```
## Not run:

## ========================================================================== ##
## generate a simulated dataset from the negative binomial mixed-effect independent model:
## Y_ij | G_i = g_i ~ NB(r_ij,p_i) where r_ij = exp(X^T beta)/a , p_i =1/(a*g_i+1)
## with G_i ~ Gamma(scale=th,shape=1/th)
set.seed(1)
sn <- 5 ## the number of repeated measures of each patient
n <- 80 ## the number of patients
loga <- - 0.5
a <- exp(loga) ## the parameter for the failure probability of the negative binomial distribution
logtheta <- 1.3
th <- exp(logtheta) ## the parameter for the gamma distribution of g_i

## No difference between the means of groups
## The model only has an intercept term beta0 = 0.5
b0 <- 0.5
u1 <- rep(exp(b0),sn) ## the mean vector for group 1 at time point 1,...,sn
u2 <- rep(exp(b0),sn) ## the mean vector for group 2 at time point 1,...,sn


DT0 <- rNBME.R(gdist="G", n=n, a=a, th=th, u1=u1, u2=u2, sn=sn)
ID <- DT0$id
Vcode <- rep(-1:(sn-2),n) # scan number -1, 0, 1, 2, 3
new <- Vcode > 0
dt1 <- data.frame(CEL=DT0$y)
logitd <- -0.5
```

```
## ============================================================================== ##


## [1]: Fit the negative binomial independent model
## where the random effects are from the gamma distribution. This is the true model.

re.gamma.ind <- fitParaIND(formula=CEL~1,data=dt1,ID=ID,RE="G",
                  p.ini=c(loga,logtheta,b0),IPRT=TRUE)
## compute the estimates of the conditional probabilities
## with sum of the new repeated measure as a summary statistics
Psum <- index.batch(olmeNB=re.gamma.ind, ID=ID,data=dt1,
                    labelnp=new,qfun="sum", IPRT=TRUE)

## compute the estimates of the conditional probabilities
## with max of the new repeated measure as a summary statistics
Pmax <- index.batch(olmeNB=re.gamma.ind, ID=ID,data=dt1,
                   labelnp=new,qfun="max", IPRT=TRUE)

## Which patient's estimated probabilities
## based on the sum and max statistics disagrees the most?
( IDBigDif <- which(rank(abs(Pmax$condProbSummary[,1]-Psum$condProbSummary[,1]))==80) )
## Show the patient's CEL counts
dt1$CEL[ID==IDBigDif]
## Show the estimated conditional probabilities based on the sum summary statistics
Psum$condProbSummary[IDBigDif,]
## Show the estimated conditional probabilities based on the max summary statistics
Pmax$condProbSummary[IDBigDif,]



## [2]: Fit the negative binomial independent model
## where the random effects are from the lognormal distribution.
re.logn.ind <- fitParaIND(formula=CEL~1,data=dt1,ID=ID,
                          RE="N",
          p.ini=c(loga,logtheta,b0),
          IPRT=TRUE)

Psum <- index.batch(olmeNB=re.logn.ind, ID=ID,data=dt1,C=TRUE,
                   labelnp=new,qfun="sum", IPRT=TRUE)

## [3]: Fit the semi-parametric negative binomial independent model


re.semi.ind <- fitSemiIND(formula=CEL~1,data=dt1,ID=ID)

Psum <- index.batch(olmeNB=re.semi.ind,ID=ID,data=dt1, i.se = FALSE,
                   labelnp=new, qfun="sum", IPRT=TRUE)



## [4]: Fit the negative binomial mixed-effect AR(1) model
## where random effects are from the gamma distribution
```

```
re.gamma.ar1 <- fitParaAR1(formula=CEL~1,data=dt1,ID=ID,
           p.ini=c(loga,logtheta,logitd,b0),
           RE="G", Vcode=Vcode,
           IPRT=TRUE)

Psum <- index.batch(olmeNB=re.gamma.ar1, ID=ID,data=dt1, labelnp=new,Vcode=Vcode,
                    qfun="sum", IPRT=TRUE,i.se=FALSE) ## i.se=TRUE requires more time...




## ========================================================================== ##
## == Which model performed the best in terms of the estimation of beta0 == ##
## ========================================================================== ##

getpoints <- function(y,estb0,sdb0=NULL,crit=qnorm(0.975))
{
points(estb0,y,col="blue",pch=16)
if (!is.null(sdb0))
{
points(c(estb0-crit*sdb0,estb0+crit*sdb0),rep(y,2),col="red",type="l")
}
}

ordermethod <- c("gamma.ind","logn.ind","semi.ind","gamma.ar1")

estb0s <- c(
re.gamma.ind$est[3,1],
re.logn.ind$est[3,1],
re.semi.ind$est[3],
re.gamma.ar1$est[4,1]
)

## The true beta0 is:
b0
c <- 1.1
plot(0,0,type="n",xlim=c(min(estb0s)-0.5,max(estb0s)*c),
ylim=c(0,5),yaxt="n",
main="Simulated from the independent model \n with random effect ~ gamma")

legend("topright",
col="red",
legend=ordermethod)
abline(v=b0,lty=3)

## [1] gamma.ind
sdb0 <- re.gamma.ind$est[3,2]
getpoints(4,estb0s[1],sdb0)

## [2] logn.ind
sdb0 <- re.logn.ind$est[3,2]
```

```
getpoints(3,estb0s[2],sdb0)

## [3] semi.ind
getpoints(2,estb0s[3])

## [4] gamma.ar1
sdb0 <- re.gamma.ar1$est[4,2]
getpoints(1,estb0s[4],sdb0)


## End(Not run)
```

`fitSemiAR1` *Fit the semi-parametric negative binomial mixed-effect AR(1) model.*

Description

This function fits the semi-parametric negative binomial mixed-effect AR(1) model in the formulation described Zhao et al (2013). The conditional distribution of response counts given random effect is modelled by Negative Binomial as described in description of `lmeNB`. The conditional dependence among the response counts of a subject is modeled with AR(1) structure. The semi-parametric procedure is employed for random effects. See descriptions of `lmeNB`.

Usage

```
fitSemiAR1(formula, data, ID, Vcode,p.ini = NULL, IPRT = TRUE, deps = 0.001, maxit=100)
```

Arguments

`formula`	See `lmeNB`.
`data`	See `lmeNB`.
`ID`	See `lmeNB`.
`Vcode`	See `lmeNB`.
`p.ini`	See `fitParaAR1`.
`IPRT`	See `lmeNB`.
`deps`	See `lmeNB`.
`maxit`	See `lmeNB`.

Details

The algorithm repeats the following four steps until a stoping criterion is satisfied:

Step 1) Estimate the coefficients of covariates by the method of weighted least squares.

Step 2) Approximate the distribution of the random effect G_i by γ_i.

Step 3) Estimate α and δ using the psudo-profile likelihood. This step calls `optim` to minimize the negative psudo log-likelihood with respect to $\log(\alpha)$) and logit(δ). The numerical integration is carried out using adaptive quadrature. When missing visits are present, the likelihood is approximated (See Zhao et al. 2013 for details).

Step 4) Estimate $Var(G_i)$ by the medhod of moment and update the weights.

All the computations are done in `R`.

Author(s)

Zhao, Y. and Kondo, Y.

References

Detection of unusual increases in MRI lesion counts in individual multiple sclerosis patients. (2013) Zhao, Y., Li, D.K.B., Petkau, A.J., Riddehough, A., Traboulsee, A., Journal of the American Statistical Association.

See Also

The main function to fit the Negative Binomial mixed-effect model: `lmeNB`,

The functions to fit the other models: `fitParaIND`, `fitParaAR1`, `fitSemiIND`,

The subroutines of `index.batch` to compute the conditional probability index: `jCP.ar1`, `CP1.ar1`, `MCCP.ar1`, `CP.ar1.se`, `CP.se`, `jCP`,

The functions to generate simulated datasets: `rNBME.R`.

Examples

```
## Not run:
## ========================================================================== ##
## generate a data based on the semi-parametric negative binomial
## mixed-effect AR(1) model.
## Under this model, the response counts follows the negative binomial:
## Y_ij | G_i = g_i ~ NB(r_ij,p_i) where r_ij = exp(X^T beta)/a , p_i =1/(a*g_i+1)
## G_i is from unknown distribution.
## For simulation purpose, we generate the sample of gi from
## the mixture of three gamma distribuions.

## The adjacent repeated measures of the same subjects are correlated
## with correlation structure:
## cov(Y_ij,Y_ij'|G_i=g_i)=d^{j-j'} E(Y_ij')*(a*g_i^2+g_i)

# log(a) = -0.5, log(th)=1.3, logit(delta) = -0.2
# b0 =  0.5, no covariates;
```

```
loga <- -0.5
logtheta <- 1.3
logitd <- -0.2
b0 <- 0.5
# 80 subjects each with 5 scans
n <- 80
sn <- 5

## generate a sample of size B from the mixture of three gamma distribution:
p1 <- 0.5
p2 <- 0.3
B <- 1000
sampledG<- c(
rgamma(n=p1*B,scale=1,shape=10),
rgamma(n=p2*B,scale=3,shape=5),
rgamma(n=(1-p1-p2)*B,scale=5,shape=5)
)


## mean is set to 1;
sampledG <- sampledG/mean(sampledG)
logvarG <- log(var(sampledG))
## hist(sampledG)

DT4 <-  rNBME.R(gdist = "NoN",
                n = n, ##  the total number of subjectss
       sn = sn,
                u1 = rep(exp(b0),sn),
       u2 = rep(exp(b0),sn),
       a = exp(loga),
       d = exp(logitd)/(1+exp(logitd)),
       othrp = sampledG
      )
Vcode<-rep(-1:(sn-2),n) # scan number -1, 0, 1, 2, 3
ID <- DT4$id
new <- Vcode > 0
dt4<-data.frame(CEL=DT4$y)
## ============================================================================== ##

## [1] Fit the negative binomial mixed-effect AR(1) model
## where random effects is from the gamma distribution


re.gamma.ar1 <- fitParaAR1(formula=CEL~1,data=dt4,ID=ID,
         Vcode=Vcode,
          p.ini=c(loga,logtheta,logitd,b0),
          ## log(a), log(theta), logit(d), b0
          RE="G",
          IPRT=TRUE)

Psum<-index.batch(olmeNB=re.gamma.ar1, data=dt4,ID=ID,Vcode=Vcode,
          labelnp=new,qfun="sum", IPRT=TRUE,i.se=FALSE)
```

```
## [2] Fit the negative binomial mixed-effect AR(1) model
## where random effects is from the log-normal distribution


re.logn.ar1<-fitParaAR1(formula=CEL~1,data=dt4,ID=ID,
        Vcode=Vcode,
        p.ini=c(loga,logtheta,logitd,b0),
        ## log(a), log(theta), logit(d), b0
        RE="N", IPRT=TRUE)

## Requires some time
Psum<-index.batch(olmeNB=re.logn.ar1,data=dt4,ID=ID,Vcode=Vcode,
          labelnp=new,qfun="sum", IPRT=TRUE)



## [3] Fit the negative binomial independent model
## where random effects is from the lognormal distribution
re.logn.ind<-fitParaIND(formula=CEL~1,data=dt4,ID=ID,
                        RE="N",
        p.ini=c(loga,logtheta,b0),
        IPRT=TRUE)

Psum <- index.batch(olmeNB=re.logn.ind,data=dt4,ID=ID,
                    labelnp=new,qfun="sum", IPRT=TRUE)


## [4] Fit the semi-parametric negative binomial AR(1) model
## This model is closest to the true model

logvarG <- log(var(sampledG))

re.semi.ar1 <- fitSemiAR1(formula=CEL~1,data=dt4,ID=ID,
                          p.ini=c(loga, logvarG, logitd,b0),Vcode=Vcode)

## compute the estimates of the conditional probabilities
## with sum of the new repeated measure as a summary statistics
Psum <- index.batch(olmeNB=re.semi.ar1, labelnp=new,data=dt4,ID=ID,Vcode=Vcode,
                    qfun="sum", IPRT=TRUE,i.se=TRUE)

## compute the estimates of the conditional probabilities
## with max of the new repeated measure as a summary statistics
Pmax <- index.batch(olmeNB=re.semi.ar1, labelnp=new,qfun="max",data=dt4,ID=ID,Vcode=Vcode,
                    IPRT=TRUE,i.se=TRUE)

## Which patient's estimated probabilities
## based on the sum and max statistics disagrees the most?
( IDBigDif <- which(rank(abs(Pmax$condProbSummary[,1]-Psum$condProbSummary[,1]))==80) )
## Show the patient's CEL counts
dt4$CEL[ID==IDBigDif]
## Show the estimated conditional probabilities based on the sum summary statistics
```

```
Psum$condProbSummary[IDBigDif,1]
## Show the estimated conditional probabilities based on the max summary statistics
Pmax$condProbSummary[IDBigDif,1]


## [5] Fit the semi-parametric negative binomial independent model


re.semi.ind <- fitSemiIND(formula=CEL~1,data=dt4,ID=ID, p.ini=c(loga, logvarG, b0))
Psum <- index.batch(olmeNB=re.semi.ind, labelnp=new,
                    data=dt4,ID=ID, qfun="sum", IPRT=TRUE,i.se=TRUE)



## ========================================================================== ##
## == Which model performed the best in terms of the estimation of beta0 == ##
## ========================================================================== ##

getpoints <- function(y,estb0,sdb0=NULL,crit=qnorm(0.975))
{
points(estb0,y,col="blue",pch=16)
if (!is.null(sdb0))
{
points(c(estb0-crit*sdb0,estb0+crit*sdb0),rep(y,2),col="red",type="l")
}
}
ordermethod <- c("gamma.ar1","logn.ar1","logn.ind","semi.ar1","semi.ind")

estb0s <- c(
re.gamma.ar1$est[4,1],
re.logn.ar1$est[4,1],
re.logn.ind$est[3,1],
re.semi.ar1$est[4],
re.semi.ind$est[3]
)

## The true beta0 is:
b0
c <- 1.1
plot(0,0,type="n",xlim=c(min(estb0s)-0.5,max(estb0s)*c),ylim=c(0,7),yaxt="n",
main <- "Simulated from the AR(1) model \n with random effect ~ a semi-parametric distribution")

legend("topright",
legend=ordermethod)
abline(v=b0,lty=3)

## [1] gamma.ar1
sdb0 <- re.gamma.ar1$est[4,2]
getpoints(6,estb0s[1],sdb0)

## [2] logn.ar1
sdb0 <- re.logn.ar1$est[4,2]
getpoints(5,estb0s[2],sdb0)
```

```
## [3] logn.ind
sdb0 <- re.logn.ind$est[3,2]
getpoints(4,estb0s[3],sdb0)

## [4] semi.ar1
getpoints(3,estb0s[4])

## [5] semi.ind
getpoints(2,estb0s[5])


## End(Not run)
```

`fitSemiIND` *Fit the semi-parametric negative binomial mixed-effect independent model.*

Description

This function fits the semi-parametric negative binomial mixed-effect independent model to repeated count responses (Zhao et al. 2013). The conditional distribution of response count given random effect is modelled by Negative Binomial as described in description of `lmeNB`. The conditional dependence among the response counts of a subject is assumed independent. The semiparametric procedure is employed for random effects. See descriptions of `lmeNB`.

Usage

```
fitSemiIND(formula,data,ID, p.ini = NULL, IPRT = TRUE, deps = 1e-04,
   maxit = 100, u.low = 0)
```

Arguments

`formula`	See `lmeNB`.
`data`	See `lmeNB`.
`ID`	See `lmeNB`.
`p.ini`	A vector of length 3 + # covariates, containing the initial values for the parameters $(\log(\alpha), \log(Var(G_i)), \beta_0, \beta_1, ...)$. `NULL` is accepted.
`IPRT`	See `lmeNB`.
`deps`	See `lmeNB`.
`maxit`	See `lmeNB`.
`u.low`	See `lmeNB`.

Details

The algorithm repeats the following four steps until a stoping criterion is satisfied:

Step 1) Given α, Estimate the coefficients of covariates by the method of generalized Least Squares.

That is, this step solves for: $argmin_{\boldsymbol{\beta}} \sum_{i=1}^{N} (\boldsymbol{Y}_i - E(\boldsymbol{Y}_i; \boldsymbol{\beta}))^T \boldsymbol{W}_i (\boldsymbol{Y}_i - E(\boldsymbol{Y}_i; \boldsymbol{\beta}))$ where the weight matrix for each patient $\boldsymbol{W}_i$ is selected to $Var(\boldsymbol{Y}_i)^{-1}$ (which is a function of α) if it exists, else it is set to be an identity matrix.

Step 2) Approximate the distribution of the random effect G_i by γ.

Step 3) Estimate α by minimizing the negative psudo-profile likelihood. The numerical minimization is carried out using `optimize` and the numerical integration is carried out using adaptive quadrature.

Step 4) Estimate $Var(G_i)$ by the medhod of moment and update the weights.

All the computations are done in `R`.

Value

`opt`	See `lmeNB`.
`diffPara`	The largest absolute difference of parameter vectors between the current and previous iterations.
`V`	`NULL`
`est`	See `lmeNB`.
`gtb`	The relative frequency table of G_i, $(i = 1, \cdots, N)$. `gh1` (the second column) contains the unique values in ascending order and `ghw` (the first column) contains the corresponding relative frequencies.
`counter`	The number of iterations before the algorithm was terminated
`gi`	A vector of length N, containing the approximated random effect $G_i, i = 1, \cdots, N$.
`RE`	`"NoN"`, denoting that the fitted model is a semi-parametric mixed-effect model.
`AR`	`FALSE`
`paraAll`	Record estimated parameters at every iteration.

Author(s)

Zhao, Y. and Kondo, Y.

References

Detection of unusual increases in MRI lesion counts in individual multiple sclerosis patients. (2013) Zhao, Y., Li, D.K.B., Petkau, A.J., Riddehough, A., Traboulsee, A., Journal of the American Statistical Association.

See Also

The main function to fit the Negative Binomial mixed-effect model: lmeNB,

The functions to fit the other models: fitParaIND, fitParaAR1,

fitSemiAR1,

The subroutines of index.batch to compute the conditional probability index: jCP.ar1, CP1.ar1, MCCP.ar1, CP.ar1.se, CP.se, jCP,

The functions to generate simulated datasets: rNBME.R.

Examples

```
## Not run:
## generate a simulated dataset from the negative binomial
## mixed-effect independent model:
## Y_ij | G_i = g_i ~ NB(r_ij,p_i) where r_ij = exp(X^T beta)/a , p_i =1/(a*g_i+1)
## with G_i is from unknown distribution
## For the simulation purpose, G_i's are from the mixture of
## the gamma and the log-normal distributions.


sn <- 5 ## the number of repeated measures of each subject
n <- 80 ## the number of subjects
logtheta <- 1.3
th <- exp(logtheta) ## the parameter for the gamma distribution of g_i
loga <- -0.5
## the parameter for the failure probability of the negative binomial distribution
a <- exp(loga)
b0 <- 0.5
u1 <- rep(exp(b0),sn) ## the mean vector for group 1 at time point 1,...,sn
u2 <- rep(exp(b0),sn) ## the mean vector for group 2 at time point 1,...,sn

DT3 <- rNBME.R(gdist="GN", n=n, a=a, th=th, u1=u1, u2=u2, sn=sn,
       othrp=list(p.mx=0.1,u.n=3,s.n=1,sh.mx = NA) ## 0  < p.mx < 1
       )

ID <- DT3$id
dt3 <- data.frame(CEL=DT3$y)

Vcode <- rep(-1:(sn-2),n) # scan number -1, 0, 1, 2, 3
new <-Vcode>0         # new scans: 1,2,3


## 1) Fit the negative binomial mixed-effect AR(1) model
## where random effects is from the gamma distribution

logitd <- -0.2
re.gamma.ar1 <- fitParaAR1(formula=CEL~1,data=dt3,ID=ID,
          Vcode=Vcode,
          p.ini=c(loga,logtheta,logitd,b0),
          ## log(a), log(theta), logit(d), b0
          RE="G",
```

```
          IPRT=TRUE)


## compute the estimates of the conditional probabilities
## with sum of the new repeated measure as a summary statistics
## i.se=FALSE,C=TRUE options for speed up!
Psum <- index.batch(olmeNB=re.gamma.ar1,data=dt3,ID=ID, Vcode=Vcode,
                    labelnp=new,qfun="sum", IPRT=TRUE,i.se=FALSE,C=TRUE,i.tol=1E-3)



## 2) Fit the negative binomial mixed-effect AR(1) model
## where random effects is from the log-normal distribution


re.logn.ar1 <- fitParaAR1(formula=CEL~1,data=dt3,ID=ID,
          Vcode=Vcode, RE="N", IPRT=TRUE)

## REQUIRES SOME TIME..
Psum <- index.batch(olmeNB=re.logn.ar1, data=dt3,ID=ID,Vcode=Vcode,
            labelnp=new,qfun="sum", IPRT=TRUE,i.se=FALSE,C=TRUE,i.tol=1E-3)



## 3) Fit the negative binomial independent model
## where random effects is from the gamma distribution
re.gamma.ind <- fitParaIND(formula=CEL~1,data=dt3,ID=ID,
                           RE="G",IPRT=TRUE)

Psum <- index.batch(olmeNB=re.gamma.ind, data=dt3,ID=ID,Vcode=Vcode,
                    labelnp=new,qfun="sum", IPRT=TRUE,i.se=TRUE)



## 4) Fit the negative binomial independent model
## where random effects is from the lognormal distribution
re.logn.ind <- fitParaIND(formula=CEL~1,data=dt3,ID=ID,
                          RE="N",
          p.ini=c(loga,logtheta,b0),
          IPRT=TRUE)

Psum <- index.batch(olmeNB=re.logn.ind,data=dt3,ID=ID,labelnp=new,qfun="sum", IPRT=TRUE)

## 5) Fit the semi-parametric negative binomial AR(1) model

logvarG <- -0.4

re.semi.ar1 <- fitSemiAR1(formula=CEL~1,data=dt3,ID=ID,Vcode=Vcode)
Psum <- index.batch(olmeNB=re.semi.ar1,data=dt3,ID=ID,Vcode=Vcode,
                    labelnp=new,qfun="sum", IPRT=TRUE,MC=TRUE,i.se=FALSE)


## 6) Fit the semi-parametric negative binomial independent model
```

```
## This is closest to the true model
re.semi.ind <- fitSemiIND(formula=CEL~1,data=dt3,ID=ID, p.ini=c(loga, logvarG, b0))

## compute the estimates of the conditional probabilities
## with sum of the new repeated measure as a summary statistics
Psum <- index.batch(olmeNB=re.semi.ind,data=dt3,ID=ID, labelnp=new,
                    qfun="sum", IPRT=TRUE,i.se=FALSE)
## compute the estimates of the conditional probabilities
## with max of the new repeated measure as a summary statistics
Pmax <- index.batch(olmeNB=re.semi.ind, data=dt3,ID=ID,labelnp=new, qfun="max",
                    IPRT=TRUE,i.se=FALSE)


## Which patient's estimated probabilities based on the sum and max
## statistics disagrees the most?
( IDBigDif <- which(rank(abs(Pmax$condProbSummary[,1]-Psum$condProbSummary[,1]))==80) )
## Show the patient's CEL counts
dt3$CEL[ID==IDBigDif]
## Show the estimated conditional probabilities based on the sum summary statistics
Psum$condProbSummary[IDBigDif,]
## Show the estimated conditional probabilities based on the max summary statistics
Pmax$condProbSummary[IDBigDif,]


## ========================================================================== ##
## == Which model performed the best in terms of the estimation of beta0 == ##
## ========================================================================== ##

getpoints <- function(y,estb0,sdb0=NULL,crit=qnorm(0.975))
{
points(estb0,y,col="blue",pch=16)
if (!is.null(sdb0))
{
points(c(estb0-crit*sdb0,estb0+crit*sdb0),rep(y,2),col="red",type="l")
}
}
ordermethod <- c("gamma.ar1","logn.ar1","gamma.ind","logn.ind","semi.ar1","semi.ind")

estb0s <- c(
re.gamma.ar1$est[4,1],
re.logn.ar1$est[4,1],
re.gamma.ind$est[3,1],
re.logn.ind$est[3,1],
re.semi.ar1$est[4],
re.semi.ind$est[3]
)

## The true beta0 is:
b0
c <- 1.1
plot(0,0,type="n",xlim=c(min(estb0s)-0.5,max(estb0s)*c),ylim=c(0,7),yaxt="n",
main="Simulated from the independent model \n with random effect ~ mixture of normal and gamma")
```

```
legend("topright",
legend=ordermethod)
abline(v=b0,lty=3)

## 1) gamma.ar1
sdb0 <- re.gamma.ar1$est[4,2]
getpoints(6,estb0s[1],sdb0)

## 2)logn.ar1
sdb0 <- re.logn.ar1$est[4,2]
getpoints(5,estb0s[2],sdb0)

## 3) gamma.ind
sdb0 <- re.gamma.ind$est[3,2]
getpoints(4,estb0s[3],sdb0)

## 4) logn.ind
sdb0 <- re.logn.ind$est[3,2]
getpoints(3,estb0s[4],sdb0)

## 5) semi.ar1
getpoints(2,estb0s[5])

## 6) semi.ind
getpoints(1,estb0s[6])

## End(Not run)
```

`index.batch` *The main function to compute the point estimates and 95% confidence intervals (for a parametric model) of the conditional probabilities* $Pr(q(\boldsymbol{Y}\_i, new) \geq q(\boldsymbol{y}\_i, new)|\boldsymbol{Y}\_i, pre = \boldsymbol{y}\_i, pre)$ *for multiple subjects.*

Description

Let m_i be the number of pre-measurements and n_i be the total number of repeated measures. Then the repeated measure of a subject can be divided into a pre-measurement set and a new measurement set as $\boldsymbol{Y}_i = (\boldsymbol{Y}_{i,pre}, \boldsymbol{Y}_{i,new})$, where $\boldsymbol{Y}_{i,pre} = (Y_{i,1}, \cdots, Y_{i,m_i})$ and $\boldsymbol{Y}_{i,new} = (Y_{i,m_i+1}, \cdots, Y_{i,n_i})$. Given an output of `fitParaIND`, `fitParaAR1`, `fitSemiIND`, `fitSemiAR1` or `lmeNB`, this function computes the probability of observing the response counts as large as those new observations of subject i, $\boldsymbol{y}_{i,new}$ conditional on the subject's previous observations $\boldsymbol{y}_{i,pre}$ for subject i. That is, this function returns a point estimate and its asymptotic 95% confidence interval (for a parametric model) of the conditional probability for each subject:

$Pr(q(\boldsymbol{Y}_{i,new}) \geq q(\boldsymbol{y}_{i,new})|\boldsymbol{Y}_{i,pre} = \boldsymbol{y}_{i,pre})$.

When the semiparametric approach is employed, the standard error and 95% confidence intervals are computed using bootstrap samples. A scalar statistic to summarize the new response counts can be either the total count, $q(\boldsymbol{Y}_{i,new}) = \sum_{j=m_i+1}^{n_i} Y_{ij}$, or the maximum, $q(\boldsymbol{Y}_{i,new}) = \max\{Y_{ij}; j = m_i + 1, \cdots, n_i\}$.

See Zhao et al.(2013), for more details.

Usage

```
index.batch(data,       labelnp,    ID,  Vcode = NULL, olmeNB = NULL, subset = NULL,
          qfun = "sum", IPRT = TRUE, i.se = TRUE, MC = FALSE, C = FALSE,i.tol=1E-75)
```

Arguments

data	See lmeNB. This dataset does not have to be the same as the one used in the computations of negative binomial mixed effect regression (fitParaIND, fitParaAR1, fitSemiIND, fitSemiAR1 or lmeNB).
labelnp	A vector of length the total number of repeated measures (=$\sum_{i=1}^{N} n_i$), indicating new measures by TRUE and pre-measures by FALSE. For examples, suppose there are three subjects of interest. The first subject has a $n_1 = 7$ repeated measures and the last *3* measures are new. The second and the third subjects both have $n_2 = n_3 = 5$ repeated measures and the last 2 measures are new. In this scenario, labelnp=c(rep(FALSE,4), rep(TRUE,3), rep(FALSE,3), rep(TRUE,2) ,rep(FALSE,3) ,rep(TRUE,2)).
ID	See lmeNB. The length of ID must be the same as nrow(data).
Vcode	Necessary only if the olmeNB is an output of AR(1) models. See lmeNB.
olmeNB	Output of fitParaIND,fitParaAR1,fitSemiIND,fitSemiAR1 or lmeNB.
subset	An optional expression indicating the subset of the subjects of that the index should be computed.
qfun	If qfun="sum", a scalar statistic to summarize the new response counts is the total count. If qfun="max", a scalar statistic to summarize the new response counts is the maximum.
IPRT	print control.
i.se	If i.se=TRUE then the standard errors of the estimator of the conditional probability are returned for the output of fitParaIND or fitParaAR1. The semiparametric approach, fitSemiIND or fitSemiAR1, do not return the standard errors.
MC	Necessary when olmeNB if the AR(1) model outputs. See CP.ar1.se.
C	See lmeNB. C=TRUE option could make computations of CPI faster for some patients.
i.tol	See lmeNB.

Details

The standard error of the point estimate on the logit scale is constructed using the delta method for the parametric model, where distributional assumption was made for random effects.

Value

The N by 4 (3, if hide the SE) numeric matrix, containing the point estimate of the conditional probability, and the lower and the upper bounds of the 95

Author(s)

Zhao, Y. and Kondo, Y.

References

Detection of unusual increases in MRI lesion counts in individual multiple sclerosis patients. (2013) Zhao, Y., Li, D.K.B., Petkau, A.J., Riddehough, A., Traboulsee, A., Journal of the American Statistical Association.

See Also

The main function to fit the Negative Binomial mixed-effect model: `lmeNB`,

The internal functions of `lmeNB` for fitting relevant models: `fitParaIND`, `fitParaAR1`, `fitSemiIND`, `fitSemiAR1`,

The subroutines of `index.batch`: `jCP.ar1`, `CP1.ar1`, `MCCP.ar1`, `CP.ar1.se`, `CP.se`, `jCP`,

The functions to generate simulated datasets: `rNBME.R`.

Examples

```
## See the examples in help files of
## fitParaIND, fitAR1IND, fitSemiIND, fitSemiAR1 and rNBME.R
```

`lmeNB` *Performs the maximum likelihood estimation for the negative binomial mixed-effect model. This function is a wrapper for* `fitParaIND`, `fitParaAR1`, `fitSemiIND` *and* `fitSemiAR1`.

Description

Let Y_{ij} be the response count at jth repeated measure from ith subject. The negative binomial mixed-effect independent model assumes that given the random effect $G_i = g_i$, the count response Y_{ij} follows the negative binomial distribution:

$Y_{ij}|G_i = g_i \sim NB(r_{ij}, p_i)$,

where p_i, the failure probability of subject i at each time point j is parametrized as $p_i = \frac{1}{g_i\alpha+1}$ and $\alpha > 0$. The model assumes $E(G_i) = 1$ so that $E(Y_{ij}|G_i = g_i) = r_{ij}g_i\alpha$ and $E(Y_{ij}) = r_{ij}g_i$. This assumption allows the interpretation of the latent random variable G_i as the subject i's activity rate relative to the overall cohort. The marginal mean $\mu_{ij} = E(Y_{ij})$ is modeled with fixed effect coefficients, $\boldsymbol{\beta}$: $\mu_{ij} = exp(\boldsymbol{X}_{ij}^T\boldsymbol{\beta})$. Furthermore, let $Var(G_i) = \theta$, then $Var(Y_{ij}) = \mu_{ij}^2\theta + \mu_{ij}(1 + (\theta + 1)\alpha)$.

Regarding the dependence structures of Y_{ij} and $Y_{ij'}$ conditional on the random effect G_i, we consider two models, namely independent and AR(1) models.

[1]: Independent model

Y_{ij} and $Y_{ij'}$ are independent conditionally on G_i. This assumption leads to

$Cov(Y_{ij}, Y_{ij'}|G_i = g_i) = 0$ and $Cov(Y_{ij}, Y_{ij'}) = \mu_{ij}\mu_{ij'}\theta$

[2]: AR(1) model

Autoregressive (1) structures for Y_{ij} and Y_ij' conditionally on G_i. That is given the random effect $G_i = g_i$, Y_{ij} depends on $Y_{i(j-1)}$ through the beta binomial thinning and is conditionally independent on $Y_{ij'}$ given $Y_{i(j-1)}$ for all $j' < j - 1$. The beta binomial thinning operator depends on a parameter δ and models to have $Cov(Y_{ij}, Y_{ij'}|G_i = g_i) = \delta^{j-j'}\mu_{ij'}(\alpha g_i^2 + g_i)$. This means that δ measures the strength of the positive AR(1) association between repeated measures of a subject: the larger δ, the stronger the positive correlations between the repeated measures of the same subject are.

Regarding the random effect G_i distribution, `lmeNB` allows three models, namely log-normal, gamma and semiparametric models. (All models assume $E(G_i) = 1$ and $Var(G_i) = \theta$.)

(1) The log-normal model

That is regular log-normal parameters are restricted as `meanlog=-log(theta+1)/2`, `sdlog = sqrt(log(theta+1))`.

(2) The gamma model

That is regular gamma parameters are restricted as `shape=1/theta`, `scale=theta`.

(3) The semiparametric model

No distributional assumption and the random effect distribution is approximated by the estimated values of the quantity: $\gamma_i = w_i \frac{y_{i+}}{\mu_{i+}} + (1 - w_i)$, where: $y_{i+} = \sum_{j=1}^{n_i} y_{ij}$, $\mu_{i+} = \sum_{j=1}^{n_i} \mu_{ij}$ and, $w_i = \sqrt{\frac{Var(G_i)}{Var(Y_{i+}/\mu_{i+})}}$. See Zhao et al. (2013) for more details.

The missing count responses, if assumed to be missing at random, can be igored. Other types of missing data are currently not accepted.

Usage

```
lmeNB(formula,data,ID,p.ini=NULL,IPRT=FALSE,AR=FALSE,
      RE=c("G","N","NoN"),deps=1e-03,Vcode=NULL,C=FALSE,
      i.tol=1e-7,o.tol=sqrt(.Machine$double.eps),labelnp,
      maxit=100,semi.boot=0,u.low=0)
```

Arguments

`formula`	An object of class "formula" (or one that can be coerced to that class): a symbolic description of the model to be fitted. The formula must contain an intercept term.
`data`	A data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. The each row must contains the data corresponding to the repeated measure j of subjects and the rows (i, j)s must be ordered in a way that measurements from a subject is clustered together

as $(1,1)$,...,$(1,n_1)$,$(2,1)$,...,$(2,n_2)$,...,(N,n_N). Missing values are accepted of the response variables are treated as missing at random and simply removed from the data when AR=FALSE. When AR=TRUE, then the missing values are allowed. See the reference for missing value treatments. Missing values in covariates are not currently accepted.

ID — A vector of length $\sum_{i=1}^{N} n_i$, containing the patient IDs that corresponds to data. i.e., c(rep(ID_1,n_1),rep(ID_2,n_2),...,rep(ID_N,n_N)). The length must be the same as the number of rows of data. Missing ID values are NOT accepted.

p.ini — The initial values of the parameters:
If AR=0, p.ini= $(\log(\alpha), \log(Var(G_i))\ , \beta_0, \beta_1, ...)$.
If AR=1, p.ini=$(\log(\alpha), \log(Var(G_i)), logit(\delta), \beta_0, \beta_1, ...)$.
NULL is accepted.

IPRT — A logical, passed to Iprt of function optim. If TRUE then print iterations.

AR — A logical, if TRUE, then the AR(1) model is employed. If FALSE, then the independent model is employed.

RE — The distribution of random effects G_i. If RE="G", then the conditional probability is computed by assuming the random effect is from a gamma distribution with mean 1 and variance θ (gamma model). If RE="N", then the conditional probability is computed by assuming the random effect is from a log-normal distribution with mean 1 and variance θ (log-normal model). If RE="NoN", then the conditional probability is computed based on the semi-parametric model with mean 1 (semiparametric model).

deps — In the semiparametric models, the algorithms are terminated when the maximum difference of fixed effect coefficients between the current and previous iterations is less than deps. Passed to fitSemiIND and fitSemiAR1.

Vcode — Necessary only if the AR(1) model is fit. A vector of length the total number of repeated measures, containing the indices of time point. For example, there are three subjects and the first two subjects do not have missing visits and completed five visits while the last subject missed the third visit and have four visits in total, then Vcode=c(1,2,3,4,5,1,2,3,4,5,1,2,4,5).

i.tol — The absolute tolerance of integrate function, which is used to integrate out the random effect of every patient. Used only in parametric methods. i.tol should be about 1E-3 for C=TRUE option.

o.tol — The relative tolerance for optim function which is used to search for the maximum likelihood. Used only in parametric methods.

labelnp — See index.batch. Necessary only for semiparametric random effect model and semi.boot > 1. To account for the varying follow-up times of the patients, the bootstrap sampling is stratified according to the follow-up time.

maxit — The maximum number of iterations. Necessary only for semiparametric random effect model.

C — If C=TRUE, then the function uses the likelihood written in C. The integration of the random effect is done using Cubature (Multi-dimensional integration) package written by Steven G. Johnson. This option could make computation faster in

some scenario. If C=FALSE, then the function the likelihood is likelihood written in R language. The integration of the random effect is done using integrate function in the stats package.

semi.boot The number of bootstrap samples to construct the bootstrap empirical confidence intervals for the fixed effect coefficients. If the value is less than 1 then the bootstrap confidence intervals are not computed. Necessary only for semiparametric random effect model.

u.low For the semiparametric procedures, we observed that the algorithm could behave very unstable when factor covariates are employed in the dataset, and data contains "few" repeated measures of one of the corresponding factor groups: The algorithm could yield unacceptably small estimate of μ_{ij}. As $Var(Y_{ij})$ and $Cov(Y_{ij}, Y_{ij'})$ are both multiples of μ_{ij} (see description above), small μ_{ij} leads to singular $Var(\boldsymbol{Y}_i)$ matrix. As a result, the algorithm breakdown when computing the weighted matrix of the weighted least square, $\boldsymbol{W}_i$=$Var(\boldsymbol{Y}_i)^{-1}$. To prevent this issue, the current algorithm takes add-hoc treatment, and replaces small μ_{ij} i.e. those with $\mu_{ij} <$ u.low with u.low when calculating the weight matrix $\boldsymbol{W}_i$.

u.low=0 means that there is no modification, and the smaller u.low, the "closer" the modified algorithm is to the original one proposed in Zhao et al. (2013). Our preliminary study indicates that u.low> 1E-4 prevents the breakdown problem and the performance of the algorithm are similar when 1E-3 < u.low < 1E-1 in terms of the root mean square error of the conditional probability index.

Details

fitParaIND calls optim to minimize the negative log-likelihood of the negative binomial model with respect to the model parameters: $\log(\alpha), \log(\theta), \beta_0, \beta_1, ...$ The Nelder-Mead algorithm is employed. The log-likelihood is obtained by marginalizing out the random effects. The numerical integration is carried out using adaptive quadrature. The missing count responses, if assumed to be missing at random, can be igored. Other types of missing data are currently not accepted.

When the estimated over-dispersion parameter (α) is close to zero, the negative binomial model reduces to the poisson model, suggesting that the negative binomial mixed-effect model might not be appropriate. When AR=1 and the estimated auto-correlation parameter (δ) is close to zero, the model is suggesting that there is no AR(1) structure among the sequentially collected responses. Hence user might use AR=0 setting which assumes no AR(1) structure.

We note that the results could be sensitive to initial values.

Value

call The input of the function.

opt If RE="G" or RE="N", then opt contains the results directly from the optim function, which is used to minimize the negative of the log-likelihood.

If RE="NoN", then opt contains the results directly from the optimize function, which is used to minimize the negative of the psudo-profile log-likelihood with respect to the dispersion parameter alpha, a.

nlk The value of the negative log-likelihood corresponding to opt$par.

V	If RE="G" or RE="N", the approximated asymptotic covariance matrix of the maximum likelihood estimators, i.e. V=solve(opt$hessian). If opt$hessian is not invertible, then V = matrix(NA,length(p.ini),length(p.ini)). If RE="NoN", AR=FALSE and semi.boot>0, V contains the bootstrap covariance matrix based on semi.boot samples.
est	The matrix of the number of fixed-effect parameters (i.e. length(p.ini)) by 2. The first column contains the estimates of the model parameters. The second column contains the standard error of the estimators, i.e., sqrt(diag(V)). If V is not evaluated, then est only has one column.

Author(s)

Zhao, Y. and Kondo, Y.

References

Detection of unusual increases in MRI lesion counts in individual multiple sclerosis patients. (2013) Zhao, Y., Li, D.K.B., Petkau, A.J., Riddehough, A., Traboulsee, A., Journal of the American Statistical Association.

A flexible mixed effect negative binomial regression model for detecting unusual increases in MRI lesion counts in individual multiple sclerosis patients. Kondo, Y., Zhao, Y., Petkau, A.J.

See Also

The subroutines of this function is: fitParaIND, fitParaAR1, fitSemiIND, fitSemiAR1,

The subroutines of index.batch to compute the conditional probability index: jCP.ar1, CP1.ar1, MCCP.ar1, CP.ar1.se, CP.se, jCP,

The functions to generate simulated datasets: rNBME.R.

Examples

```
## Not run:
## ============================================================================== ##
## generate a simulated dataset from the negative binomial mixed-effect
## independent model:
## Y_ij | G_i = g_i ~ NB(r_ij,p_i) where r_ij = exp(X^T beta)/a , p_i =1/(a*g_i+1)
## with G_i ~ log-normal(E(G_i)=1,var(G_i)=th)
set.seed(1)
sn <- 5 # the number of repeated measures of each patient
n <- 80 ## the number of patients
loga <- - 0.5
a <- exp(loga) ## dispersion parameter
logtheta <- 1.3
th <- exp(logtheta) ## the variance of the gamma distributed random effect g_i


## No difference between the means of groups
## The model only has an intercept term beta0 <- 0.5
```

```
b0 <- 0.5
u1 <- rep(exp(b0),sn)  ## the mean vector for group 1 at time point 1,...,sn
u2 <- rep(exp(b0),sn) ## the mean vector for group 2 at time point 1,...,sn

## DT.ind is generated from the IND model
DT.ind<- rNBME.R(gdist="N", n=n, a=a, th=th, u1=u1, u2=u2, sn=sn)
## DT.ar is generated from AR(1) model with delta=0.4
DT.ar<- rNBME.R(gdist="N", n=n, a=a, th=th, u1=u1, u2=u2, sn=sn, d=0.4)

dt.ind<-data.frame(CEL=DT.ind$y,Vcode=DT.ind$vn-2,ID=DT.ind$id)
dt.ar<-data.frame(CEL=DT.ar$y,Vcode=DT.ar$vn-2,ID=DT.ar$id)
## ============================================================================== ##
#### ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ ####
#### Fit various parametric independent models ####
#### ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ ####
tst1 <- lmeNB(CEL~1, data=dt.ind, ID=dt.ind$ID, IPRT=TRUE)            ## gamma Gi
tst2 <- lmeNB(CEL~1, data=dt.ar, ID=dt.ar$ID, IPRT=TRUE)              ## gamma Gi
tst3 <- lmeNB(CEL~1, data=dt.ind, ID=dt.ind$ID, IPRT=TRUE, RE="N")    ## log-normal Gi
tst4 <- lmeNB(CEL~1, data=dt.ar, ID=dt.ar$ID, IPRT=TRUE, RE="N")      ## log-normal Gi
tst5 <- lmeNB(CEL~Vcode, data=dt.ind, ID=dt.ind$ID, IPRT=TRUE, RE="N")## log-normal Gi
## conditional probability index with the fitted results of tst5
Psum5 <- index.batch(olmeNB=tst5, labelnp=dt.ind$Vcode >= 1, data=dt.ind, ID=dt.ind$ID)


#### ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ ####
#### Fit the semi-parametric independent model ####
#### ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ ####
tst6 <- lmeNB(CEL~1, data=dt.ind, ID=dt.ind$ID, IPRT=TRUE, RE="NoN")
tst7 <- lmeNB(CEL~Vcode, data=dt.ind, ID=dt.ind$ID, IPRT=TRUE, RE="NoN",
              semi.boot=100,labelnp=dt.ind$Vcode >= 1)
## semi.boot = 100 option computes bootstrap SE and 95
## conditional probability index with fitting result of tst7
Psum7 <- index.batch(olmeNB=tst7,labelnp=dt.ind$Vcode >= 1, data=dt.ind,
                     ID=dt.ind$ID, Vcode=Vcode)

#### ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ ####
#### Fit the parametric AR1 models ####
#### ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ ####
tst8 <- lmeNB(CEL~1, data=dt.ind, ID=dt.ind$ID, IPRT=TRUE,AR=TRUE,Vcode=dt.ind$Vcode)
tst9 <- lmeNB(CEL~1, data=dt.ar, ID=dt.ar$ID, IPRT=TRUE,AR=TRUE,Vcode=dt.ar$Vcode)
## conditional probability index
Psum9 <- index.batch(olmeNB=tst9, labelnp=dt.ind$Vcode >= 1, data=dt.ind,
                     ID=dt.ind$ID,Vcode=dt.ind$Vcode)



## End(Not run)
```

RElmeNB *Calculate predicted values of E(Gi|Yi) given the estimates of parameters*

Description

Compute predicted values of random effects for each patient

Usage

```
RElmeNB(theta, alpha, betas, delta, formula, ID, Vcode = NULL,
        data, AR, RE, rel.tol = .Machine$double.eps^0.8, expG = FALSE)
```

Arguments

theta	A scalar containing the estimated variance of the random effect distribution, θ.
alpha	A scalar containing the estimated dispersion parameter, α.
betas	A vector containing the estimated regression coefficients, $\boldsymbol{\beta}$.
delta	AR(1) parameter, δ
ID	See lmeNB.
Vcode	Necessary only if the AR(1) model is used. See lmeNB.
RE	The distribution of random effects G_i. If RE="G" then the random effects are assumed to be from the gamma distribution. If RE="N" then they are assumed to be form the log-normal distribution. The current version of RElmeNB only accept parametric model.
AR	See lmeNB.
formula	See lmeNB.
data	See lmeNB.
rel.tol	relative tolerance for the integration of the random effect. passed to integrate function.
expG	Internal use only

Value

return the predicted RE of each patient.

Author(s)

Zhao, Y. and Kondo, Y.

References

Detection of unusual increases in MRI lesion counts in individual multiple sclerosis patients. (2013) Zhao, Y., Li, D.K.B., Petkau, A.J., Riddehough, A., Traboulsee, A., Journal of the American Statistical Association.

See Also

The main function to fit the Negative Binomial mixed-effect model: lmeNB,

The subroutines of this function is: fitParaIND, fitParaAR1, fitSemiIND, fitSemiAR1,

The subroutines of index.batch to compute the conditional probability index: jCP.ar1, CP1.ar1, MCCP.ar1, CP.ar1.se, CP.se, jCP,

The functions to generate simulated datasets: rNBME.R.

Examples

```
## See the examples in help files of rNBME.R.
```

---

rNBME.R *Simulate a dataset from the negative binomial mixed-effect independent/AR(1) model*

Description

This function simulates a dataset based on the negative binomial mixed-effect independent/AR(1) model with two treatment groups described in Zhao et al (2013). The group mean can be different at each time point, but no other covariates are allowed.

See fitParaIND, fitParaAR1 for details of the model explanations.

Usage

```
rNBME.R(
gdist = "G", n = 200, sn = 5, th = exp(1.3),
u1 = rep(1.5, 5), u2 = rep(1.5, 5),
a = exp(-0.5),d=NULL,  othrp = list(u.n = 3, s.n = 0.5, p.mx = 0.05, sh.mx = NA)
)
```

Arguments

gdist	The distribution of the random effect term G_i. If gdist="G", G_i is from the gamma distribution. If gdist="N", G_i is from the log normal distribution. If gdist="U", G_i (on the log scale) is from the uniform distribution. If gdist="GN", G_i is from the mixture of the gamma distribution and the normal distribution. If the generated values are negative, they are truncated to zero. If gdist="NoN", G_i is sampled from the pre-specified vector othrp with replacement.
n	The number of patients. It must be an even number.

sn	The number of repeated measures per patient. Generated datasets are balanced design.
th	If gdist="G", th is a scale parameter of the gamma distribution. If gdist="N" or gdist=="U", th is $Var(G_i)$. If gdist="GN", see details. If gdist="NoN", this parameter is not used.
u1	A vector of length *sn*, specifying the mean of the treatment group 1 $E(Y_ij)$ = u1[j].
u2	A vector of length sn, specifying the mean of the treatment group 2 $E(Y_ij)$ = u2[j].
a	The dispersion parameter α of the negative binomial mixed-effect independent model. See description in *lmeNB*.
d	If d=NULL, generate data from the independent model. If d is a scalar between 0 and 1, then d is δ in the AR(1) model, and generate datasets from the AR(1) model.
othrp	If gdist="GN", parameters for the GN option. See details. If gdist="NoN", othrp is a vector, containing a sample of G_i, which is treated as a population and G_i is resampled.

Details

The generated datasets have equal number of scans per person.

The number of patients in the two groups are the same.

If gdist=="GN", datasets are generated from:

othrp$p.mx*N(mean=othrp$u.n,s.d=othrp$s.n) + (1-othrp$p.mx)*gamma(scale=th,shape),

where shape of the gamma distribution is chosen to ensure $E(G_i) = 1$.

Value

id	The vector of length n*sn containing patient IDs: rep(1:n,each=sn)
vn	The vector of length n*sn containing the indicies of time points: rep(1:sn, n)
gp	The vector of length n*sn containing the indicies of the treatment groups
y	The vector of length n*sn containing generated response counts
g	The vector of length n*sn containing generated random effect terms
Gpara	The record of the distribution and parameter specifications used to generate the dataset

Author(s)

Zhao, Y. and Kondo, Y.

References

Detection of unusual increases in MRI lesion counts in individual multiple sclerosis patients. (2013) Zhao, Y., Li, D.K.B., Petkau, A.J., Riddehough, A., Traboulsee, A., Journal of the American Statistical Association.

See Also

The main function to fit the Negative Binomial mixed-effect model: lmeNB,

The functions to fit related models: fitParaIND, fitParaAR1, fitSemiIND, fitSemiAR1,

The subroutines of index.batch to compute the conditional probability index: jCP.ar1, CP1.ar1, MCCP.ar1, CP.ar1.se, CP.se, jCP,

Examples

```
## Not run:
## See the examples in help files of fitParaIND, fitParaAR1, fitSemiIND, fitSemiAR1 and lmeNB


## End(Not run)
```

Index