Package ‘geeCRT’

April 15, 2021

Type Package

Title Bias-Corrected GEE for Cluster Randomized Trials

Version 0.1.0

Date 2021-04-14

Maintainer Hengshi Yu <hengshi@umich.edu>

Description Population-averaged models have been increasingly used in the design and analysis of cluster randomized trials (CRTs). To facilitate the applications of population-averaged models in CRTs, the package implements the generalized estimating equations (GEE) and matrix-adjusted estimating equations (MAEE) approaches to jointly estimate the marginal mean models correlation models both for general CRTs and stepped wedge CRTs. Despite the general GEE/MAEE approach, the package also implements a fast cluster-period GEE method by Li et al. (2021) <doi:10.1093/biostatistics/kxaa056> specifically for stepped wedge CRTs with large and variable cluster-period sizes and gives a simple and efficient estimating equations approach based on the cluster-period means to estimate the intervention effects as well as correlation parameters. In addition, the package also provides functions for generating correlated binary data with specific mean vector and correlation matrix based on the multivariate probit method in Emrich and Piedmonte (1991) <doi:10.1080/00031305.1991.10475828> or the conditional linear family method in Qaqish (2003) <doi:10.1093/biomet/90.2.455>.

License GPL (>= 2)

LazyData TRUE

Depends R (>= 3.6.0)

Imports MASS, rootSolve, mvtnorm

Suggests knitr, rmarkdown

VignetteBuilder knitr

RoxygenNote 7.1.1

Encoding UTF-8

NeedsCompilation no

Author Hengshi Yu [aut, cre],
Fan Li [aut],
Paul Rathouz [aut],
Elizabeth L. Turner [aut],
John Preisser [aut]

Repository CRAN

Date/Publication 2021-04-15 07:10:02 UTC

R topics documented:

`cpgeeSWD` *Cluster-Period GEE for Estimating the Mean and Correlation Parameters in Cross-Sectional SW-CRTs*

Description

cpgeeSWD implements the cluster-period GEE developed for cross-sectional stepped wedge cluster randomized trials (SW-CRTs). It provides valid estimation and inference for the treatment effect and intraclass correlation parameters within the GEE framework, and is computationally efficient for SW-CRTs with large cluster sizes. The program currently only allows for a marginal mean model with discrete period effects and the intervention indicator without additional covariates. The program offers bias-corrected ICC estimates as well as bias-corrected sandwich variances for both the treatment effect parameter and the ICC parameters. The technical details of the cluster-period GEE approach are provided in Li et al. (2020+).

Usage

```
cpgeeSWD(
  y,
  X,
  id,
  m,
  corstr,
  family = "binomial",
  maxiter = 500,
  epsilon = 0.001,
  printrange = TRUE,
  alpadj = FALSE,
  rho.init = NULL
)
```

Arguments

- `y` a vector specifying the cluster-period means (proportions)
- `X` design matrix for the marginal mean model, including period indicator and intervention indicator
- `id` a vector specifying cluster identifier
- `m` a vector of the cluster-period sizes
- `corstr` correlation structure specified for the individual-level outcomes, could be `'exchangeable'`, `'nest_exch'` or `'exp_decay'`
- `family` See corresponding documentation to `glm`. The current version only supports `family = 'binomial'`
- `maxiter` maximum number of iterations for Fisher scoring updates
- `epsilon` tolerance for convergence
- `printrange` print details of range violations when `family = 'binomial'`. The default is `TRUE`
- `alpadj` if `TRUE`, performs bias adjustment for the alpha estimating equations. The default is `FALSE`
- `rho.init` user-specified initial value for the decay parameter when `corstr = 'exp_decay'`

Value

`outbeta` estimates of marginal mean model parameters and standard errors with different finite-sample bias corrections. The current version supports model-based standard error (MB), the sandwich standard error (BC0) extending Zhao and Prentice (2001), the sandwich standard errors (BC1) extending Kauermann and Carroll (2001), the sandwich standard errors (BC2) extending Mancl and DeRouen (2001), and the sandwich standard errors (BC3) extending the Fay and Graubard (2001). A summary of these bias-corrections can also be found in Lu et al. (2007), and Li et al. (2018).

`outalpha` estimates of correlation parameters and standard errors with different finite-sample bias corrections. The current version supports the sandwich standard error (BC0) extending Zhao and Prentice (2001), the sandwich standard errors (BC1) extending Kauermann and Carroll (2001), the sandwich standard errors (BC2) extending Mancl and DeRouen (2001), and the sandwich standard errors (BC3) extending the Fay and Graubard (2001). A summary of these bias-corrections can also be found in Preisser et al. (2008).

`beta` a vector of estimates for marginal mean model parameters

`alpha` a vector of estimates of correlation parameters

`MB` model-based covariance estimate for the marginal mean model parameters

`BC0` robust sandwich covariance estimate of the marginal mean model and correlation parameters

`BC1` robust sandwich covariance estimate of the marginal mean model and correlation parameters with the Kauermann and Carroll (2001) correction

`BC2` robust sandwich covariance estimate of the marginal mean model and correlation parameters with the Mancl and DeRouen (2001) correction

`BC3` robust sandwich covariance estimate of the marginal mean model and correlation parameters with the Fay and Graubard (2001) correction

`niter` number of iterations used in the Fisher scoring updates for model fitting

Author(s)

Hengshi Yu <hengshi@umich.edu>, Fan Li <fan.f.li@yale.edu>, Paul Rathouz <paul.rathouz@austin.utexas.edu>, Elizabeth L. Turner <liz.turner@duke.edu>, John Preisser <jpreisse@bios.unc.edu>

References

Zhao, L. P., Prentice, R. L. (1990). Correlated binary regression using a quadratic exponential model. Biometrika, 77(3), 642-648.

Mancl, L. A., DeRouen, T. A. (2001). A covariance estimator for GEE with improved small sample properties. Biometrics, 57(1), 126-134.

Kauermann, G., Carroll, R. J. (2001). A note on the efficiency of sandwich covariance matrix estimation. Journal of the American Statistical Association, 96(456), 1387-1396.

Fay, M. P., Graubard, B. I. (2001). Small sample adjustments for Wald type tests using sandwich estimators. Biometrics, 57(4), 1198-1206.

Lu, B., Preisser, J. S., Qaqish, B. F., Suchindran, C., Bangdiwala, S. I., Wolfson, M. (2007). A comparison of two bias corrected covariance estimators for generalized estimating equations. Biometrics, 63(3), 935-941.

Preisser, J. S., Lu, B., Qaqish, B. F. (2008). Finite sample adjustments in estimating equations and covariance estimators for intracluster correlations. Statistics in Medicine, 27(27), 5764-5785.

Li, F., Turner, E. L., Preisser, J. S. (2018). Sample size determination for GEE analyses of stepped wedge cluster randomized trials. Biometrics, 74(4), 1450-1458.

Li, F. (2020). Design and analysis considerations for cohort stepped wedge cluster randomized trials with a decay correlation structure. Statistics in Medicine, 39(4), 438-455.

Li, F., Yu, H., Rathouz, P., Turner, E. L., Preisser, J. S. (2021). Marginal modeling of cluster-period means and intraclass correlations in stepped wedge designs with binary outcomes. Biostatistics, kxaa056.

Examples

```
# Simulated SW-CRT example with binary outcome

########################################################################
### Example 1): simulated SW-CRT with smaller cluster-period sizes (5~10)
########################################################################

sampleSWCRT = sampleSWCRTSmall

#############################################################
### cluster-period id, period, outcome, and design matrix ###
#############################################################

### id, period, outcome
id = sampleSWCRT$id; period =  sampleSWCRT$period; y =  sampleSWCRT$y_bin
X = as.matrix(sampleSWCRT[, c('period1', 'period2', 'period3', 'period4', 'treatment')])

m = as.matrix(table(id, period)); n = dim(m)[1]; t = dim(m)[2]
```

```
clp_mu<-tapply(y,list(id,period), FUN=mean); y_cp <- c(t(clp_mu))

### design matrix for correlation parameters
trt <- tapply(X[, t + 1], list(id, period), FUN=mean); trt <- c(t(trt))

time <- tapply(period,list(id, period), FUN = mean); time <- c(t(time)); X_cp <- matrix(0, n * t, t)

s = 1
for(i in 1:n){for(j in 1:t){X_cp[s, time[s]] <- 1; s = s + 1}}
X_cp <- cbind(X_cp, trt); id_cp <- rep(1:n, each= t); m_cp <-  c(t(m))

#####################################################
### cluster-period matrix-adjusted estimating equations (MAEE)
### with exchangeable, nested exchangeable and exponential decay correlation structures ###
#####################################################

# exchangeable
est_maee_exc <- cpgeeSWD(y = y_cp, X = X_cp, id = id_cp,
                         m = m_cp, corstr = 'exchangeable',
                         alpadj = TRUE)
print(est_maee_exc)

# nested exchangeable
est_maee_nex <- cpgeeSWD(y = y_cp, X = X_cp, id = id_cp,
                         m = m_cp, corstr = 'nest_exch',
                         alpadj = TRUE)
print(est_maee_nex)

# exponential decay
est_maee_ed <- cpgeeSWD(y = y_cp, X = X_cp, id = id_cp,
                        m = m_cp, corstr = 'exp_decay',
                        alpadj = TRUE)
print(est_maee_ed)

#####################################################
### cluster-period GEE
### with exchangeable, nested exchangeable and exponential decay correlation structures ###
#####################################################

# exchangeable
est_uee_exc <- cpgeeSWD(y = y_cp, X = X_cp, id = id_cp,
                        m = m_cp, corstr = 'exchangeable',
                        alpadj = FALSE)
print(est_uee_exc)

# nested exchangeable
est_uee_nex <- cpgeeSWD(y = y_cp, X = X_cp, id = id_cp,
                        m = m_cp, corstr = 'nest_exch',
                        alpadj = FALSE)
print(est_uee_nex)

# exponential decay
est_uee_ed <- cpgeeSWD(y = y_cp, X = X_cp, id = id_cp,
```

```
                       m = m_cp, corstr = 'exp_decay',
                       alpadj = FALSE)
print(est_uee_ed)

########################################################################
### Example 2): simulated SW-CRT with larger cluster-period sizes (20~30)
########################################################################

sampleSWCRT = sampleSWCRTLarge

#############################################################
### cluster-period id, period, outcome, and design matrix ###
#############################################################

### id, period, outcome
id = sampleSWCRT$id; period =  sampleSWCRT$period; y =  sampleSWCRT$y_bin
X = as.matrix(sampleSWCRT[, c('period1', 'period2', 'period3', 'period4', 'period5', 'treatment')])

m = as.matrix(table(id, period)); n = dim(m)[1]; t = dim(m)[2]
clp_mu<-tapply(y,list(id,period), FUN=mean); y_cp <- c(t(clp_mu))

### design matrix for correlation parameters
trt <- tapply(X[, t + 1], list(id, period), FUN=mean); trt <- c(t(trt))

time <- tapply(period,list(id, period), FUN = mean); time <- c(t(time)); X_cp <- matrix(0, n * t, t)

s = 1
for(i in 1:n){for(j in 1:t){X_cp[s, time[s]] <- 1; s = s + 1}}
X_cp <- cbind(X_cp, trt); id_cp <- rep(1:n, each= t); m_cp <-  c(t(m))

#####################################################
### cluster-period matrix-adjusted estimating equations (MAEE)
### with exchangeable, nested exchangeable and exponential decay correlation structures ###
#####################################################

# exchangeable
est_maee_exc <- cpgeeSWD(y = y_cp, X = X_cp, id = id_cp,
                         m = m_cp, corstr = 'exchangeable',
                         alpadj = TRUE)
print(est_maee_exc)

# nested exchangeable
est_maee_nex <- cpgeeSWD(y = y_cp, X = X_cp, id = id_cp,
                         m = m_cp, corstr = 'nest_exch',
                         alpadj = TRUE)
print(est_maee_nex)

# exponential decay
est_maee_ed <- cpgeeSWD(y  = y_cp, X = X_cp, id = id_cp,
                        m = m_cp, corstr = 'exp_decay',
                        alpadj = TRUE)
print(est_maee_ed)
```

```
####################################################
### cluster-period GEE
### with exchangeable, nested exchangeable and exponential decay correlation structures ###
####################################################

# exchangeable
est_uee_exc <- cpgeeSWD(y = y_cp, X = X_cp, id = id_cp,
                        m = m_cp, corstr = 'exchangeable',
                        alpadj = FALSE)
print(est_uee_exc)

# nested exchangeable
est_uee_nex <- cpgeeSWD(y = y_cp, X = X_cp, id = id_cp,
                        m = m_cp, corstr = 'nest_exch',
                        alpadj = FALSE)
print(est_uee_nex)

# exponential decay
est_uee_ed <- cpgeeSWD(y = y_cp, X = X_cp, id = id_cp,
                       m = m_cp, corstr = 'exp_decay',
                       alpadj = FALSE)
print(est_uee_ed)
```

`geeCRT` *geeCRT: a package for implementing the bias-corrected generalized estimating equations in analyzing cluster randomized trials*

Description

geeCRT: a package for implementing the bias-corrected generalized estimating equations in analyzing cluster randomized trials

geeCRT functions

The simbinPROBIT function performs correlated binary outcome data simulation using the multivariate probit method The simbinCLF function performs correlated binary outcome data simulation using the conditional linear family method The cpgeeSWD function performs cluster-period generalized estimating equations for estimating the marginal mean and correlation parameters in cross-sectional stepped wedge cluster randomized trials The geemaee function performs matrix-adjusted generalized estimating equations on estimating the marginal mean and correlation parameters in cluster randomized trials geemaee *GEE and Matrix-adjusted Estimating Equations (MAEE) for Estimating the Marginal Mean and Correlation Parameters in CRTs*

Description

geemaee implements the GEE and matrix-adjusted estimating equations (MAEE) for analyzing cluster randomized trials (CRTs). It supports estimation and inference for the marginal mean and intraclass correlation parameters within the population-averaged modeling framework. With suitable choice of the design matrices, the function can be used to analyze parallel, crossover and stepped wedge cluster randomized trials. The program also offers bias-corrected intraclass correlation estimates, as well as bias-corrected sandwich variances for both the marginal mean and correlation parameters. The technical details of the GEE and MAEE approach are provided in Preisser (2008) and Li et al. (2018, 2019).

Usage

```
geemaee(
  y,
  X,
  id,
  Z,
  family,
  maxiter = 500,
  epsilon = 0.001,
  printrange = TRUE,
  alpadj = FALSE,
  shrink = "ALPHA",
  makevone = TRUE
)
```

Arguments

`y`	a vector specifying the outcome variable across all clusters
`X`	design matrix for the marginal mean model, including the intercept
`id`	a vector specifying cluster identifier
`Z`	design matrix for the correlation model, should be all pairs j < k for each cluster
`family`	See corresponding documentation to `glm`. The current version only supports `'continuous'` and `'binomial'`
`maxiter`	maximum number of iterations for Fisher scoring updates
`epsilon`	tolerance for convergence. The default is 0.001
`printrange`	print details of range violations. The default is `TRUE`
`alpadj`	if `TRUE`, performs bias adjustment for the correlation estimating equations. The default is `FALSE`

`shrink`	method to tune step sizes in case of non-convergence including 'THETA' or 'ALPHA'. The default is 'ALPHA'
`makevone`	if TRUE, it assumes unit variances for the correlation parameters in the correlation estimating equations. The default is TRUE

Value

`outbeta` estimates of marginal mean model parameters and standard errors with different finite-sample bias corrections. The current version supports model-based standard error (MB), the sandwich standard error (BC0) extending Liang and Zeger (1986), the sandwich standard errors (BC1) extending Kauermann and Carroll (2001), the sandwich standard errors (BC2) extending Mancl and DeRouen (2001), and the sandwich standard errors (BC3) extending the Fay and Graubard (2001). A summary of these bias-corrections can also be found in Lu et al. (2007), and Li et al. (2018).

`outalpha` estimates of intraclass correlation parameters and standard errors with different finite-sample bias corrections. The current version supports the sandwich standard error (BC0) extending Zhao and Prentice (2001), the sandwich standard errors (BC1) extending Kauermann and Carroll (2001), the sandwich standard errors (BC2) extending Mancl and DeRouen (2001), and the sandwich standard errors (BC3) extending the Fay and Graubard (2001). A summary of these bias-corrections can also be found in Preisser et al. (2008).

`beta` a vector of estimates for marginal mean model parameters

`alpha` a vector of estimates of correlation parameters

`MB` model-based covariance estimate for the marginal mean model parameters

`BC0` robust sandwich covariance estimate of the marginal mean model and correlation parameters

`BC1` robust sandwich covariance estimate of the marginal mean model and correlation parameters with the Kauermann and Carroll (2001) correction

`BC2` robust sandwich covariance estimate of the marginal mean model and correlation parameters with the Mancl and DeRouen (2001) correction

`BC3` robust sandwich covariance estimate of the marginal mean model and correlation parameters with the Fay and Graubard (2001) correction

`niter` number of iterations used in the Fisher scoring updates for model fitting

Author(s)

Hengshi Yu <hengshi@umich.edu>, Fan Li <fan.f.li@yale.edu>, Paul Rathouz <paul.rathouz@austin.utexas.edu>, Elizabeth L. Turner <liz.turner@duke.edu>, John Preisser <jpreisse@bios.unc.edu>

References

Liang, K. Y., Zeger, S. L. (1986). Longitudinal data analysis using generalized linear models. Biometrika, 73(1), 13-22.

Prentice, R. L. (1988). Correlated binary regression with covariates specific to each binary observation. Biometrics, 1033-1048.

Zhao, L. P., Prentice, R. L. (1990). Correlated binary regression using a quadratic exponential model. Biometrika, 77(3), 642-648.

Prentice, R. L., Zhao, L. P. (1991). Estimating equations for parameters in means and covariances of multivariate discrete and continuous responses. Biometrics, 825-839.

Sharples, K., Breslow, N. (1992). Regression analysis of correlated binary data: some small sample results for the estimating equation approach. Journal of Statistical Computation and Simulation, 42(1-2), 1-20.

Mancl, L. A., DeRouen, T. A. (2001). A covariance estimator for GEE with improved small sample properties. Biometrics, 57(1), 126-134.

Kauermann, G., Carroll, R. J. (2001). A note on the efficiency of sandwich covariance matrix estimation. Journal of the American Statistical Association, 96(456), 1387-1396.

Fay, M. P., Graubard, B. I. (2001). Small sample adjustments for Wald type tests using sandwich estimators. Biometrics, 57(4), 1198-1206.

Lu, B., Preisser, J. S., Qaqish, B. F., Suchindran, C., Bangdiwala, S. I., Wolfson, M. (2007). A comparison of two bias corrected covariance estimators for generalized estimating equations. Biometrics, 63(3), 935-941.

Preisser, J. S., Lu, B., Qaqish, B. F. (2008). Finite sample adjustments in estimating equations and covariance estimators for intracluster correlations. Statistics in Medicine, 27(27), 5764-5785.

Li, F., Turner, E. L., Preisser, J. S. (2018). Sample size determination for GEE analyses of stepped wedge cluster randomized trials. Biometrics, 74(4), 1450-1458.

Li, F., Forbes, A. B., Turner, E. L., Preisser, J. S. (2019). Power and sample size requirements for GEE analyses of cluster randomized crossover trials. Statistics in Medicine, 38(4), 636-649.

Li, F. (2020). Design and analysis considerations for cohort stepped wedge cluster randomized trials with a decay correlation structure. Statistics in Medicine, 39(4), 438-455.

Li, F., Yu, H., Rathouz, P., Turner, E. L., Preisser, J. S. (2021). Marginal modeling of cluster-period means and intraclass correlations in stepped wedge designs with binary outcomes. Biostatistics, kxaa056.

Examples

```
# Simulated SW-CRT examples

#################################################################
### function to create the design matrix for correlation parameters
### under the nested exchangeable correlation structure
#################################################################
createzCrossSec = function (m) {
      Z = NULL
      n = dim(m)[1]
      for (i in 1:n) {
          alpha_0 = 1; alpha_1 = 2; n_i = c(m[i, ]); n_length = length(n_i)
          POS = matrix(alpha_1, sum(n_i), sum(n_i))
          loc1 = 0; loc2 = 0
          for (s in 1:n_length) {
              n_t = n_i[s]; loc1 = loc2 + 1; loc2 = loc1 + n_t - 1
              for (k in loc1:loc2) {
                  for (j in loc1:loc2) {
                      if (k != j) {POS[k, j] = alpha_0} else {POS[k, j] = 0}}}}
```

```
          zrow = diag(2); z_c = NULL
          for (j in 1:(sum(n_i) - 1)) {
              for (k in (j + 1):sum(n_i)) {z_c = rbind(z_c, zrow[POS[j,k],])}}
          Z = rbind(Z, z_c)}
     return(Z)}

########################################################################
### Example 1): simulated SW-CRT with smaller cluster-period sizes (5~10)
########################################################################

sampleSWCRT = sampleSWCRTSmall

###############################################################
### Individual-level id, period, outcome, and design matrix ###
###############################################################

id = sampleSWCRT$id; period =  sampleSWCRT$period;
X = as.matrix(sampleSWCRT[, c('period1', 'period2', 'period3', 'period4', 'treatment')])

m = as.matrix(table(id, period)); n = dim(m)[1]; t = dim(m)[2];

### design matrix for correlation parameters
Z = createzCrossSec(m)

################################################################
### (1) Matrix-adjusted estimating equations and GEE
### on continous outcome with nested exchangeable correlation structure
################################################################

### MAEE
est_maee_ind_con = geemaee(y = sampleSWCRT$y_con, X = X, id = id,
                           Z = Z, family = 'continuous',
                           maxiter = 500, epsilon = 0.001,
                           printrange = TRUE, alpadj = TRUE,
                           shrink = 'ALPHA', makevone = FALSE)
print(est_maee_ind_con)


### GEE
est_uee_ind_con = geemaee(y = sampleSWCRT$y_con, X = X, id = id,
                          Z = Z, family = 'continuous',
                          maxiter = 500, epsilon = 0.001,
                          printrange = TRUE, alpadj = FALSE,
                          shrink = 'ALPHA', makevone = FALSE)
print(est_uee_ind_con)


###############################################################
### (2) Matrix-adjusted estimating equations and GEE
### on binary outcome with nested exchangeable correlation structure
###############################################################

### MAEE
```

```
est_maee_ind_bin = geemaee(y = sampleSWCRT$y_bin, X = X, id = id,
                           Z = Z, family = 'binomial',
                           maxiter = 500, epsilon = 0.001,
                           printrange = TRUE, alpadj = TRUE,
                           shrink = 'ALPHA', makevone = FALSE)
print(est_maee_ind_bin)


### GEE
est_uee_ind_bin = geemaee(y = sampleSWCRT$y_bin, X = X, id = id,
                          Z = Z, family = 'binomial',
                          maxiter = 500, epsilon = 0.001,
                          printrange = TRUE, alpadj = FALSE,
                          shrink = 'ALPHA', makevone = FALSE)
print(est_uee_ind_bin)



## This will elapse longer.
#########################################################################
### Example 2): simulated SW-CRT with larger cluster-period sizes (20~30)
#########################################################################

sampleSWCRT = sampleSWCRTLarge

################################################################
### Individual-level id, period, outcome, and design matrix ###
################################################################

id = sampleSWCRT$id; period =  sampleSWCRT$period;
X = as.matrix(sampleSWCRT[, c('period1', 'period2', 'period3', 'period4', 'period5', 'treatment')])

m = as.matrix(table(id, period)); n = dim(m)[1]; t = dim(m)[2];

### design matrix for correlation parameters
Z = createzCrossSec(m)

#################################################################
### (1) Matrix-adjusted estimating equations and GEE
### on continous outcome with nested exchangeable correlation structure
#################################################################

### MAEE
est_maee_ind_con = geemaee(y = sampleSWCRT$y_con, X = X, id = id,
                           Z = Z, family = 'continuous',
                           maxiter = 500, epsilon = 0.001,
                           printrange = TRUE, alpadj = TRUE,
                           shrink = 'ALPHA', makevone = FALSE)
print(est_maee_ind_con)


### GEE
est_uee_ind_con = geemaee(y = sampleSWCRT$y_con, X = X, id = id,
```

```
                          Z = Z, family = 'continuous',
                          maxiter = 500, epsilon = 0.001,
                          printrange = TRUE, alpadj = FALSE,
                          shrink = 'ALPHA', makevone = FALSE)
print(est_uee_ind_con)

###############################################################
### (2) Matrix-adjusted estimating equations and GEE
### on binary outcome with nested exchangeable correlation structure
###############################################################

### MAEE
est_maee_ind_bin = geemaee(y = sampleSWCRT$y_bin, X = X, id = id,
                           Z = Z, family = 'binomial',
                           maxiter = 500, epsilon = 0.001,
                           printrange = TRUE, alpadj = TRUE,
                           shrink = 'ALPHA', makevone = FALSE)
print(est_maee_ind_bin)


### GEE
est_uee_ind_bin = geemaee(y = sampleSWCRT$y_bin, X = X, id = id,
                          Z = Z, family = 'binomial',
                          maxiter = 500, epsilon = 0.001,
                          printrange = TRUE, alpadj = FALSE,
                          shrink = 'ALPHA', makevone = FALSE)
print(est_uee_ind_bin)
```

`print.cpgeeSWD` *The print format for cpgeeSWD output*

Description

The print format for cpgeeSWD output

Usage

```
## S3 method for class 'cpgeeSWD'
print(x, ...)
```

Arguments

x	The object of cpgeeSWD output
...	further arguments passed to or from other methods

Value

The output from `print`

`print.geemaee` *The print format for geemaee output*

Description

The print format for geemaee output

Usage

```
## S3 method for class 'geemaee'
print(x, ...)
```

Arguments

x	The object of geemaee output
...	further arguments passed to or from other methods

Value

The output from `print`

`sampleSWCRTLarge` *simulated large SW-CRT data*

Description

Simulated cross-sectional individual-level SW-CRT data with 12 clusters and 5 periods. The cluster-period size is uniformly distributed between 20 and 30. The correlated binary and continuous outcomes are used for analysis as examples.

Format

A data frame with 1508 rows and 10 variables:

period1 indicator of being at period 1

period2 indicator of being at period 2

period3 indicator of being at period 3

period4 indicator of being at period 4

period5 indicator of being at period 5

treatment indicator of being treated

id cluster identification number

period period order number

y_bin binary outcome variable

y_con continous outcome variable

`sampleSWCRTSmall` *simulated small SW-CRT data*

Description

Simulated cross-sectional individual-level SW-CRT data with 12 clusters and 4 periods. The cluster-period size is uniformly distributed between 5 and 10. The correlated binary and continuous outcomes are used for analysis as examples.

Format

A data frame with 373 rows and 9 variables:

period1 indicator of being at period 1

period2 indicator of being at period 2

period3 indicator of being at period 3

period4 indicator of being at period 4

treatment indicator of being treated

id cluster identification number

period period order number

y_bin binary outcome variable

y_con continous outcome variable

simbinCLF *Generating Correlated Binary Data using the Conditional Linear Family Method.*

Description

simbinCLF generates correlated binary data using the conditional linear family method (Qaqish, 2003). It simulates a vector of binary outcomes according the specified marginal mean vector and correlation structure. Natural constraints and compatibility between the marginal mean and correlation matrix are checked.

Usage

```
simbinCLF(mu, Sigma, n = 1)
```

Arguments

mu	a mean vector when n = 1 or is NULL, otherwise a list of mean vectors for the n clusters
Sigma	a correlation matrix when n = 1 or is NULL, otherwise a list of correlation matrices for the n clusters
n	number of clusters. The default is 1

Value

y a vector of simulated binary outcomes for n clusters.

Author(s)

Hengshi Yu <hengshi@umich.edu>, Fan Li <fan.f.li@yale.edu>, Paul Rathouz <paul.rathouz@austin.utexas.edu>, Elizabeth L. Turner <liz.turner@duke.edu>, John Preisser <jpreisse@bios.unc.edu>

References

Qaqish, B. F. (2003). A family of multivariate binary distributions for simulating correlated binary variables with specified marginal means and correlations. Biometrika, 90(2), 455-463.

Preisser, J. S., Qaqish, B. F. (2014). A comparison of methods for simulating correlated binary variables with specified marginal means and correlations. Journal of Statistical Computation and Simulation, 84(11), 2441-2452.

Examples

```
#######################################################################
# Simulate 2 clusters, 3 periods and cluster-period size of 5 #######
#######################################################################
```

```
t = 3; n = 2; m = 5

# means of cluster 1
u_c1 = c(0.4, 0.3, 0.2)
u1 <- rep(u_c1, c(rep(m, t)))
# means of cluster 2
u_c2 = c(0.35, 0.25, 0.2)
u2 <- rep(u_c2, c(rep(m, t)))

# List of mean vectors
mu = list()
mu[[1]] = u1; mu[[2]] = u2;

# List of correlation matrices

## correlation parameters
alpha0 = 0.03; alpha1 = 0.015; rho = 0.8

## (1) exchangeable
Sigma = list()
Sigma[[1]] = diag(m * t) * ( 1 - alpha1) + matrix(alpha1, m * t,  m * t )
Sigma[[2]] = diag(m * t) * ( 1 - alpha1) + matrix(alpha1, m * t,  m * t )

y_exc = simbinCLF(mu = mu, Sigma = Sigma, n = n)

## (2) nested exchangeable
Sigma = list()
cor_matrix = matrix(alpha1, m * t,  m * t)
loc1 = 0; loc2 = 0
for(t in 1:t){loc1 = loc2 + 1; loc2 = loc1 + m - 1
 for(i in loc1:loc2){for(j in loc1:loc2){
        if(i != j){cor_matrix[i, j] = alpha0}else{cor_matrix[i, j] = 1}}}}

Sigma[[1]] = cor_matrix; Sigma[[2]] = cor_matrix

y_nex = simbinCLF(mu = mu, Sigma = Sigma, n = n)

## (3) exponential decay

Sigma = list()

### function to find the period of the ith index
region_ij<-function(points, i){diff = i - points
    for(h in 1:(length(diff) - 1)){if(diff[h] > 0 & diff[h + 1] <= 0){find <- h}}
 return(find)}

cor_matrix = matrix(0,  m * t,  m * t)
useage_m = cumsum(m * t); useage_m = c(0, useage_m)

for(i in 1:(m * t)){i_reg = region_ij(useage_m, i)
     for(j in 1:(m * t)){j_reg = region_ij(useage_m, j)
         if(i_reg == j_reg & i != j){
             cor_matrix[i, j] = alpha0}else if(i == j){cor_matrix[i, j] = 1
```

```
}else if(i_reg != j_reg){cor_matrix[i,j] = alpha0 * (rho^(abs(i_reg - j_reg)))}}}

 Sigma[[1]] = cor_matrix; Sigma[[2]] = cor_matrix

 y_ed = simbinCLF(mu = mu, Sigma = Sigma, n = n)
```

---

simbinPROBIT *Generating Correlated Binary Data using the Multivariate Probit Method.*

Description

simbinPROBIT generates correlated binary data using the multivariate Probit method (Emrich and Piedmonte, 1991). It simulates a vector of binary outcomes according the specified marginal mean vector and correlation structure. Constraints and compatibility between the marginal mean and correlation matrix are checked.

Usage

```
simbinPROBIT(mu, Sigma, n = 1)
```

Arguments

mu	a mean vector when `n = 1` or is `NULL`, otherwise a list of mean vectors for the `n` clusters
Sigma	a correlation matrix when `n = 1` or is `NULL`, otherwise a list of correlation matrices for the `n` clusters
n	number of clusters. The default is `1`

Value

`y` a vector of simulated binary outcomes for `n` clusters.

Author(s)

Hengshi Yu <hengshi@umich.edu>, Fan Li <fan.f.li@yale.edu>, Paul Rathouz <paul.rathouz@austin.utexas.edu>, Elizabeth L. Turner <liz.turner@duke.edu>, John Preisser <jpreisse@bios.unc.edu>

References

Emrich, L. J., & Piedmonte, M. R. (1991). A method for generating high-dimensional multivariate binary variates. The American Statistician, 45(4), 302-304.

Preisser, J. S., Qaqish, B. F. (2014). A comparison of methods for simulating correlated binary variables with specified marginal means and correlations. Journal of Statistical Computation and Simulation, 84(11), 2441-2452.

Examples

```
####################################################################
# Simulate 2 clusters, 3 periods and cluster-period size of 5 #######
####################################################################

t = 3; n = 2; m = 5

# means of cluster 1
u_c1 = c(0.4, 0.3, 0.2)
u1 <- rep(u_c1, c(rep(m, t)))
# means of cluster 2
u_c2 = c(0.35, 0.25, 0.2)
u2 <- rep(u_c2, c(rep(m, t)))

# List of mean vectors
mu = list()
mu[[1]] = u1; mu[[2]] = u2;

# List of correlation matrices

## correlation parameters
alpha0 = 0.03; alpha1 = 0.015; rho = 0.8

## (1) exchangeable
Sigma = list()
Sigma[[1]] = diag(m * t) * ( 1 - alpha1) + matrix(alpha1, m * t,  m * t )
Sigma[[2]] = diag(m * t) * ( 1 - alpha1) + matrix(alpha1, m * t,  m * t )
y_exc = simbinPROBIT(mu = mu, Sigma = Sigma, n = n)

## (2) nested exchangeable
Sigma = list()
cor_matrix = matrix(alpha1, m * t,  m * t)
loc1 = 0; loc2 = 0
for(t in 1:t){loc1 = loc2 + 1; loc2 = loc1 + m - 1
 for(i in loc1:loc2){for(j in loc1:loc2){
        if(i != j){cor_matrix[i, j] = alpha0}else{cor_matrix[i, j] = 1}}}}

Sigma[[1]] = cor_matrix; Sigma[[2]] = cor_matrix
y_nex = simbinPROBIT(mu = mu, Sigma = Sigma, n = n)

## (3) exponential decay

Sigma = list()

### function to find the period of the ith index
region_ij<-function(points, i){diff = i - points
    for(h in 1:(length(diff) - 1)){if(diff[h] > 0 & diff[h + 1] <= 0){find <- h}}
 return(find)}

cor_matrix = matrix(0,  m * t,  m * t)
useage_m = cumsum(m * t); useage_m = c(0, useage_m)
```

```
for(i in 1:(m * t)){i_reg = region_ij(useage_m, i)
     for(j in 1:(m * t)){j_reg = region_ij(useage_m, j)
         if(i_reg == j_reg & i != j){
             cor_matrix[i, j] = alpha0}else if(i == j){cor_matrix[i, j] = 1
}else if(i_reg != j_reg){cor_matrix[i,j] = alpha0 * (rho^(abs(i_reg - j_reg)))}}}
 Sigma[[1]] = cor_matrix; Sigma[[2]] = cor_matrix
 y_ed = simbinPROBIT(mu = mu, Sigma = Sigma, n = n)
```

Index