Package 'BayesRGMM'

January 20, 2025

Type Package

Title Bayesian Robust Generalized Mixed Models for Longitudinal Data

Version 2.2

Depends R (>= 3.5.0)

Date 2022-05-06

Description To perform model estimation using MCMC algorithms with Bayesian methods for incomplete longitudinal studies on binary and ordinal outcomes that are measured repeatedly on subjects over time with drop-outs. Details about the method can be found in the vignette or <https://sites.google.com/view/kuojunglee/r-packages/bayesrgmm>.

License GPL-2

URL https://sites.google.com/view/kuojunglee/r-packages

Encoding UTF-8

Imports Rcpp (>= 1.0.1), MASS, batchmeans, abind, reshape, msm, mvtnorm, plyr, Rdpack

RdMacros Rdpack

LinkingTo Rcpp, RcppArmadillo, RcppDist

Suggests testthat

RoxygenNote 7.1.2

NeedsCompilation yes

Author Kuo-Jung Lee [aut, cre] (<https://orcid.org/0000-0002-7388-4738>),
Hsing-Ming Chang [ctb],
Ray-Bing Chen [ctb],
Keunbaik Lee [ctb],
Chanmin Kim [ctb]

Maintainer Kuo-Jung Lee <kuojunglee@ncku.edu.tw>

Repository CRAN

Date/Publication 2022-05-10 12:30:13 UTC

Contents

`AR1.cor` *AR(1) correlation matrix*

Description

Generate a correlation matrix for AR(1) model

Usage

```
AR1.cor(n, rho)
```

Arguments

`n`	size of matrix
`rho`	correlation between -1 to 1

Value

$n \times n$ AR(1) correlation matrix

Details

The correlation matrix is created as

$$\begin{pmatrix} 1 & \rho & \rho^2 & \cdots & \rho^{n-1} \\ \rho & 1 & \rho & \cdots & \rho^{n-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \rho^2 & \rho & 1 & \cdots & \rho^{n-3} \end{pmatrix}$$

Examples

```
AR1.cor(5, 0.5)
```

BayesCumulativeProbitHSD

Perform MCMC algorithm to generate the posterior samples for longitudinal ordinal data

Description

This function is used to generate the posterior samples using MCMC algorithm from the cumulative probit model with the hypersphere decomposition applied to model the correlation structure in the serial dependence of repeated responses.

Usage

```
BayesCumulativeProbitHSD(
  fixed,
  data,
  random,
  Robustness,
  subset,
  na.action,
  HS.model,
  hyper.params,
  num.of.iter,
  Interactive
)
```

Arguments

`fixed`	a two-sided linear formula object to describe fixed-effects with the response on the left of a '~' operator and the terms separated by '+' or '*' operators, on the right. The specification `first*second` indicates the cross of `first` and `second`. This is the same as `first + second + first:second`.
`data`	an optional data frame containing the variables named in '`fixed`' and '`random`'. It requires an "integer" variable named by '`id`' to denote the identifications of subjects.
`random`	a one-sided linear formula object to describe random-effects with the terms separated by '+' or '*' operators on the right of a '~' operator.
`Robustness`	logical. If 'TRUE' the distribution of random effects is assumed to be t-distribution; otherwise normal distribution.
`subset`	an optional expression indicating the subset of the rows of '`data`' that should be used in the fit. This can be a logical vector, or a numeric vector indicating which observation numbers are to be included, or a character vector of the row names to be included. All observations are included by default.
`na.action`	a function that indicates what should happen when the data contain NA's. The default action ('`na.omit`', inherited from the '`factory fresh`' value of

	‘getOption("na.action")’) strips any observations with any missing values in any variables.
HS.model	a specification of the correlation structure in HSD model: • HS.model = ~0 denotes independence, that is, R_i is an identity matrix, • HS.model = ~IndTime+$\cdots$+IndTimer denotes AR(r) correlation structure, • HS.model = ~DiffTime1+$\cdots$+DiffTimer denotes correlation structure related to rth order of time difference.
hyper.params	specify the values in hyperparameters in priors.
num.of.iter	an integer to specify the total number of iterations; default is 20000.
Interactive	logical. If ’TRUE’ when the program is being run interactively for progress bar and ’FALSE’ otherwise.

Value

a list of posterior samples, parameters estimates, AIC, BIC, CIC, DIC, MPL, RJR, predicted values, and the acceptance rates in MH are returned.

Note

Only a model either HSD (‘HS.model’) or ARMA (‘arma.order’) model should be specified in the function. We’ll provide the reference for details of the model and the algorithm for performing model estimation whenever the manuscript is accepted.

Author(s)

Kuo-Jung Lee kuojunglee@ncku.edu.tw

References

Lee K, Chen R, Kwak M, Lee K (2021). “Determination of correlations in multivariate longitudinal data with modified Cholesky and hypersphere decomposition using Bayesian variable selection approach.” *Statistics in Medicine*, **40**(4), 978–997.

Lee K, Cho H, Kwak M, Jang EJ (2020). “Estimation of covariance matrix of multivariate longitudinal data using modified Choleksky and hypersphere decompositions.” *Biometrics*, **76**(1), 75–86.

Examples

```
## Not run:
library(BayesRGMM)
rm(list=ls(all=TRUE))

Fixed.Effs = c(-0.1, 0.1, -0.1) #c(-0.8, -0.3, 1.8, -0.4)
P = length(Fixed.Effs)
q = 1 #number of random effects
T = 7 #time points
N = 100 #number of subjects
Num.of.Cats = 3 #in KBLEE simulation studies, please fix it.
num.of.iter = 1000 #number of iterations
```

```
HSD.para = c(-0.9, -0.6) #the parameters in HSD model
a = length(HSD.para)
w = array(runif(T*T*a), c(T, T, a)) #design matrix in HSD model

for(time.diff in 1:a)
w[, , time.diff] = 1*(as.matrix(dist(1:T, 1:T, method="manhattan")) ==time.diff)

x = array(0, c(T, P, N))
for(i in 1:N){
    #x[,, i] = t(rmvnorm(P, rep(0, T), AR1.cor(T, Cor.in.DesignMat)))
    x[, 1, i] = 1:T
    x[, 2, i] = rbinom(1, 1, 0.5)
    x[, 3, i] = x[, 1, i]*x[, 2, i]
}

DesignMat = x

#Generate a data with HSD model


#MAR
CPREM.sim.data = SimulatedDataGenerator.CumulativeProbit(
 Num.of.Obs = N, Num.of.TimePoints = T, Num.of.Cats = Num.of.Cats,
 Fixed.Effs = Fixed.Effs, Random.Effs = list(Sigma = 0.5*diag(1), df=3),
 DesignMat = DesignMat, Missing = list(Missing.Mechanism = 2,
 MissingRegCoefs=c(-0.7, -0.2, -0.1)),
 HSD.DesignMat.para = list(HSD.para = HSD.para, DesignMat = w))

print(table(CPREM.sim.data$sim.data$y))
print(CPREM.sim.data$classes)

BCP.output = BayesCumulativeProbitHSD(
    fixed = as.formula(paste("y~", paste0("x", 1:P, collapse="+"))),
    data=CPREM.sim.data$sim.data, random = ~ 1, Robustness = TRUE,
    subset = NULL, na.action='na.exclude', HS.model = ~IndTime1+IndTime2,
    hyper.params=NULL, num.of.iter=num.of.iter, Interactive=0)

BCP.Est.output = BayesRobustProbitSummary(BCP.output)

## End(Not run)
```

`BayesRobustProbit` *Perform MCMC algorithm to generate the posterior samples*

Description

This function is used to generate the posterior samples using MCMC algorithm from the probit model with either the hypersphere decomposition or ARMA models applied to model the correlation structure in the serial dependence of repeated responses.

Usage

```
BayesRobustProbit(
  fixed,
  data,
  random,
  Robustness = TRUE,
  subset = NULL,
  na.action = "na.exclude",
  arma.order = NULL,
  HS.model = NULL,
  hyper.params = NULL,
  num.of.iter = 20000,
  Interactive = FALSE
)
```

Arguments

`fixed`	a two-sided linear formula object to describe fixed-effects with the response on the left of a ‘`~`’ operator and the terms separated by ‘`+`’ or ‘`*`’ operators, on the right. The specification `first*second` indicates the cross of `first` and `second`. This is the same as `first + second + first:second`.
`data`	an optional data frame containing the variables named in ‘`fixed`’ and ‘`random`’. It requires an “integer” variable named by ‘`id`’ to denote the identifications of subjects.
`random`	a one-sided linear formula object to describe random-effects with the terms separated by ‘`+`’ or ‘`*`’ operators on the right of a ‘`~`’ operator.
`Robustness`	logical. If ’TRUE’ the distribution of random effects is assumed to be t-distribution; otherwise normal distribution.
`subset`	an optional expression indicating the subset of the rows of ‘`data`’ that should be used in the fit. This can be a logical vector, or a numeric vector indicating which observation numbers are to be included, or a character vector of the row names to be included. All observations are included by default.
`na.action`	a function that indicates what should happen when the data contain NA’s. The default action (‘`na.omit`’, inherited from the ‘`factory fresh`’ value of ‘`getOption("na.action")`’) strips any observations with any missing values in any variables.
`arma.order`	a specification of the order in an ARMA model: the two integer components (p, q) are the AR order and the MA order.
`HS.model`	a specification of the correlation structure in HSD model: • `HS.model = ~0` denotes independence, that is, R_i is an identity matrix, • `HS.model = ~IndTime+`$\cdots$`+IndTimer` denotes AR(r) correlation structure, • `HS.model = ~DiffTime1+`$\cdots$`+DiffTimer` denotes correlation structure related to rth order of time difference.
`hyper.params`	specify the values in hyperparameters in priors.
`num.of.iter`	an integer to specify the total number of iterations; default is 20000.

`Interactive`	logical. If 'TRUE' when the program is being run interactively for progress bar and 'FALSE' otherwise.

Value

a list of posterior samples, parameters estimates, AIC, BIC, CIC, DIC, MPL, RJR, predicted values, and the acceptance rates in MH are returned.

Note

Only a model either HSD ('`HS.model`') or ARMA ('`arma.order`') model should be specified in the function. We'll provide the reference for details of the model and the algorithm for performing model estimation whenever the manuscript is accepted.

Author(s)

Kuo-Jung Lee kuojunglee@ncku.edu.tw

References

Lee K, Chen R, Kwak M, Lee K (2021). "Determination of correlations in multivariate longitudinal data with modified Cholesky and hypersphere decomposition using Bayesian variable selection approach." *Statistics in Medicine*, **40**(4), 978–997.

Lee K, Cho H, Kwak M, Jang EJ (2020). "Estimation of covariance matrix of multivariate longitudinal data using modified Choleksky and hypersphere decompositions." *Biometrics*, **76**(1), 75–86.

Examples

```
## Not run:
library(BayesRGMM)
rm(list=ls(all=TRUE))
Fixed.Effs = c(-0.2, -0.3, 0.8, -0.4) #c(-0.2,-0.8, 1.0, -1.2)
P = length(Fixed.Effs)
q = 1 #number of random effects
T = 5 #time points
N = 100 #number of subjects
num.of.iter = 100 #number of iterations
HSD.para = c(-0.5,  -0.3) #the parameters in HSD model
a = length(HSD.para)
w = array(runif(T*T*a), c(T, T, a)) #design matrix in HSD model

for(time.diff in 1:a)
w[, , time.diff] = 1*(as.matrix(dist(1:T, 1:T, method="manhattan"))
 ==time.diff)

#Generate a data with HSD model
 HSD.sim.data = SimulatedDataGenerator(
 Num.of.Obs = N, Num.of.TimePoints = T, Fixed.Effs = Fixed.Effs,
 Random.Effs = list(Sigma = 0.5*diag(1), df=3),
Cor.in.DesignMat = 0., Missing = list(Missing.Mechanism = 2,
 RegCoefs = c(-1.5, 1.2)), Cor.Str = "HSD",
 HSD.DesignMat.para = list(HSD.para = HSD.para, DesignMat = w))

hyper.params = list(
        sigma2.beta = 1,
        sigma2.delta = 1,
        v.gamma = 5,
        InvWishart.df = 5,
        InvWishart.Lambda = diag(q) )

HSD.output = BayesRobustProbit(
fixed = as.formula(paste("y~-1+", paste0("x", 1:P, collapse="+"))),
data=HSD.sim.data$sim.data, random = ~ 1, Robustness=TRUE,
HS.model = ~IndTime1+IndTime2, subset = NULL, na.action='na.exclude',
hyper.params = hyper.params, num.of.iter = num.of.iter,
 Interactive=0)

## End(Not run)
```

BayesRobustProbitSummary

To summarizes model estimation outcomes

Description

It provides basic posterior summary statistics such as the posterior point and confidence interval estimates of parameters and the values of information criterion statistics for model comparison.

Usage

```
BayesRobustProbitSummary(object, digits = max(1L, getOption("digits") - 4L))
```

Arguments

`object`	output from the function `BayesRobustProbit`.
`digits`	rounds the values in its first argument to the specified number of significant digits.

Value

a list of posterior summary statistics and corresponding model information

Examples

```
## Not run:
library(BayesRGMM)
rm(list=ls(all=TRUE))
Fixed.Effs = c(-0.2, -0.3, 0.8, -0.4) #c(-0.2,-0.8, 1.0, -1.2)
P = length(Fixed.Effs)
q = 1 #number of random effects
```

```
T = 5 #time points
N = 100 #number of subjects
num.of.iter = 100 #number of iterations
HSD.para = c(-0.5,  -0.3) #the parameters in HSD model
a = length(HSD.para)
w = array(runif(T*T*a), c(T, T, a)) #design matrix in HSD model

for(time.diff in 1:a)
w[, , time.diff] = 1*(as.matrix(dist(1:T, 1:T, method="manhattan"))
 == time.diff)

#Generate a data with HSD model
HSD.sim.data = SimulatedDataGenerator(Num.of.Obs = N, Num.of.TimePoints = T,
Fixed.Effs = Fixed.Effs, Random.Effs = list(Sigma = 0.5*diag(1), df=3),
Cor.in.DesignMat = 0., Missing = list(Missing.Mechanism = 2,
 RegCoefs = c(-1.5, 1.2)), Cor.Str = "HSD",
HSD.DesignMat.para = list(HSD.para = HSD.para, DesignMat = w))

hyper.params = list(
        sigma2.beta = 1,
        sigma2.delta = 1,
        v.gamma = 5,
        InvWishart.df = 5,
        InvWishart.Lambda = diag(q) )

HSD.output = BayesRobustProbit(
 fixed = as.formula(paste("y~-1+", paste0("x", 1:P, collapse="+"))),
data=HSD.sim.data$sim.data, random = ~ 1, Robustness=TRUE,
 HS.model = ~IndTime1+IndTime2, subset = NULL, na.action='na.exclude',
 hyper.params = hyper.params, num.of.iter = num.of.iter, Interactive =0)

BayesRobustProbitSummary(HSD.output)

## End(Not run)
```

`CorrMat.HSD` *To compute the correlation matrix in terms of hypersphere decomposition approach*

Description

The correlation matrix is reparameterized via hyperspherical coordinates angle parameters for trigonometric functions, and the angle parameters are referred to hypersphere (HS) parameters. In order to obtain the unconstrained estimation of angle parameters and to reduce the number of parameters for facilitating the computation, we model the correlation structures of the responses in terms of the generalized linear models

Usage

```
CorrMat.HSD(w, delta)
```

Arguments

w	a design matrix is used to model the HS parameters as functions of subject-specific covariates.
delta	an $a \times 1$ vector of unknown parameters to model the HS parameters.

Value

a correlation matrix

Author(s)

Kuo-Jung Lee kuojunglee@ncku.edu.tw

References

Zhang W, Leng C, Tang CY (2015). “A joint modelling approach for longitudinal studies.” *Journal of Royal Statistical Society, Series B*, **77**, 219–238.

Examples

```
## Not run:
library(BayesRGMM)
rm(list=ls(all=TRUE))
T = 5 #time points
HSD.para = c(-0.5,  -0.3) #the parameters in HSD model
a = length(HSD.para)
w = array(runif(T*T*a), c(T, T, a)) #design matrix in HSD model
signif(CorrMat.HSD(w, HSD.para), 4)

## End(Not run)
```

GSPS *The German socioeconomic panel study data*

Description

The German socioeconomic panel study data was taken from the first twelve annual waves (1984 through 1995) of the German Socioeconomic Panel (GSOEP) which surveys a representative sample of East and West German households. The data provide detailed information on the utilization of health care facilities, characteristics of current employment, and the insurance schemes under which individuals are covered. We consider the sample of individuals aged 25 through 65 from the West German subsample and of German nationality. The sample contained 3691 male and 3689 female individuals which make up a sample of 14,243 male and 13,794 female person-year observations.

Usage

```
data(GSPS)
```

Format

A data frame with 27326 rows and 25 variables

id person - identification number

female female = 1; male = 0

year calendar year of the observation

age age in years

hsat health satisfaction, coded 0 (low) - 10 (high)

handdum handicapped = 1; otherwise = 0

handper degree of handicap in percent (0 - 100)

hhninc household nominal monthly net income in German marks / 1000

hhkids children under age 16 in the household = 1; otherwise = 0

educ years of schooling

married married = 1; otherwise = 0

haupts highest schooling degree is Hauptschul degree = 1; otherwise = 0

reals highest schooling degree is Realschul degree = 1; otherwise = 0

fachhs highest schooling degree is Polytechnical degree = 1; otherwise = 0

abitur highest schooling degree is Abitur = 1; otherwise = 0

univ highest schooling degree is university degree = 1; otherwise = 0

working employed = 1; otherwise = 0

bluec blue collar employee = 1; otherwise = 0

whitec white collar employee = 1; otherwise = 0

self self employed = 1; otherwise = 0

beamt civil servant = 1; otherwise = 0

docvis number of doctor visits in last three months

hospvis number of hospital visits in last calendar year

public insured in public health insurance = 1; otherwise = 0

addon insured by add-on insurance = 1; otherswise = 0

Source

JAE Archive

References

Riphahn RT, Wambach A, Million A (2003). "Incentive effects in the demand for health care: a bivariate panel count data estimation." *Journal of Applied Econometrics*, **18**(4), 387–405.

SimulatedDataGenerator

Generate simulated data with either ARMA or MCD correlation structures.

Description

This function is used to generate simulated data for simulation studies with ARMA and MCD correlation structures.

Usage

```
SimulatedDataGenerator(
  Num.of.Obs,
  Num.of.TimePoints,
  Fixed.Effs,
  Random.Effs,
  Cor.in.DesignMat,
  Missing,
  Cor.Str,
  HSD.DesignMat.para,
  ARMA.para
)
```

Arguments

`Num.of.Obs`	the number of subjects will be simulated.
`Num.of.TimePoints`	the maximum number of time points among all subjects.
`Fixed.Effs`	a vector of regression coefficients.
`Random.Effs`	a list of covariance matrix and the degree of freedom, e.g., `list(Sigma = 0.5*diag(1), df=3)`.
`Cor.in.DesignMat`	the correlation of covariates in the design matrix.
`Missing`	a list of the missing mechanism of observations, 0: data is complete, 1: missing complete at random, 2: missing at random related to responses , and 3: 2: missing at random related to covariates and the corresponding regression coefficients (weights) on the previous observed values either responses or covariates, e.g., `Missing = list(Missing.Mechanism = 3, RegCoefs = c(0.4, 1.2, -2.1))`.
`Cor.Str`	the model for correlation structure, `ARMA'' or` HSD".
`HSD.DesignMat.para`	if `Cor.Str="HSD"`, you need to specify the list of parameters in HSD correlation structure, e.g., `HSD.DesignMat.para = list(HSD.para = HSD.para, DesignMat = w)`.
`ARMA.para`	if `Cor.Str="ARMA"`, you need to specify the list of parameters in AMRA correlation structure, e.g., `ARMA.para = list(AR.para=0.1, MA.para=0.2)`.

Value

a list containing the following components:

sim.data The simulated response variables y, covariates x's, and subject identifcation '`id`'.

beta.true The given values of fixed effects .

ARMA.para The given values of parameters in ARMA model.

HSD.para The given values of parameters in ARMA model.

Examples

```
## Not run:
library(BayesRGMM)
rm(list=ls(all=TRUE))
Fixed.Effs = c(-0.2, -0.3, 0.8, -0.4)
P = length(Fixed.Effs)
q = 1 #number of random effects
T = 5 #time points
N = 100 #number of subjects
num.of.iter = 100 #number of iterations
HSD.para = c(-0.5,  -0.3) #the parameters in HSD model
a = length(HSD.para)
w = array(runif(T*T*a), c(T, T, a)) #design matrix in HSD model

for(time.diff in 1:a)
w[, , time.diff] = 1*(as.matrix(dist(1:T, 1:T, method="manhattan"))
 ==time.diff)

#Generate a data with HSD model
HSD.sim.data = SimulatedDataGenerator(Num.of.Obs = N, Num.of.TimePoints = T,
Fixed.Effs = Fixed.Effs, Random.Effs = list(Sigma = 0.5*diag(1), df=3),
Cor.in.DesignMat = 0., Missing = list(Missing.Mechanism = 2,
 RegCoefs = c(-1.5, 1.2)), Cor.Str = "HSD",
 HSD.DesignMat.para = list(HSD.para = HSD.para, DesignMat = w))

#the proportion of 1's
ones = sum(HSD.sim.data$sim.data$y==1, na.rm=T)
num.of.obs = sum(!is.na(HSD.sim.data$sim.data$y))
print(ones/num.of.obs)

#the missing rate in the simulated data
print(sum(is.na(HSD.sim.data$sim.data$y)))


#=============================================================================#
#Generate a data with ARMA model
ARMA.sim.data = SimulatedDataGenerator(Num.of.Obs = N, Num.of.TimePoints = T,
Fixed.Effs = Fixed.Effs, Random.Effs = list(Sigma = 0.5*diag(1), df=3),
Cor.in.DesignMat = 0., list(Missing.Mechanism = 2, RegCoefs = c(-1.5, 1.2)),
Cor.Str = "ARMA", ARMA.para=list(AR.para = 0.8))

## End(Not run)
```

SimulatedDataGenerator.CumulativeProbit

Simulating a longitudinal ordinal data with HSD correlation structures.

Description

This function is used to simulate data for the cumulative probit mixed-effects model with HSD correlation structures.

Usage

```
SimulatedDataGenerator.CumulativeProbit(
  Num.of.Obs,
  Num.of.TimePoints,
  Num.of.Cats,
  Fixed.Effs,
  Random.Effs,
  DesignMat,
  Missing,
  HSD.DesignMat.para
)
```

Arguments

`Num.of.Obs`	the number of subjects will be simulated.
`Num.of.TimePoints`	the maximum number of time points among all subjects.
`Num.of.Cats`	the number of categories.
`Fixed.Effs`	a vector of regression coefficients.
`Random.Effs`	a list of covariance matrix and the degree of freedom, e.g., `list(Sigma = 0.5*diag(1), df=3)`.
`DesignMat`	a design matrix.
`Missing`	a list of the missing mechanism of observations, 0: data is complete, 1: missing complete at random, 2: missing at random related to responses , and 3: 2: missing at random related to covariates and the corresponding regression coefficients (weights) on the previous observed values either responses or covariates, e.g., `Missing = list(Missing.Mechanism = 3, RegCoefs = c(0.4, 1.2, -2.1))`.
`HSD.DesignMat.para`	the list of parameters in HSD correlation structure, e.g., `HSD.DesignMat.para = list(HSD.para = HSD.para, DesignMat = w)`.

Value

a list containing the following components:

sim.data The simulated response variables y, covariates x's, and subject identifcation 'id'.

beta.true The given values of fixed effects.

classes The intervals of classes.

HSD.para The given values of parameters in HSD model.

Examples

```
## Not run:
library(BayesRGMM)
rm(list=ls(all=TRUE))

Fixed.Effs = c(-0.1, 0.1, -0.1)
P = length(Fixed.Effs)
q = 1 #number of random effects
T = 7 #time points
N = 100 #number of subjects
Num.of.Cats = 3 #number of categories
num.of.iter = 1000 #number of iterations

HSD.para = c(-0.9, -0.6) #the parameters in HSD model
a = length(HSD.para)
w = array(runif(T*T*a), c(T, T, a)) #design matrix in HSD model

for(time.diff in 1:a)
w[, , time.diff] = 1*(as.matrix(dist(1:T, 1:T, method="manhattan"))
==time.diff)


x = array(0, c(T, P, N))
for(i in 1:N){
    x[, 1, i] = 1:T
    x[, 2, i] = rbinom(1, 1, 0.5)
    x[, 3, i] = x[, 1, i]*x[, 2, i]
}

DesignMat = x

#MAR
CPREM.sim.data = SimulatedDataGenerator.CumulativeProbit(
 Num.of.Obs = N, Num.of.TimePoints = T, Num.of.Cats = Num.of.Cats,
 Fixed.Effs = Fixed.Effs, Random.Effs = list(Sigma = 0.5*diag(1), df=3),
 DesignMat = DesignMat, Missing = list(Missing.Mechanism = 2,
 MissingRegCoefs=c(-0.7, -0.2, -0.1)),
 HSD.DesignMat.para = list(HSD.para = HSD.para, DesignMat = w))


print(table(CPREM.sim.data$sim.data$y))
print(CPREM.sim.data$classes)
```

```
## End(Not run)
```

Index