Package 'hyreg2'

May 8, 2026

Type Package

Title Estimate Latent Classes on a Mixture of Continuous and Dichotomous Data

Version 1.1.2

Description The hybrid model likelihood as described by Ramos-Goñi et al. (2017) <doi:10.1097/MLR.0000000000000283> is implemented and and embedded in a latent class framework. The package is based on 'flexmix' and among others contains an M-step-driver as described by Leisch (2004) <doi:10.18637/jss.v011.i08>. Users can, for example, estimate latent classes for EQ-5D value sets and address preference heterogeneity. Both uncensored and censored data are supported.
Furthermore, heteroscedasticity can be taken into account. It is possible to control for different covariates on the continuous and dichotomous data and start values can differ between the expected latent classes.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 7.3.1

Imports flexmix, bbmle, ggplot2, methods, stats, utils

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

Depends R (>= 2.10)

VignetteBuilder knitr

NeedsCompilation no

Author Svenja Elkenkamp [aut, cre],
Kim Rand [aut],
John Grosser [aut],
EuroQol [fnd]

Maintainer Svenja Elkenkamp <svenja.elkenkamp@uni-bielefeld.de>

Repository CRAN

Date/Publication 2026-03-17 12:40:09 UTC

Contents

`FLXMRhyreg` *M-step driver to be used in flexmix*

Description

Function used in flexmix M-Step to estimate hybrid model

Usage

```
FLXMRhyreg(
  formula = . ~ .,
  family = c("hyreg"),
  type = NULL,
  type_cont = NULL,
  type_dich = NULL,
  variables_both = NULL,
  variables_cont = NULL,
  variables_dich = NULL,
  stv = NULL,
  offset = NULL,
  opt_method = "BFGS",
  optimizer = "optim",
  lower = -Inf,
  upper = Inf,
  formula_type_classic = TRUE,
  ...
)
```

Arguments

formula	model formula, automatically provided by hyreg2 and flexmix::flexmix()
family	default "hyreg", needed for flexmix::flexmix()
type	character vector containing the indicator whether that datapoint (row) contains continuous or dichotomous data, see Details of hyreg2
type_cont	value of type referring to continuous data, see Details of hyreg2
type_dich	value of type referring to dichotomous data, see Details of hyreg2
variables_both	character vector; variables to be fitted on both continuous and dichotomous data. see Details of hyreg2
variables_cont	character vector; variables to be fitted only on continous data. see Details of hyreg2
variables_dich	character vactor; variables to be fitted only on dichotomous data. see Details of hyreg2
stv	named vector or list of named vectors containing start values for all coefficients from formula, including theta, see Details of hyreg2
offset	offset as in flexmix::flexmix(), default NULL
opt_method	character, optimization method to be used in optimizer, default "BFGS"
optimizer	character, optimizer to be used in bbmle::mle2(), default "optim"
lower	lower bound for censored data. If this is used, opt_method must be set to "L-BFGS-B", default -INF,
upper	upper bound for censored data. If this is used, opt_method must be set to "L-BFGS-B",default INF
formula_type_classic	logical; is the provided formula a typical R formula containing only variables or does it include variables and parameters? default TRUE
...	additional arguments for flexmix::flexmix() or bbmle::mle2()

Value

a model object, that can be used in hyreg2 as input for parameter model in flexmix::flexmix()

a model object, that can be used in hyreg2 as input for parameter model in flexmix::flexmix

Author(s)

Svenja Elkenkamp and Kim Rand

Examples

```
formula <- y ~  -1 + x1 + x2 + x3
the$k <- 2
stv <- setNames(c(0.2,0,1,1,1),c(colnames(simulated_data_norm)[3:5],c("sigma","theta")))
```

```
x <- model.matrix(formula,simulated_data_norm)
y <- simulated_data_norm$y
w <- 1
model <- FLXMRhyreg(formula = formula,
                    family=c("hyreg"),
                    type =  simulated_data_norm$type,
                    stv = stv,
                    type_cont = "TTO",
                    type_dich = "DCE_A",
                    opt_method = "L-BFGS-B",
                    control = list(iter.max = 1000, verbose = 4),
                    offset = NULL,
                    optimizer = "optim",
                variables_both =  names(stv)[!is.element(names(stv),c("sigma","theta"))],
                    variables_cont = NULL,
                    variables_dich = NULL,
                    lower = -Inf,
                    upper = Inf,
)
```

FLXMRhyreg_het *M-step driver to be used in flexmix accounting for heteroscedastisity*

Description

Function used in flexmix M-Step to estimate hybrid model accounting for heteroscedastisity

Usage

```
FLXMRhyreg_het(
  data,
  formula = . ~ .,
  formula_sigma = formula_sigma,
  family = c("hyreg"),
  type = NULL,
  type_cont = NULL,
  type_dich = NULL,
  variables_both = NULL,
  variables_cont = NULL,
  variables_dich = NULL,
  stv = NULL,
  stv_sigma = NULL,
  offset = NULL,
  opt_method = "BFGS",
  optimizer = "optim",
  lower = -Inf,
  upper = Inf,
  ...
)
```

Arguments

data	a data.frame containing the data, see Details of hyreg2_het
formula	linear model formula
formula_sigma	formula for estimation of sigma to account for heteroscedasticity, see Details hyreg2_het
family	default "hyreg", needed for flexmix::flexmix()
type	character vector containing the indicator whether that datapoint (row) contains continuous or dichotomous data, see Details of hyreg2_het
type_cont	value of type referring to continuous data, see Details of hyreg2_het
type_dich	value of type referring to dichotomous data, see Details of hyreg2_het
variables_both	character vector; variables to be fitted on both continuous and dichotomous data. see Details of hyreg2_het
variables_cont	character vector; variables to be fitted only on continous data. see Details of hyreg2_het
variables_dich	character vactor; variables to be fitted only on dichotomous data. see Details of hyreg2_het
stv	named vector or list of named vectors containing start values for all coefficients from formula, including theta, see Details of hyreg2_het
stv_sigma	named vector with start values for sigma estimation. Have to be the same variables as given in formula_sigma, see Details of hyreg2_het
offset	offset as in flexmix::flexmix(), default NULL
opt_method	character, optimization method to be used in optimizer, default "BFGS"
optimizer	character, optimizer to be used in bbmle::mle2(), default "optim"
lower	lower bound for censored data. If this is used, opt_method must be set to "L-BFGS-B", default -INF,
upper	upper bound for censored data. If this is used, opt_method must be set to "L-BFGS-B",default INF
...	additional arguments for flexmix::flexmix() or bbmle::mle2()

Value

a model object, that can be used in hyreg2_het as input for parameter model in flexmix::flexmix()

Author(s)

Svenja Elkenkamp and Kim Rand

Examples

```
formula <- y ~  -1 + x1 + x2 + x3
formula_sigma <- y ~ x1 + x2 + x3
stv <- setNames(c(0.2,0,1,1),c(colnames(simulated_data_norm)[3:5],c("theta")))
stv_sigma <- setNames(c(0.2,0.2,0.1,1),c(colnames(simulated_data_norm)[3:5],c("(Intercept)")))
```

```
x <- model.matrix(formula,simulated_data_norm)
y <- simulated_data_norm$y
w <- 1
model <- FLXMRhyreg_het( data = simulated_data_norm,
                     formula = formula,
                     formula_sigma = formula_sigma,
                    family=c("hyreg"),
                    type =  simulated_data_norm$type,
                    stv = stv,
                    stv_sigma = stv_sigma,
                    type_cont = "TTO",
                    type_dich = "DCE_A",
                    opt_method = "L-BFGS-B",
                    control = list(iter.max = 1000, verbose = 4),
                    offset = NULL,
                    optimizer = "optim",
                    variables_both =  names(stv)[!is.element(names(stv),c("theta"))],
                    variables_cont = NULL,
                    variables_dich = NULL,
                    lower = -Inf,
                    upper = Inf,
)
```

`get_stv` *extract parameter estimates as named vector*

Description

function to export coefficient values and names from a `model` fitted by `hyreg2` or `hyreg2_het` These values can be used as `stv` for a new model with k > 1

Usage

```
get_stv(mod, comp = "Comp.1")
```

Arguments

`mod`	`model`output from hyreg2 oder hyreg2_het. If `latent` was `"cont"` or `"dich"` only one element of the output list can be used.
`comp`	`charachter`, default `"Comp.1"`. `"Comp.x"` indicating values from which model `component` (x) should be exported

Value

`named vector` of parameter estimates from `mod`. Can be used as `stv` for additional model estimations using `hyreg2` or `hyreg2_het`

Author(s)

Svenja Elkenkamp

Examples

```
formula <- y ~  -1 + x1 + x2 + x3 | id

k <- 1
stv <- setNames(c(0.2,0,1,1,1),c(colnames(simulated_data_norm)[3:5],c("sigma","theta")))
control = list(iter.max = 1000, verbose = 4)
rm(counter)
mod <- hyreg2(formula = formula,
                    data =  simulated_data_norm,
                    type =  simulated_data_norm$type,
                    stv = stv,
                    k = k,
                    type_cont = "TTO",
                    type_dich = "DCE_A",
                    opt_method = "L-BFGS-B",
                    control = control,
                    latent = "both",
                    id_col = "id"
)
new_stv <- get_stv(mod)

# these new_stv can be used in an other estimation using hyreg2 as stv
```

---

give_class *decode classes by the model*

Description

This function can be used to decode the classified classes by the model generated using `hyreg2` or `hyreg2_het`

Usage

```
give_class(data, model, id_col = NULL)
```

Arguments

`data`	a `dataframe`, which was used to estimate the `model`
`model`	a flexmix `model`object estimated using `hyreg2()` or `hyreg2_het()`
`id_col`	`character`-string, name of grouping variable, which must be a column of the provided `data`. the parameter must be specified, if the provided `model` was estimated under control for `groups`

Value

dataframe of two columns, first column named as provided id_col or "observation" if id_col was not given as an input. second column named "mod_comp" indicating the assigned class for this group or observation

Author(s)

Svenja Elkenkamp & John Grosser

Examples

```
# estimate a model using simulated_data_norm

### using grouping variable id ####
formula <- y ~  -1 + x1 + x2 + x3 | id
k <- 1
stv <- setNames(c(0.2,0.2,0.2,1,1),c(colnames(simulated_data_norm)[3:5],c("sigma","theta")))
control <- list(iter.max = 1000, verbose = 4)

hyflex_mod <- hyreg2(formula = formula,
                     data =  simulated_data_norm,
                     type =  simulated_data_norm$type,
                     stv = stv,
                     k = k,
                     type_cont = "TTO",
                     type_dich = "DCE_A",
                     opt_method = "L-BFGS-B",
                     control = control,
                     latent = "both",
                     id_col = "id"
)
# use of function give_class
give_class(data = simulated_data_norm,
model = hyflex_mod,
id_col = "id")
```

---

hyreg2 *Estimating hybrid models*

Description

Estimation of hybrid model using continuous and dichotomous data e.g. EQ-5D data

Usage

```
hyreg2(
  formula,
  data,
  type,
  type_cont,
  type_dich,
  k = 1,
  control = NULL,
  stv = NULL,
  offset = NULL,
  opt_method = "BFGS",
  optimizer = "optim",
  lower = -Inf,
  upper = Inf,
  latent = "both",
  id_col = NULL,
  classes_only = FALSE,
  variables_both = NULL,
  variables_dich = NULL,
  variables_cont = NULL,
  formula_type_classic = TRUE,
  ...
)
```

Arguments

formula	model formula, can be linear or non-linear. For non-linear formulas, variables and parameters must be provided and formula_type_classic must be set to FALSE. Using |xg will include a grouping variable xg. see Details.
data	a data.frame containing the data. see Details.
type	either the name of the column in data containing an indicator of whether an observation is continuous or dichotomous (as character), or a vector containing the indicator. see Details.
type_cont	value of type referring to continuous data. see Details.
type_dich	Value of type referring to dichotomous data. see Details.
k	numeric. Number of latent classes to be estimated via flexmix::flexmix().
control	control list for flexmix::flexmix().
stv	named vector or list of named vectors containing start values for all coefficients formula, including sigma and theta, see Details
offset	offset as in flexmix::flexmix()
opt_method	charachter, optimization method to be used in optimizer, default "BFGS"
optimizer	charachter,optimizer to be used in bbmle::mle2(), default "optim"
lower	numeric, lower bound for censored data, default -INF. If this is used, opt_method must be set to "L-BFGS-B",

upper	numeric, upper bound for censored data, default INF. If this is used, opt_method must be set to "L-BFGS-B",
latent	charachter,data type to use in component identification, must be one of "both", "cont" or "dich", default "both", see Details
id_col	character, name of the grouping variable, only needed if latent != "both", see Details
classes_only	logical, default FALSE, indicates whether the function should perform only classification, rather than both classification and model estimation, only possible for latent != "both", see Datails
variables_both	character vector; variables to be fitted on both continuous and dichotomous data. If not specified, all variables from formula are used. If provided and not all variables from formula are included, variables_cont and variables_dich must be provided as well, while one of them can be NULL, see Details.
variables_dich	character vector; variables to be fitted only on dichotomous data, if provided, variables_both and variables_cont must be provided as well.
variables_cont	character vector; variables to be fitted only on continous data. If provided, variables_both and variables_dich must be provided as well.
formula_type_classic	logical; is the provided formula a classic R formula containing only variables (TRUE) or does it include both variables and parameters (FALSE)? default TRUE, see Details
...	additional arguments for flexmix::flexmix() or bbmle::mle2()

Details

see details of different inputs listed below.

Value

model object of type flexmix or list of model objects of type flexmix. Please note, that the estimates for sigma and theta are on a log-scale and have to be transformed using exp()to get the correct estimated values.

formula

a classic R formula containing only variables(e.g.y ~ x1 + x2 + . . .) can be provided as well as a formula including variables and parameters (non-classic) e.g. y ~ x1 * beta1 + x2 * beta2 or y ~ 1/exp(x1 * beta1 + x2 * beta2), where beta are the parameters to be estimated and thexs are column names from the dataset. Non-linear models and the 8-parameter model for EQ-5D data can only be estimated using a non-classic formula. If the provided formula is non-classic, formula_type_classic must be set to FALSE. When estimating an intercept, the formula must explicitly include a parameter named "INTERCEPT"(without a corresponding variable from the dataset) Additionally, it is possible to include a grouping variable for repeated measures by using "| xg" where xg is the column containing the group-memberships. The resulting formula will look like this: y ~ x1 + x2 +. . . | xg. In flexmix, this is called the concomitant variable specification: the model is fit conditional on grouping, so that all observations with the same group are treated as belonging together when computing likelihood contributions. One possible grouping variable

can be an id number to identify answers by the same participants. We highly recommend using a grouping variable, since otherwise the algorithm for k = 2 tends to classify all continuous data into one estimated class and all dichotomous data into the other.

data

a dataframe having the following columns: all independent variables (x) and the dependent variable y used in `formula`, one column for the grouping variable xg if grouping should be used, e.g. id numbers of participants with repeated measurements, one column indicating if the observations belongs to continuous or dichotomous data with the entries `type_cont` and `type_dich` (e.g., for a column called `"type"` with the entries "TTO" for continuous datapoints and "DCE" for dichotomous datapoints, `type_cont` will be "TTO" and `type_dich` will be "DCE"). One row should match one observation (one datapoint).

start values (stv)

if the same start values `stv` are to be used for all latent classes, the given start values must be a `named vector`. Otherwise (if different start values are assumed for each latent class), a `list` of named vectors should be used . In this case, there must be one entry in the list for each latent class. Each start value vector must include start values for sigma and theta. Currently, it is necessary to use the names `"sigma"` and `"theta"` for these values. If users are unsure for which variables start values must be provided (in the linear formula case), this can be checked by calling `colnames(model.matrix(formula,data))`. In this call, the `formula` should not include the grouping variable.

latent, id_col, classes_only

in some situations, it can be useful to identify the latent classes on only one `type` of data while estimating the model parameters on both `types` of data. In such cases, the input variable `latent` can be used to specify on which type of data the classification should be done. If "`cont`" or "`dich`" is used, `formula` must contain a grouping variable and additionally the input parameter `id_col` must be specified and gives the name, i.e. a `character string`, of the grouping variable for classification. Some groups may be removed from the data, since they have only continuous or only dichotomous observations. Then in a first step, a model is estimated only on the continuous/dichotomous data and the achieved classification is stored. In a next step, model parameters are estimated separately for each identified class on both `types` of data using this classification. The output object of `hyreg2` in this case is a `list` of k models. Additionally, at position k+1 of the list, a data frame containing the corresponding classifications from the first step is returned. Each element k in the list contains the estimated parameters for one of the latent classes. When setting the input variable `classes_only` to `TRUE`, the second step is left out and the estimated classes from step one are given as output.

variables_both, variables_cont, variables_dich

It is possible to specify partial coefficients, which are used only on continuous or dichotomous data.

- Example: Suppose different models should be specified for continuous and dichotomous data:
- Model continuous data: `y ~ x1 + x3`
- Model dichotomous data: `y ~ x1 + x2`

- The formula input to hyreg2 must then include all parameters that occur in either model: y ~ x1 + x2 + x3
- The assignment of parameters to data types is then achieved via the input arguments variables_both, variables_cont, and variables_dich:
- variables_both = "x1",
- variables_cont = "x3" and
- variables_dich = "x2".
- Every variable included in the provided formula (except the grouping variable) must appear in exactly one of these vectors. One of the variables_ vectors can also be NULL, if no variables should be used only on this type of the data.

Author(s)

Svenja Elkenkamp, Kim Rand and John Grosser

Examples

```
formula <- y ~  -1 + x1 + x2 + x3 | id

k <- 2
stv <- setNames(c(0.2,0,1,1,1),c(colnames(simulated_data_norm)[3:5],c("sigma","theta")))
control = list(iter.max = 1000, verbose = 4)
rm(counter)
mod <- hyreg2(formula = formula,
                    data =  simulated_data_norm,
                    type =  simulated_data_norm$type, # also "type" would work
                    stv = stv,
                    k = k,
                    type_cont = "TTO",
                    type_dich = "DCE_A",
                    opt_method = "L-BFGS-B",
                    control = control,
                    latent = "cont",
                    id_col = "id"
)
summary_hyreg2(mod)
```

---

hyreg2_het *Estimating hybrid models accounting for heteroscedasticity in continuous data*

Description

Estimation of hybrid model using continuous and dichotomous data e.g. EQ-5D data

Usage

```
hyreg2_het(
  formula,
  formula_sigma = NULL,
  data,
  type,
  type_cont,
  type_dich,
  k = 1,
  control = NULL,
  stv = NULL,
  stv_sigma = NULL,
  offset = NULL,
  opt_method = "BFGS",
  optimizer = "optim",
  lower = -Inf,
  upper = Inf,
  latent = "both",
  id_col = NULL,
  classes_only = FALSE,
  variables_both = NULL,
  variables_dich = NULL,
  variables_cont = NULL,
  ...
)
```

Arguments

formula	linear model formula. Using |xg will include a grouping variable xg. see Details.
formula_sigma	linear formula linear formula for sigma estimation. If formula_sigma is not provided, formula (excluding any grouping variables) is used by default, see Details
data	a data.frame containing the data. see Details.
type	either the name of the column in data containing an indicator of whether an observation is continuous or dichotomous (as character), or a vector containing the indicator. see Details.
type_cont	value of type referring to continuous data. see Details.
type_dich	Value of type referring to dichotomous data. see Details.
k	numeric. Number of latent classes to be estimated via flexmix::flexmix().
control	control list for flexmix::flexmix().
stv	named vector or list of named vectors containing start values for all coefficients formula, including sigma and theta, see Details
stv_sigma	named vector with start values for sigma estimation. Names must correspond to the variables as given in formula_sigma, see Details
offset	offset as in flexmix::flexmix()

opt_method	charachter, optimization method to be used in optimizer, default "BFGS"
optimizer	charachter,optimizer to be used in bbmle::mle2(), default "optim"
lower	numeric, lower bound for censored data, default -INF. If this is used, opt_method must be set to "L-BFGS-B",
upper	numeric, upper bound for censored data, default INF. If this is used, opt_method must be set to "L-BFGS-B",
latent	charachter,data type to use in component identification, must be one of "both", "cont" or "dich", default "both", see Details
id_col	character, name of the grouping variable, only needed if latent != "both", see Details
classes_only	logical, default FALSE, indicates whether the function should perform only classification, rather than both classification and model estimation, only possible for latent != "both", see Datails
variables_both	character vector; variables to be fitted on both continuous and dichotomous data. If not specified, all variables from formula are used. If provided and not all variables from formula are included, variables_cont and variables_dich must be provided as well, while one of them can be NULL, see Details.
variables_dich	character vector; variables to be fitted only on dichotomous data, if provided, variables_both and variables_cont must be provided as well.
variables_cont	character vector; variables to be fitted only on continous data. If provided, variables_both and variables_dich must be provided as well.
...	additional arguments for flexmix::flexmix() or bbmle::mle2()

Details

see details of different inputs listed below

Value

model object of type flemix, coefficients named ..._h are coefficients for heteroscedasticity

formula

a classic R formula of the form y ~ x1 + x2 + . . . should be provided. Additionally, it is possible to include a grouping variable for repeated measures by using "| xg" where xg is the column containing the group-memberships. The resulting formula will look like this: y ~ x1 + x2 +. . . | xg. In flexmix, this is called the concomitant variable specification: the model is fit conditional on grouping, so that all observations with the same group are treated as belonging together when computing likelihood contributions. One possible grouping variable can be an id number to identify answers by the same participants. We highly recommend using a grouping variable, since otherwise the algorithm for k = 2 tends to classify all continuous data into one estimated class and all dichotomous data into the other.

data

a dataframe having the following columns: all independent variables (x) and the dependent variable y used in `formula`, one column for the grouping variable xg if grouping should be used, e.g. id numbers of participants with repeated measurements, one column indicating if the observations belongs to continuous or dichotomous data with the entries `type_cont` and `type_dich` (e.g., for a column called `"type"` with the entries "TTO" for continuous datapoints and "DCE" for dichotomous datapoints, `type_cont` will be "TTO" and `type_dich` will be "DCE"). One row should match one observation (one datapoint).

start values (stv)

if the same start values `stv` are to be used for all latent classes, the given start values must be a `named vector`. Otherwise (if different start values are assumed for each latent class), a `list` of named vectors should be used . In this case, there must be one entry in the list for each latent class. Each start value vector must include start values for sigma and theta. Currently, it is necessary to use the names `"sigma"` and `"theta"` for these values. If users are unsure for which variables start values must be provided, this can be checked by calling `colnames(model.matrix(formula,data))`. In this call, the `formula` should not include the grouping variable.

formula_sigma, stv_sigma

To account for heteroscedasticity in the data, an additional formula `formula_sigma` and an additional vector of starting values for this formula (`stv_sigma`) can be specified. The provided `formula_sigma` must be linear and the vector `stv_sigma` must contain start values for all parameters used in the formula. If neither `formula_sigma` nor `stv_sigma` are provided, the same inputs as for `formula` (without controlling for groups) and `stv` (without sigma) are used. The estimates for `sigma` can be identified in the model output by the ending `"_h"`. It is important to note that, when using `hyreg2_het`, neither `stv` nor `stv_sigma` are allowed to include `sigma`, because `sigma` is estimated with its own formula (in contrast to `hyreg2`, where `sigma` must always be specified in `stv`).

latent, id_col, classes_only

in some situations, it can be useful to identify the latent classes on only one `type` of data while estimating the model parameters on both `types` of data. In such cases, the input variable `latent` can be used to specify on which type of data the classification should be done. If "`cont`" or "`dich`" is used, the input parameter `id_col` must be specified and gives the name, i.e. a `character string`, of the grouping variable for classification. Some groups may be removed from the data, since they have only continuous or only dichotomous observations. Then in a first step, a model is estimated only on the continuous/dichotomous data and the achieved classification is stored. In a next step, model parameters are estimated separately for each identified class on both `types` of data using this classification. The output object of `hyreg2` in this case is a `list` of k models. Additionally, at position k+1 of the list, a data frame containing the corresponding classifications from the first step is returned. Each element k in the list contains the estimated parameters for one of the latent classes. When setting the input variable `classes_only` to `TRUE`, the second step is left out and the estimated classes from step one are given as output.

variables_both, variables_cont, variables_dich

It is possible to specify partial coefficients, which are used only on continuous or dichotomous data.

- Example: Suppose different models should be specified for continuous and dichotomous data:
- Model continuous data: `y ~ x1 + x3`
- Model dichotomous data: `y ~ x1 + x2`
- The `formula` input to `hyreg2` must then include all parameters that occur in either model: `y ~ x1 + x2 + x3`
- The assignment of parameters to data types is then achieved via the input arguments `variables_both`, `variables_cont`, and `variables_dich`:
- `variables_both` = "x1",
- `variables_cont` = "x3" and
- `variables_dich` = "x2".
- Every variable included in the provided `formula` (except the grouping variable) must appear in exactly one of these vectors. One of the `variables_` vectors can also be `NULL`, if no variables should be used only on this type of the data.

Author(s)

Svenja Elkenkamp, Kim Rand and John Grosser

Examples

```
formula <- y ~  -1 + x1 + x2 + x3
formula_sigma <- y ~  x1 + x2 + x3

k <- 1
stv <- setNames(c(0.2,0,1,1),c(colnames(simulated_data_norm)[3:5],c("theta")))
stv_sigma <- setNames(c(0.2,0,1,1),c(colnames(simulated_data_norm)[3:5],c("(Intercept)")))
control = list(iter.max = 1000, verbose = 4)
rm(counter)
mod <- hyreg2_het(formula = formula,
                  formula_sigma = formula_sigma,
                   data =  simulated_data_norm,
                   type =  simulated_data_norm$type, # or "type"
                   stv = stv,
                   stv_sigma = stv_sigma,
                   k = k,
                   type_cont = "TTO",
                   type_dich = "DCE_A",
                   opt_method = "L-BFGS-B",
                   control = control,
                   latent = "both",
                   id_col = "id"
)
summary_hyreg2(mod)
```

plot_hyreg2 *plot function for* hyreg2

Description

Function to visualize model results by hyreg2 or hygre2_het

Usage

```
plot_hyreg2(
  data,
  x,
  y,
  id_col,
  class_df_model,
  type_to_plot = NULL,
  colors = NULL
)
```

Arguments

data	a dataframe, which was used to estimate the model using hyreg2() or hyreg2_het()
x	charachter string, column of data to be plotted on x-axis
y	charachter string, column of data to be plotted on y-axis
id_col	charachter sting, grouping variable, same as was given in model. if model was estimated without grouping, see Details
class_df_model	dataframe of two columns indicating which group belongs to which class, first column named as input id_col, second column named "mod_comp". this input can be generated using the give_class() function, see Details.
type_to_plot	list of two charachter elements. First: columnname of column containing indicator for type of data, Second: value of column type, that should be used for the plot, see details of hyreg2() inputs type and type_cont,type_dich
colors	charachter vector, colors to be used in ggplot, default NULL - than colors are choosen automatically

Details

id_col must be provided anyway, even if the model was estimated without grouping variable. Since there might be no grouping varibale in the data, we recommend to create a new column called "observation" in data using the rownames/observationnumbers as charachter values and use this column as input for id_col in plot_hyreg2, additionally you can use class_df_model = give_class(data,model,"observation"), see example

Value

ggplot object visualizing x against y by classes

Author(s)

Svenja Elkenkamp & John Grosser

Examples

```
# estimate a model using simulated_data_rnorm

formula <- y ~  -1 + x1 + x2 + x3 | id
k <- 2
stv <- setNames(c(0.2,0.2,0.2,1,1),c(colnames(simulated_data_norm)[3:5],c("sigma","theta")))
control <- list(iter.max = 1000, verbose = 4)

hyflex_mod <- hyreg2(formula = formula,
                     data =  simulated_data_norm,
                     type =  simulated_data_norm$type,
                     stv = stv,
                     k = k,
                     type_cont = "TTO",
                     type_dich = "DCE_A",
                     opt_method = "L-BFGS-B",
                     control = control,
                     latent = "cont",
                     id_col = "id"
)
# plotting the variables id against y
plot_hyreg2(data = simulated_data_norm,
          x = "id",
          y = "y",
          id_col = "id",
          class_df_model = give_class(data = simulated_data_norm,
                                model = hyflex_mod,
                                id = "id"))
```

---

simulated_data *simulated_data*

Description

simulated_data

Usage

```
simulated_data
```

Format

`simulated_data`**:**

A simulated data frame with 480 rows and 25 columns, following a combination of normal and binomial distribution

type type of data, "TTO" indicates normal distribution, "DCE_A" indicates binomial distribution

y result of the formula y ~ -1 + mo2 + mo3 + ... + ad4 + ad5

mo2, mo3, mo4, mo5, sc2, sc3, sc4, sc5, ua2, ua3, ua4, ua5, pd2, pd3, pd4, pd5,, ad2, ad3, ad4, ad5, dummy variables for EQ5D data simulation

class original class of the data point

id id number of observations to simulated different persons

y_cens column y censored at 2 (upper boundary) ...

Source

simulated with true parameter values: Class 1: sigma = 0.02, theta = 2 and c(mo2,mo3,mo4,mo5) = c(0.001, 0.05, 0.08, 0.1), c(sc2,sc3,sc4,sc5) = c(0.01, 0.2, 0.36, 0.5), c(ua2,au3,ua4,ua5) = c(0.015, 0.25, 0.5, 0.8), c(pd2,pd3,pd4,pd5) = c(0.1, 0.3, 0.4, 0.6), c(ad2,ad3,ad4,ad5) = c(0.09, 0.19, 0.6, 0.7)

Class 2: sigma = 0.1, theta = 3 and c(mo2,mo3,mo4,mo5) = c(0.2, 0.4, 0.6, 0.8), c(sc2,sc3,sc4,sc5) = c(0.1, 0.3, 0.4, 0.5), c(ua2,au3,ua4,ua5) = c(0.2, 0.25, 0.6, 0.7), c(pd2,pd3,pd4,pd5) = c(0.05, 0.2, 0.27, 0.8), c(ad2,ad3,ad4,ad5) = c(0.15, 0.35, 0.4, 0.65)

`simulated_data_mo` *simulated_data_mo*

Description

simulated_data_mo

Usage

```
simulated_data_mo
```

Format

`simulated_data_mo`**:**

A simulated data frame with 480 rows and 9 columns, following a combination of normal and binomial distribution

type type of data, "TTO" indicates normal distribution, "DCE_A" indicates binomial distribution

y result of the formula y ~ -1 + mo2 + mo3 + mo4 + mo5

mo2, mo3, mo4, mo5 dummy variables

class original class of the data point

id id number of observations to simulated different persons

y_cens column y censored at 0 (lower boundary) ...

Source

simulated with true parameter values: Class 1: sigma = 0.001, theta = 0.2 and c(mo2,mo3,mo4,mo5) = c(0.005, 0.01, 0.08, 0.1) Class 2: sigma = 0.1, theta = 2 and c(mo2,mo3,mo4,mo5) = c(0.2, 0.4, 0.6, 0.8)

`simulated_data_norm` *simulated_data_norm*

Description

simulated_data_norm

Usage

```
simulated_data_norm
```

Format

`simulated_data_norm`**:**

A simulated data frame with 600 rows and 9 columns, following a combination of normal and binomial distribution

type type of data, "TTO" indicates normal distribution, "DCE_A" indicates binomial distribution

y result of the formula y ~ x1 + x2 + x3

x1, x2, x3 random numbers from rnorm

class original class of the data point

id id number of observations to simulated different persons

y_non result of the formula y ~ (x1 * beta1 + x2 * beta3) * (x1 * beta1 + x3 * beta3)

y_cens column y censored at 3 ...

Source

simulated with true parameter values: Class 1: sigma = 1.0, theta = 5 and c(x1,x2,x3) = c(0.5, -0.3, 0.8) Class 2: sigma = 0.5, theta = 2 and c(x1,x2,x3) = c(1.4, 2.3, -0.2)

summary_hyreg2 *model summary for hyreg2 objects*

Description

get model parameters of model generated by hyreg2 or hyreg2_het

Usage

```
summary_hyreg2(object)
```

Arguments

object	modelobject generated with hyreg2() or hyreg2_het()

Value

summary object of bbmle::mle2() model, Please note that the outputs for sigma and theta are on a log-scale and have to be transformed using exp()to get the correct estimated values.

Author(s)

Svenja Elkenkamp

Examples

```
formula <- y ~  -1 + x1 + x2 + x3 | id
k <- 1
stv <- setNames(c(0.2,0,1,1,1),c(colnames(simulated_data_norm)[3:5],c("sigma","theta")))
control = list(iter.max = 1000, verbose = 4)
rm(counter)
mod <- hyreg2(formula = formula,
                    data =  simulated_data_norm,
                    type =  simulated_data_norm$type,
                    stv = stv,
                    k = k,
                    type_cont = "TTO",
                    type_dich = "DCE_A",
                    opt_method = "L-BFGS-B",
                    control = control,
                    latent = "both",
                    id_col = "id"
)
summary_hyreg2(mod)
```

the *creating environment for package internal objects*

Description

creating environment for package internal objects

Usage

```
the
```

Format

An object of class `environment` of length 1.

Index