Package ‘remiod’

May 9, 2026

Type Package

Title Reference-Based Multiple Imputation for Ordinal/Binary Response

Version 1.0.2

Description Reference-based multiple imputation of ordinal and binary responses under Bayesian framework, as described in Wang and Liu (2022) <doi:10.48550/arXiv.2203.02771>. Methods for missing-not-at-random include Jump-to-Reference (J2R), Copy Reference (CR), and Delta Adjustment which can generate tipping point analysis.

License GPL (>= 2)

URL `https://github.com/xsswang/remiod`

RoxygenNote 7.2.2

LazyData true

Imports JointAI, rjags, coda, foreach, data.table, future, doFuture, mathjaxr, survival, ggplot2, ordinal, progressr, Matrix, mcmcse

SystemRequirements JAGS (http://mcmc-jags.sourceforge.net/)

Suggests knitr, rmarkdown, bookdown, R.rsp, ggpubr, testthat (>= 3.0.0), spelling

VignetteBuilder knitr, R.rsp

Encoding UTF-8

Config/testthat/edition 3

Language en-US

Depends R (>= 2.10)

NeedsCompilation no

Author Ying Liu [aut],
Tony Wang [aut, cre] (ORCID: <`https://orcid.org/0000-0002-6760-5456`>)

Maintainer Tony Wang <`xwang@imedacs.com`>

Repository CRAN

Date/Publication 2022-11-18 17:20:02 UTC

Contents

`antidep` *wide format of continuous response of antidepressant data.*

Description

A data set containing the treatment and continuous responses measured at baseline and 4 post-baseline visits

Usage

```
antidep
```

Format

A data frame with 172 rows and 6 variables:

PID Patient ID

tx Treatment, 1 for treated and 0 for placebo

y0 HADM-17 measurement at the baseline

y1, y2, y4, y6 Change score of HADM-17 measurement at the post-baseline week 1, 2, 4, and 6.

Source

https://www.lshtm.ac.uk/research/centres-projects-groups/missing-data#dia-missing-data

---

`clm_MI_CR` *Apply Copy-Reference(CR) Method to Update JAGS MCMC outputs under MAR for Cumulative Logistic Model*

Description

Internal function to obtain Copy-Reference(CR) MCMC from an MAR object.

Usage

```
clm_MI_CR(object, treatment, start = NULL, end = NULL, thin = NULL,
  exclude_chains = NULL, subset = FALSE, ord_cov_dummy = TRUE,
  seed = NULL, mess = FALSE, ...)
```

Arguments

`object`	an object of class remoid
`treatment`	the variable name of treatment. Reference level of treatment should be coded as 0.
`start`	first iteration to be used.
`end`	last iteration to be used.
`thin`	thinning to be applied.
`exclude_chains`	optional vector of numbers, indexing MCMC chains to be excluded from the output.
`subset`	subset of parameters (columns of the mcmc object) to be used.
`ord_cov_dummy`	optional. specify whether ordinal variables should be treated as categorical variables or continuous variables when they are included as covariates in the sequential imputation models. Default is `TRUE`, dummy variables will be created accordingly.
`seed`	optional seed value.
`mess`	logical, should messages be displayed?
`...`	optional arguments pass from main function.

Value

A matrix of MCMC samples with all monitored parameters.A subset of the MCMC sample can be selected using `start`, `end` and `thin`.

`clm_MI_delta` *Apply Delta adjustment to Update JAGS MCMC outputs under MAR for Cumulative Logistic Model*

Description

Internal function to obtain delta-adjusted MCMC from an MAR object.

Usage

```
clm_MI_delta(object, treatment, delta = 0, start = NULL, end = NULL,
  exclude_chains = NULL, thin = NULL, subset = FALSE,
  ord_cov_dummy = TRUE, seed = NULL, mess = FALSE, ...)
```

Arguments

`object`	an object of class remiod
`treatment`	the variable name of treatment. Reference level of treatment should be coded as 0.
`delta`	specific value used for Delta adjustment, applicable only for method="delta".
`start`	first iteration to be used.
`end`	last iteration to be used.
`exclude_chains`	optional vector of numbers, indexing MCMC chains to be excluded from the output.
`thin`	thinning to be applied.
`subset`	subset of parameters (columns of the mcmc object) to be used.
`ord_cov_dummy`	optional. specify whether ordinal variables should be treated as categorical variables or continuous variables when they are included as covariates in the sequential imputation models. Default is `TRUE`, dummy variables will be created accordingly.
`seed`	optional seed value.
`mess`	logical, should messages be displayed?
`...`	optional arguments pass from main function.

Value

A matrix of MCMC samples with all monitored parameters.A subset of the MCMC sample can be selected using `start`, `end` and `thin`.

`clm_MI_J2R` *Apply Jump-to-Reference(J2R) Method to Update JAGS MCMC outputs under MAR for Cumulative Logistic Model*

Description

Internal function to obtain Jump-to-Reference(J2R) MCMC from an MAR object.

Usage

```
clm_MI_J2R(object, treatment, start = NULL, end = NULL, thin = NULL,
  exclude_chains = NULL, subset = FALSE, ord_cov_dummy = TRUE,
  seed = NULL, mess = FALSE, ...)
```

Arguments

`object`	an object of class remiod
`treatment`	the variable name of treatment. Reference level of treatment should be coded as 0.
`start`	first iteration to be used.
`end`	last iteration to be used.
`thin`	thinning to be applied.
`exclude_chains`	optional vector of numbers, indexing MCMC chains to be excluded from the output.
`subset`	subset of parameters (columns of the mcmc object) to be used.
`ord_cov_dummy`	optional. specify whether ordinal variables should be treated as categorical variables or continuous variables when they are included as covariates in the sequential imputation models. Default is `TRUE`, dummy variables will be created accordingly.
`seed`	optional seed value.
`mess`	logical, should messages be displayed?
`...`	optional arguments pass from main function.

Value

A matrix of MCMC samples with all monitored parameters .A subset of the MCMC sample can be selected using `start`, `end` and `thin`.

`extract_MIdata` *Extract a specified number of multiple imputed datasets*

Description

Extract a specified number of multiple imputed datasets

Usage

```
extract_MIdata(object, method = c("MAR", "J2R", "CR", "delta"), delta = 0,
  mi.setting = NULL, M = 10, minspace = 2, mess = FALSE)
```

Arguments

`object`	object inheriting from class 'remoid'
`method`	a method for obtaining multiple-imputed dataset. Options include `MAR`, `J2R`, `CR`, and `delta` adjustment. Default is MAR.
`delta`	specific value used for Delta adjustment, applicable only for method="delta".
`mi.setting`	a list of arguments related to multiple imputation, including trtvar, algorithm, method, include, exclude_chains, thin, start, end, and seed.
`M`	number of imputed datasets
`minspace`	minimum number of iterations between iterations to be chosen as imputed values (to prevent strong correlation between imputed datasets in the case of high autocorrelation of the MCMC chains).
`mess`	logical; should messages be given? Default is TRUE.

Value

A `data.frame` in which the imputed datasets are stacked onto each other. The variable `Imp_` indexes the imputation, while `.rownr` links the rows to the rows of the original data. In cross-sectional datasets the variable `.id` is added as subject identifier.

Examples

```
# data(schizow)

test = remiod(formula = y6 ~ tx + y0 + y1 + y3, data = schizow,
              trtvar = 'tx', algorithm = 'jags', method="MAR",
              ord_cov_dummy = FALSE, n.adapt = 10, n.chains = 1,
              n.iter = 100, thin = 2, warn = FALSE, seed = 1234)

extdt = extract_MIdata(object=test, method="J2R",mi.setting=NULL, M=10, minspace=2)
```

`get_MI_RB` *Create multiple imputed datasets based on assigned imputation method.*

Description

Internal function, creates multiple imputed datasets based on assigned imputation method returns multiple imputed datasets stacked onto each other (i.e., long format; optionally including the original, incomplete data).

Usage

```
get_MI_RB(object, treatment, method = c("MAR", "J2R", "CR", "delta"),
  delta = 0, exclude_chains = NULL, start = NULL, end = NULL,
  seed = NULL, thin = NULL, subset = FALSE, include = TRUE,
  ord_cov_dummy = TRUE, mess = TRUE, ...)
```

Arguments

`object`	an object of class JointAI
`treatment`	the variable name of treatment. Reference level of treatment should be coded as 0.
`method`	a method for obtaining multiple-imputed dataset. Options include MAR, J2R, CR, and Delta adjustment.
`delta`	specific value used for Delta adjustment, applicable only for method="delta".
`exclude_chains`	optional vector of numbers, indexing MCMC chains to be excluded from the output.
`start`	first iteration to be used.
`end`	last iteration to be used.
`seed`	optional seed value.
`thin`	thinning to be applied.
`subset`	subset of parameters (columns of the mcmc object) to be used.
`include`	should the original, incomplete data be included? Default is `TRUE`.
`ord_cov_dummy`	optional. specify whether ordinal variables should be treated as categorical variables or continuous variables when they are included as covariates in the sequential imputation models. Default is `TRUE`, dummy variables will be created accordingly.
`mess`	logical, should messages be displayed?
`...`	optional arguments pass from main function.

Value

A `data.frame` in which the original data (if `include = TRUE`) and the imputed datasets are stacked onto each other.
The variable `Imputation_` indexes the imputation, while `.rownr` links the rows to the rows of the original data. In cross-sectional datasets the variable `.id` is added as subject identifier.

`get_Mlist` *Prepare imputation-model-related information*

Description

Internal function to extract information of imputation models.

Usage

```
get_Mlist(object)
```

Arguments

`object`	object inheriting from class `remoid`.

Value

a list include raw data, imputation models, model types, fixed effects, random effects if any, reference categories corresponding to categorical variables in models, and interaction terms.

`get_subset` *Extract specific parameters from MCMC samples*

Description

Extract specific parameters from MCMC samples

Usage

```
get_subset(object, subset, warn = TRUE, mess = TRUE)
```

Arguments

`object`	an object of class `MCMC`.
`subset`	subset of parameters (columns of the mcmc object) to be used. See https://nerler.github.io/JointAI/articles/ for key-words of subseting parameters. Besides, `selected_parms` and `selected_vars` are new key-words for arbitrarily selecting parameters.
`warn`	logical, should warning messages be displayed?
`mess`	logical, should messages be displayed?

Examples

```
data(schizow)

test = remiod(formula = y6 ~ tx + y0 + y1 + y3, data = schizow,
              trtvar = 'tx', algorithm = 'jags', method="MAR",
              ord_cov_dummy = FALSE, n.adapt = 10, n.chains = 1,
              n.iter = 0, thin = 1, warn = FALSE, seed = 1234)

pms = c("beta[2]","alpha[2]","alpha[6]","alpha[9]")
mcsub = get_subset(object = test$mc.mar, subset=c(selected_parms = list(pms)))
```

`glm_MI_CR` *Apply Copy-Reference(CR) Method to Update JAGS MCMC outputs under MAR for Generalized Linear Model*

Description

Internal function to obtain Copy-Reference(CR) MCMC from an MAR object.

Usage

```
glm_MI_CR(object, treatment, start = NULL, end = NULL, thin = NULL,
  exclude_chains = NULL, subset = FALSE, seed = 5432, mess = FALSE,
  ...)
```

Arguments

`object`	an object of class remiod
`treatment`	the variable name of treatment. Reference level of treatment should be coded as 0.
`start`	first iteration to be used
`end`	last iteration to be used
`thin`	thinning to be applied
`exclude_chains`	optional vector of numbers, indexing MCMC chains to be excluded from the output
`subset`	subset of parameters (columns of the mcmc object) to be used
`seed`	optional seed value.
`mess`	logical, should messages be displayed?
`...`	optional arguments pass from main function.

Value

A matrix of MCMC samples with all monitored parameters.A subset of the MCMC sample can be selected using `start`, `end` and `thin`.

`glm_MI_delta` *Apply Delta adjustment to Update JAGS MCMC outputs under MAR for Generalized Linear Model*

Description

Internal function to obtain delta-adjusted MCMC from an MAR object.

Usage

```
glm_MI_delta(object, treatment, delta = 0, start = NULL, end = NULL,
  thin = NULL, exclude_chains = NULL, subset = FALSE, seed = 5432,
  mess = FALSE, ...)
```

Arguments

`object`	an object of class remiod
`treatment`	the variable name of treatment. Reference level of treatment should be coded as 0.
`delta`	specific value used for Delta adjustment, applicable only for method="delta".
`start`	first iteration to be used
`end`	last iteration to be used
`thin`	thinning to be applied
`exclude_chains`	optional vector of numbers, indexing MCMC chains to be excluded from the output
`subset`	subset of parameters (columns of the mcmc object) to be used
`seed`	optional seed value.
`mess`	logical, should messages be displayed?
`...`	optional arguments pass from main function.

Value

A matrix of MCMC samples with all monitored parameters.A subset of the MCMC sample can be selected using `start`, `end` and `thin`.

`glm_MI_J2R` *Apply Jump-to-Reference(J2R) Method to Update JAGS MCMC outputs under MAR for Generalized Linear Model*

Description

Internal function to obtain Jump-to-Reference(J2R) MCMC from an MAR object.

Usage

```
glm_MI_J2R(object, treatment, start = NULL, end = NULL, thin = NULL,
  exclude_chains = NULL, subset = FALSE, seed = 5432, mess = FALSE,
  ...)
```

Arguments

`object`	an object of class remiod
`treatment`	the variable name of treatment. Reference level of treatment should be coded as 0.
`start`	first iteration to be used
`end`	last iteration to be used
`thin`	thinning to be applied
`exclude_chains`	optional vector of numbers, indexing MCMC chains to be excluded from the output
`subset`	subset of parameters (columns of the mcmc object) to be used
`seed`	optional seed value.
`mess`	logical, should messages be displayed?
`...`	optional arguments pass from main function.

Value

A matrix of MCMC samples with all monitored parameters.A subset of the MCMC sample can be selected using `start`, `end` and `thin`.

`list.models` *Listing the sequence of models used for imputation*

Description

Listing the sequence of models used for imputation

Usage

```
list.models(object, details = FALSE, print = TRUE)
```

Arguments

`object`	an object of class `remiod`
`details`	logical. Default is FALSE, where listing all models in formula format. If TRUE, details of each models will be presented.
`print`	logical. Default is TRUE to print all imputation models or detailed imputation models.

Value

a list of formula of imputation models. If `details=TRUE`, information on the conditional distributions of the covariates in each imputation models. Note: the sequence of conditional models together specifies the joint distribution.

Examples

```
# data(schizow)

test = remiod(formula = y6 ~ tx + y0 + y1 + y3, data = schizow,
              trtvar = 'tx', algorithm = 'jags', method="MAR",
              ord_cov_dummy = FALSE, n.adapt = 10, n.chains = 1,
              n.iter = 10, thin = 2, warn = FALSE, seed = 1234)

list.models(test)
```

`mcmcplot` *Visualizing the posterior sample Creates a set of plots for visually evaluating convergence and mixing of the chains from the MCMC sample of an object of class 'remiod'.*

Description

Visualizing the posterior sample Creates a set of plots for visually evaluating convergence and mixing of the chains from the MCMC sample of an object of class 'remiod'.

Usage

```
mcmcplot(object, what = c("trace", "density"), subset = c(analysis_main =
  TRUE), outcome = NULL, mi.setting = NULL, nrow = NULL, ncol = NULL,
  use_ggplot = TRUE, mess = TRUE, warn = FALSE, ...)
```

Arguments

`object`	an object inheriting from class 'remiod'
`what`	select either trace or density plots from MCMC samples
`subset`	subset of parameters/variables/nodes (columns in the MCMC sample). Follows the same principle as the argument `monitor_params` in `*_imp`.
`outcome`	optional; vector identifying a subset of sub-models included in the output, either by specifying their indices (using the order used in the list of model formulas), or their names (LHS of the respective model formula as character string)
`mi.setting`	a list of arguments for extracting MI data set, which will be used to update the one in `remoid` object. Default is NULL, meaning no update to the mi.setting in `remoid` object.
`nrow`	optional; number of rows in the plot layout; automatically chosen if unspecified
`ncol`	optional; number of columns in the plot layout; automatically chosen if unspecified
`use_ggplot`	logical; Should ggplot be used instead of the base graphics?
`mess`	logical; should messages be given? Default is `TRUE`.
`warn`	logical; should warnings be given? Default is `TRUE`.
`...`	Arguments passed on to `graphics::matplot` `lty,lwd,lend` vector of line types, widths, and end styles. The first element is for the first column, the second element for the second column, etc., even if lines are not plotted for all columns. Line types will be used cyclically until all plots are drawn. `col` vector of colors. Colors are used cyclically. `cex` vector of character expansion sizes, used cyclically. This works as a multiple of `par("cex")`. `NULL` is equivalent to `1.0`.

`bg`	vector of background (fill) colors for the open plot symbols given by `pch = 21:25` as in `points`. The default `NA` corresponds to the one of the underlying function `plot.xy`.
`add`	logical. If `TRUE`, plots are added to current one, using `points` and `lines`.
`verbose`	logical. If `TRUE`, write one line of what is done.

Value

plots of traces or densities of MCMC samples for selected parameters in imputation models.

Examples

```
# data(schizow)

test = remiod(formula = y6 ~ tx + y0 + y1 + y3, data = schizow,
              trtvar = 'tx', algorithm = 'jags', method="MAR",
              ord_cov_dummy = FALSE, n.adapt = 10, n.chains = 1,
              n.iter = 10, thin = 2, warn = FALSE, seed = 1234)

p1 = mcmcplot(object=test, what="trace")
```

`miAnalyze` *Takes multiply imputed datasets (as generated by the `extract_MIdata()` function) and runs an analysis function on each of them.*

Description

Takes multiply imputed datasets (as generated by the `extract_MIdata()` function) and runs an analysis function on each of them.

Usage

```
miAnalyze(formula, family = NULL, data, pool = TRUE)
```

Arguments

`formula`	a two sided model formula (see `formula`).
`family`	only for `glm`: a description of the distribution and link function to be used in the model. This can be a character string naming a family function, a family function or the result of a call to a family function. (For more details see below and `family`.)
`data`	the output object of `extract_MIdata()` function.
`pool`	logical. If TRUE, estimates from each imputed data set will be pooled together according to Rubin's rules. Default is TRUE.

Details

`rubin_rules` applies Rubin's rules (Rubin, 1987) for pooling together the results from a multiple imputation procedure. The pooled point `Estimate` is is the average across the point estimates from the complete-data analyses. The `SE` is the square-root of the sum of two terms representing the within-variance and the between-variance (see Little-Rubin (2002)). The function also returns 95% confidence interval, based on the estimated pooled degrees of freedom that can be used for inference based on the t-distribution.

Value

A list containing

- list of estimated coefficients and standard error from each imputed data.
- pooled estimates based Rubin's rule if `pool = TRUE`.

`opm_MI_CR` *Apply Copy-Reference(CR) Method to Update JAGS MCMC outputs under MAR for probit Model*

Description

Internal function to obtain Copy-Reference(CR) MCMC from an MAR object.

Usage

```
opm_MI_CR(object, treatment, start = NULL, end = NULL, thin = NULL,
  exclude_chains = NULL, subset = FALSE, seed = NULL, mess = FALSE,
  ...)
```

Arguments

`object`	an object of class remoid
`treatment`	the variable name of treatment. Reference level of treatment should be coded as 0.
`start`	first iteration to be used.
`end`	last iteration to be used.
`thin`	thinning to be applied.
`exclude_chains`	optional vector of numbers, indexing MCMC chains to be excluded from the output.
`subset`	subset of parameters (columns of the mcmc object) to be used.
`seed`	optional seed value.
`mess`	logical, should messages be displayed?
`...`	optional arguments pass from main function.

Value

A matrix of MCMC samples with all monitored parameters.A subset of the MCMC sample can be selected using `start`, `end` and `thin`.

`opm_MI_delta` *Apply Delta adjustment to Update JAGS MCMC outputs under MAR for Cumulative Logistic Model*

Description

Internal function to obtain delta-adjusted MCMC from an MAR object.

Usage

```
opm_MI_delta(object, treatment, delta = 0, start = NULL, end = NULL,
  thin = NULL, exclude_chains = NULL, subset = FALSE, seed = NULL,
  mess = FALSE, ...)
```

Arguments

`object`	an object of class remiod
`treatment`	the variable name of treatment. Reference level of treatment should be coded as 0.
`delta`	specific value used for Delta adjustment, applicable only for method="delta".
`start`	first iteration to be used.
`end`	last iteration to be used.
`thin`	thinning to be applied.
`exclude_chains`	optional vector of numbers, indexing MCMC chains to be excluded from the output.
`subset`	subset of parameters (columns of the mcmc object) to be used.
`seed`	optional seed value.
`mess`	logical, should messages be displayed?
`...`	optional arguments pass from main function.

Value

A matrix of MCMC samples with all monitored parameters.A subset of the MCMC sample can be selected using `start`, `end` and `thin`.

`opm_MI_J2R` *Apply Jump-to-Reference(J2R) Method to Update JAGS MCMC outputs under MAR for Cumulative Logistic Model*

Description

Internal function to obtain Jump-to-Reference(J2R) MCMC from an MAR object.

Usage

```
opm_MI_J2R(object, treatment, start = NULL, end = NULL, thin = NULL,
  exclude_chains = NULL, subset = FALSE, seed = NULL, mess = FALSE,
  ...)
```

Arguments

`object`	an object of class remiod
`treatment`	the variable name of treatment. Reference level of treatment should be coded as 0.
`start`	first iteration to be used.
`end`	last iteration to be used.
`thin`	thinning to be applied.
`exclude_chains`	optional vector of numbers, indexing MCMC chains to be excluded from the output.
`subset`	subset of parameters (columns of the mcmc object) to be used.
`seed`	optional seed value.
`mess`	logical, should messages be displayed?
`...`	optional arguments pass from main function.

Value

A matrix of MCMC samples with all monitored parameters .A subset of the MCMC sample can be selected using `start`, `end` and `thin`.

`remiod` *Reference-Based Multiple Imputation for Ordinal/Binary Response*

Description

Reference-Based Multiple Imputation for Ordinal/Binary Response

Usage

```
remiod(formula, data, trtvar, refcats = NULL, family = NULL,
  method = "MAR", delta = 0, algorithm = c("tang_seq", "jags"),
  rinv = 1e-04, scheme = 2, model_order = NULL, models = NULL,
  ord_cov_dummy = TRUE, n.chains = 2, n.adapt = 100, n.iter = 1000,
  thin = 2, start = NULL, end = NULL, seed = 1234,
  exclude_chains = NULL, subset = NULL, include = FALSE, mess = TRUE,
  warn = FALSE, progress.bar = TRUE, ...)
```

Arguments

`formula`	a two sided model formula (see `formula`) or a list of such formulas; (more details below).
`data`	a `data.frame` containing the original data (more details below)
`trtvar`	the name of treatment variable. When necessary, its reference category, i.e. control arm, can be set in `refcats` argument.
`refcats`	optional; either one of `"first"`, `"last"`, `"largest"` (which sets the category for all categorical variables) or a named list specifying which category should be used as reference category per categorical variable. Options are the category label, the category number, or one of "first" (the first category), "last" (the last category) or "largest" (chooses the category with the most observations). Default is "first". If reference categories are specified for a subset of the categorical variables the default will be used for the remaining variables. (See also `set_refcat`)
`family`	only for `glm_imp` and `glmm_imp`/`glmer_imp`: a description of the distribution and link function to be used in the model. This can be a character string naming a family function, a family function or the result of a call to a family function. (For more details see below and `family`.)
`method`	a method for obtaining multiple-imputed dataset. Options include `MAR`, `J2R`, `CR`, and `delta` adjustment. Default is MAR.
`delta`	specific value used for Delta adjustment, applicable only for method="delta".
`algorithm`	either algorithm `tang_seq` proposed by Tang (2018) or `jags` the original method inherited in JAGS (Plummer 2003).
`rinv`	a small number used to adjusting Fish information matrix
`scheme`	scheme of distribution used for proposing coefficients of imputation models. scheme=1: beta ~ N(mean + inv(I)*score, inv(I)); scheme=2: beta ~ N(mean , inv(I)).

`model_order`	optional. manually specify an order for imputation models.
`models`	optional; named vector specifying the types of models for (incomplete) covariates. This arguments replaces the argument `meth` used in earlier versions. If `NULL` (default) models will be determined automatically based on the class of the respective columns of `data`.
`ord_cov_dummy`	optional. specify whether ordinal variables should be treated as categorical variables or continuous variables when they are included as covariates in the sequential imputation models. Default is `TRUE`, dummy variables will be created accordingly.
`n.chains`	number of MCMC chains
`n.adapt`	number of iterations for adaptation of the MCMC samplers (see `adapt`)
`n.iter`	number of iterations of the MCMC chain (after adaptation; see `coda.samples`)
`thin`	thinning interval (integer; see `window.mcmc`). For example, `thin = 1` (default) will keep the MCMC samples from all iterations; `thin = 5` would only keep every 5th iteration.
`start`	the first iteration of interest (see `window.mcmc`)
`end`	the last iteration of interest (see `window.mcmc`)
`seed`	optional; seed value (for reproducibility)
`exclude_chains`	optional vector of the index numbers of chains that should be excluded
`subset`	subset of parameters/variables/nodes (columns in the MCMC sample). Follows the same principle as the argument `monitor_params` and `selected_parms`.
`include`	logical, if TRUE, raw data will be included in imputed data sets with imputation ID = 0.
`mess`	logical; should messages be given? Default is `TRUE`.
`warn`	logical; should warnings be given? Default is `TRUE`.
`progress.bar`	character string specifying the type of progress bar. Possible values are "text" (default), "gui", and "none" (see `update`). Note: when sampling is performed in parallel it is not possible to display a progress bar.
`...`	additional, optional arguments `trunc` named list specifying limits of truncation for the distribution of the named incomplete variables (see the vignette ModelSpecification) `hyperpars` list of hyper-parameters, as obtained by `default_hyperpars()` `scale_vars` named vector of (continuous) variables that will be centred and scaled (such that mean = 0 and sd = 1) when they enter a linear predictor to improve convergence of the MCMC sampling. Default is that all numeric variables and integer variables with >20 different values will be scaled. If set to `FALSE` no scaling will be done. `custom` named list of JAGS model chunks (character strings) that replace the model for the given variable. `append_data_list` list that will be appended to the list containing the data that is passed to **rjags** (`data_list`). This may be necessary if additional data / variables are needed for custom (covariate) models.

`progress.bar` character string specifying the type of progress bar. Possible values are "text" (default), "gui", and "none" (see `update`). Note: when sampling is performed in parallel it is not possible to display a progress bar.

`quiet` logical; if `TRUE` then messages generated by **rjags** during compilation as well as the progress bar for the adaptive phase will be suppressed, (see `jags.model`)

`keep_scaled_mcmc` should the "original" MCMC sample (i.e., the scaled version returned by `coda.samples()`) be kept? (The MCMC sample that is re-scaled to the scale of the data is always kept.)

`modelname` character string specifying the name of the model file (including the ending, either .R or .txt). If unspecified a random name will be generated.

`modeldir` directory containing the model file or directory in which the model file should be written. If unspecified a temporary directory will be created.

`overwrite` logical; whether an existing model file with the specified `<modeldir>/<modelname>` should be overwritten. If set to `FALSE` and a model already exists, that model will be used. If unspecified (`NULL`) and a file exists, the user is asked for input on how to proceed.

`keep_model` logical; whether the created JAGS model file should be saved or removed from (`FALSE`; default) when the sampling has finished.

Value

A list includes (1) Information from JAGS modeling and MCMC samples and (2) A `data.frame` in which the original data (if `include = TRUE`) and the imputed datasets are stacked onto each other. The variable `Imputation_` indexes the imputation, while `.rownr` links the rows to the rows of the original data. In cross-sectional datasets the variable `.id` is added as subject identifier.

Examples

```
data(schizow)

test = remiod(formula = y6 ~ tx + y0 + y1 + y3, data = schizow,
              trtvar = 'tx', algorithm = 'jags', method="MAR",
              ord_cov_dummy = FALSE, n.adapt = 10, n.chains = 1,
              n.iter = 10, thin = 2, warn = FALSE, seed = 1234)
```

`schizo` *National Institute of Mental Health shizophrenia study*

Description

Schizophrenia data from a randomized controlled trial with patients assigned to either drug or placebo group. "Severity of Illness" was measured, at weeks 0, 1, . . . , 6, on a four category ordered scale. Most of the observations where made on weeks 0, 1, 3, and 6.

Usage

`data(schizo)`

Format

A data frame with 1603 observations on 437 subjects. Five vectors contain information on

`id` patient ID.

`imps79` original response measurements on a numerical scale.

`imps79b` binary response based on the cut-off value of 3.5 to the measurements on a numerical scale: 0 = normal to mildly ill and 1 = moderately to extremely ill

`imps79o` ordinal response on a 4 category scale, "normal or borderline mentally ill" < "mildly or moderately ill", "markedly ill", "severely or among the most extremely ill".

`tx` treatment indicator: 1 for drug, 0 for placebo.

`week` week.

Details

The documentation file was copied from the **mixcat** package and slightly modified.

Source

`https://hedeker.people.uic.edu/ml.html`

References

Hedeker, D. and R. Gibbons (2006). *Longitudinal Data Analysis*. New Jersey, USA: John Wiley & Sons.

`schizob` *wide format of binary response of Schizophrenia data.*

Description

A dataset containing the treatment and binary responses measured at baseline and 3 post-baseline visits

Usage

`schizob`

Format

A data frame with 437 rows and 5 variables:

tx treatment, 1 for treated and 0 for placebo

y0 binary response at the baseline

y1, y3, y6 binary response at the post-baseline week 1, 3, and 6.

Source

long-to-wise tranformation of schizo data, i.e. schizob = data.table::dcast(schizo, id + tx ~ week, value.var = "imps79b")

`schizow` *wide format of ordinal response of Schizophrenia data.*

Description

A dataset containing the treatment and ordinal responses measured at baseline and 3 post-baseline visits

Usage

```
schizow
```

Format

A data frame with 437 rows and 5 variables:

tx treatment, 1 for treated and 0 for placebo

y0 ordinal response at the baseline

y1, y3, y6 ordinal response at the post-baseline week 1, 3, and 6.

Source

long-to-wise tranformation of schizo data, i.e. schizow = data.table::dcast(schizo, id + tx ~ week, value.var = "imps79o")

`summary` *Summarize the results from an object of class remiod*

Description

Obtain and print the `summary`, (fixed effects) coefficients (`coef`) and credible interval (`confint`).

Usage

```
summary(object, ...)

## S3 method for class 'remiod'
summary(object, start = NULL, end = NULL, thin = NULL,
  quantiles = c(0.025, 0.975), outcome = NULL, exclude_chains = NULL,
  warn = TRUE, mess = TRUE, ...)

## S3 method for class 'summary.remiod'
print(x, digits = 3, ...)

## S3 method for class 'summary.remiod'
coef(object, start = NULL, end = NULL,
  thin = NULL, subset = NULL, exclude_chains = NULL, warn = TRUE,
  mess = TRUE, ...)
```

Arguments

`object`	object inheriting from class 'remiod'
`...`	additional, optional arguments `trunc` named list specifying limits of truncation for the distribution of the named incomplete variables (see the vignette ModelSpecification) `hyperpars` list of hyper-parameters, as obtained by `default_hyperpars()` `scale_vars` named vector of (continuous) variables that will be centred and scaled (such that mean = 0 and sd = 1) when they enter a linear predictor to improve convergence of the MCMC sampling. Default is that all numeric variables and integer variables with >20 different values will be scaled. If set to `FALSE` no scaling will be done. `custom` named list of JAGS model chunks (character strings) that replace the model for the given variable. `append_data_list` list that will be appended to the list containing the data that is passed to **rjags** (`data_list`). This may be necessary if additional data / variables are needed for custom (covariate) models. `progress.bar` character string specifying the type of progress bar. Possible values are "text" (default), "gui", and "none" (see `update`). Note: when sampling is performed in parallel it is not possible to display a progress bar. `quiet` logical; if `TRUE` then messages generated by **rjags** during compilation as well as the progress bar for the adaptive phase will be suppressed, (see `jags.model`) `keep_scaled_mcmc` should the "original" MCMC sample (i.e., the scaled version returned by `coda.samples()`) be kept? (The MCMC sample that is re-scaled to the scale of the data is always kept.) `modelname` character string specifying the name of the model file (including the ending, either .R or .txt). If unspecified a random name will be generated. `modeldir` directory containing the model file or directory in which the model file should be written. If unspecified a temporary directory will be created.

	`overwrite` logical; whether an existing model file with the specified `<modeldir>/<modelname>` should be overwritten. If set to `FALSE` and a model already exists, that model will be used. If unspecified (`NULL`) and a file exists, the user is asked for input on how to proceed.
	`keep_model` logical; whether the created JAGS model file should be saved or removed from (`FALSE`; default) when the sampling has finished.
`start`	the first iteration of interest (see `window.mcmc`)
`end`	the last iteration of interest (see `window.mcmc`)
`thin`	thinning interval (integer; see `window.mcmc`). For example, `thin = 1` (default) will keep the MCMC samples from all iterations; `thin = 5` would only keep every 5th iteration.
`quantiles`	posterior quantiles
`outcome`	specify outcome variable to select imputation model(s) to summarize. Default generates summaries for all models.
`exclude_chains`	optional vector of the index numbers of chains that should be excluded
`warn`	logical; should warnings be given? Default is `TRUE`.
`mess`	logical; should messages be given? Default is `TRUE`.
`x`	an object of class `summary.remiod`
`digits`	the minimum number of significant digits to be printed in values.
`subset`	subset of parameters/variables/nodes (columns in the MCMC sample). Follows the same principle as the argument `monitor_params` and `selected_parms`.

Value

summary information, including parameter posterior mean, posterior SD, quantiles, tail probability `tail-prob`, Gelman-Rubin criterion `GR-crit`, the ratio of the Monte Carlo error and posterior standard deviation) for specified parameters `MCE/SD`.

Examples

```
# data(schizow)

test = remiod(formula = y6 ~ tx + y0 + y1 + y3, data = schizow,
              trtvar = 'tx', algorithm = 'jags', method="MAR",
              ord_cov_dummy = FALSE, n.adapt = 50, n.chains = 1,
              n.iter = 50, thin = 2, warn = FALSE, seed = 1234)

summary(object = test, outcome = c("y6","y3"))
```

`tang_MI_RB` *Implement controlled multiple imputation algorithms proposed by Tang*

Description

Internal function, creates multiple imputed datasets based on assigned imputation method with the algorithm of Tang's sequential modeling.

Usage

```
tang_MI_RB(object, dtimp, treatment, method = "MAR", delta = 0,
  ord_cov_dummy = FALSE, exclude_chains = NULL, include = FALSE,
  thin = 1)
```

Arguments

`object`	object inheriting from class 'remoid'
`dtimp`	imputed complete data sets from `remiod` function.
`treatment`	name of the treatment variable.
`method`	a method for obtaining multiple-imputed dataset. Options include `MAR`, `J2R`, `CR`, and `delta` adjustment. Default is MAR.
`delta`	specific value used for Delta adjustment, applicable only for method="delta".
`ord_cov_dummy`	optional. specify whether ordinal variables should be treated as categorical variables or continuous variables when they are included as covariates in the sequential imputation models. Default is `TRUE`, dummy variables will be created accordingly.
`exclude_chains`	optional vector of the index numbers of chains that should be excluded
`include`	logical, if TRUE, raw data will be included in imputed data sets with imputation ID = 0.
`thin`	thinning to be applied.

Value

multiple imputed datasets stacked onto each other (i.e., long format; optionally including the original incomplete data).
The variable `Imputation_` indexes the imputation, while `.rownr` links the rows to the rows of the original data. In cross-sectional datasets the variable `.id` is added as subject identifier.

Index