Package ‘RobinCar’

May 7, 2026

Type Package

Title Robust Inference for Covariate Adjustment in Randomized Clinical Trials

Version 1.2.0

Description
Performs robust estimation and inference when using covariate adjustment and/or covariate-adaptive randomization in randomized clinical trials.
Ting Ye, Jun Shao, Yanyao Yi, Qinyuan Zhao (2023) <doi:10.1080/01621459.2022.2049278>.
Ting Ye, Marlena Bannick, Yanyao Yi, Jun Shao (2023) <doi:10.1080/24754269.2023.2205802>.
Ting Ye, Jun Shao, Yanyao Yi (2023) <doi:10.1093/biomet/asad045>.
Marlena Bannick, Jun Shao, Jingyi Liu, Yu Du, Yanyao Yi, Ting Ye (2024) <doi:10.1093/biomet/asaf029>.
Xiaoyu Qiu, Yuhan Qian, Jaehwan Yi, Jinqiu Wang, Yu Du, Yanyao Yi, Ting Ye (2025) <doi:10.48550/arXiv.2408.12541>.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

Imports dplyr, magrittr, tidyr, emulator, numDeriv, tidyverse, stats, rlang, survival, fastDummies, data.table, broom, SuperLearner, AIPW, MASS, Rdpack (>= 0.7)

Depends R (>= 2.10)

Suggests knitr, rmarkdown, ranger, forcats, matrixcalc, testthat (>= 3.0.0)

RdMacros Rdpack

Config/testthat/edition 3

NeedsCompilation no

Author Marlena Bannick [cre, aut] (ORCID:
<`https://orcid.org/0000-0001-6279-5978`>),
Yuhan Qian [aut],
Ting Ye [aut],
Yanyao Yi [aut],
Faith Bian [aut]

Maintainer Marlena Bannick <`marlena.bannick@outlook.com`>

Repository CRAN

Date/Publication 2026-04-24 20:20:16 UTC

Contents

`car_pb` *Generate permuted block treatment assignments*

Description

Generate permuted block treatment assignments

Usage

```
car_pb(z, trt_label, trt_alc, blocksize = 4L)
```

Arguments

`z`	The car_strata design matrix, as a data frame with factor variables
`trt_label`	Treatment label
`trt_alc`	Treatment allocation vector
`blocksize`	Permuted block blocksize

Value

A vector of treatment assignments with labels from the ‘trt_label‘ argument, based on stratified permuted block randomization.

Examples

```
# Create car_strata variables
library(fastDummies)
library(dplyr)

x <- runif(100)
z <- cut(x, breaks=c(0, 0.25, 0.5, 0.75, 1.0))
z <- dummy_cols(z) %>%
     mutate(across(where(is.numeric), as.factor))

car_pb(z[, 2:5], c(0, 1, 2), trt_alc=c(1/4, 1/2, 1/4), blocksize=4L)
```

`car_ps` *Generate Pocock-Simon minimization treatment assignments*

Description

Generate Pocock-Simon minimization treatment assignments

Usage

```
car_ps(z, treat, ratio, imb_measure, p_bc = 0.8)
```

Arguments

`z`	The car_strata design matrix
`treat`	A vector of length k (the number of treatment arms), which labels the treatment arms being compared.
`ratio`	A vector of length k (the number of treatment arms), which indicates the allocation ratio, e.g., c(1,1,1) for equal allocation with three treatment arms.
`imb_measure`	What measure of imbalance should be minimzed during randomization – either "Range" or "SD"
`p_bc`	The biased probability, i.e., the probability of assigning each patient to the arm that minimizes the imbalance. Default is 0.8

Value

res treatment assignment vector

A vector of treatment assignments with labels from the ‘treat‘ argument, based on Pocock-Simon’s minimization.

Author(s)

Ting Ye Yanyao Yi

Examples

```
# Create car_strata variables
library(fastDummies)
library(dplyr)

x <- runif(100)
z <- cut(x, breaks=c(0, 0.25, 0.5, 0.75, 1.0))
z <- dummy_cols(z)
A <- car_ps(
  z=z[, 2:5],
  treat=c(0, 1, 2),
  ratio=c(1, 1, 1),
  imb_measure="Range"
)
```

`car_sr` *Generate simple randomization treatment assignments*

Description

Generate simple randomization treatment assignments

Usage

```
car_sr(n, p_trt)
```

Arguments

`n`	Number of observations
`p_trt`	Proportion allotted to treatment

Value

A vector of treatment assignments as 0’s and 1’s based on simple randomization.

Examples

```
car_sr(10, p_trt=0.4)
```

`data_gen` *Data generation function from JRSS-B paper*

Description

Data generation function from JRSS-B paper

Usage

```
data_gen(
  n,
  theta,
  randomization,
  p_trt,
  case = c("case1", "case2", "case3", "case4", "case5")
)
```

Arguments

`n`	total number of subjects to be generated
`theta`	true treatment effect
`randomization`	randomization method in c("SR","CABC","permuted_block","minimization","urn")
`p_trt`	proportion of treatment arm
`case`	simulation case in the paper

Value

A data frame with the following columns:

`t`	event time
`delta`	event indicator
`I1`	assignment to treatment group 1
`I0`	assignment to treatment group 0
`model_z1`, `model_z2`	covariates
`car_strata1`, ...	strata variables

`data_gen2` *Data generation function from covariate adjusted log-rank paper*

Description

Data generation function from covariate adjusted log-rank paper

Usage

```
data_gen2(
  n,
  theta,
  randomization,
  p_trt,
  case = c("case1", "case2", "case3", "case4"),
  blocksize = 4
)
```

Arguments

`n`	total number of subjects to be generated
`theta`	true treatment effect
`randomization`	randomization method in c("SR","CABC","permuted_block","minimization","urn")
`p_trt`	proportion of treatment arm
`case`	simulation case in the paper
`blocksize`	block size for permuted block design

Value

A data frame with the following columns:

`t`	event time
`delta`	event indicator
`I1`	assignment to treatment group 1
`I0`	assignment to treatment group 0
`model_w3`	covariates
`car_strata1, ...`	strata variables

`print.CalibrationResult` *Print calibration result*

Description

Print calibration result

Usage

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

Arguments

`x`	A GLMModel result. If you'd like to calibrate a linear adjustment, use ‘robincar_glm‘ instead of ‘robincar_linear‘.
`...`	Additional arguments

Value

Prints the treatment mean estimates (and variances) based on a calibration on top of a GLM working model, along with the settings used. See RobinCar::robincar_calibrate().

`print.ContrastResult` *Print contrast result*

Description

Print contrast result

Usage

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

Arguments

`x`	A ContrastResult object
`...`	Additional arguments

Value

Prints estimates (and variances) of treatment contrasts based on a linear or GLM working model, along with the settings used. See RobinCar::robincar_contrast()

`print.GLMModelResult` *Print glm model result*

Description

Print glm model result

Usage

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

Arguments

`x`	A GLMModelResult object
`...`	Additional arguments

Value

Prints the treatment mean estimates (and variances) based on a GLM working model, along with the settings used. See RobinCar::robincar_glm().

`print.LinModelResult` *Print linear model result*

Description

Print linear model result

Usage

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

Arguments

`x`	A LinModelResult object
`...`	Additional arguments

Value

Prints the treatment mean estimates (and variances) based on a linear working model, along with the settings used. See RobinCar::robincar_linear().

`print.MHResult` *Print MH result*

Description

Print MH result

Usage

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

Arguments

`x`	A ContrastResult object
`...`	Additional arguments

Value

Prints estimates (and variances) of treatment contrasts based on MH risk difference or ATE

`print.TTEResult` *Print TTE result*

Description

Print TTE result

Usage

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

Arguments

`x`	A TTEResult object
`...`	Additional arguments

Value

Prints results of time-to-event covariate adjusted analyses including covariate-adjusted (stratified) logrank, robust Cox score, and covariate-adjusted hazard ratio. Prints summary statistics about number of observations and events, possibly by strata, and the test statistics and/or estimates, and p-values. See RobinCar::robincar_tte() and RobinCar::robincar_covhr().

`robincar_calibrate` *Perform linear or joint calibration*

Description

Uses linear or joint calibration to "calibrate" the estimates from a linear or GLM-type adjustment. Linear calibration fits a linear model with treatment (and treatment-by-covariate interactions) and with the predicted $\hat{\boldsymbol{\mu}}(X_i) = (\hat{\mu}_1(X_i), \ldots, \hat{\mu}_K(X_i))$ as constructed covariates where K is the number of treatment groups; joint calibration also includes Z_i the strata variables as covariates.

Usage

```
robincar_calibrate(result, joint = FALSE, add_x = NULL)
```

Arguments

`result`	A GLMModelResult
`joint`	If true, then performs joint calibration with the $\hat{\boldsymbol{\mu}}(X_i)$ and strata Z_i to achieve universality and efficiency gain rather than just linear calibration that uses $\hat{\boldsymbol{\mu}}(X_i)$.
`add_x`	Additional x to use in the calibration. Must have been in the original dataset that robincar_glm was called on.

Value

A result object that has the same structure as RobinCar::robincar_glm(), with the argument ‘result‘ included as "original" in the list.

`robincar_contrast` *Estimate a treatment contrast*

Description

Estimate a treatment contrast using the result of RobinCar::robincar_linear(), RobinCar::robincar_glm(), or RobinCar::robincar_SL() using the delta method.

Usage

```
robincar_contrast(result, contrast_h, contrast_dh = NULL)
```

Arguments

`result`	A LinModelResult or GLMModelResult
`contrast_h`	An optional function to specify a desired contrast
`contrast_dh`	An optional jacobian function for the contrast

Value

A contrast object which has the following attributes:

`result`	A dplyr::tibble() with the label of the treatment contrast (e.g., 1 vs. 0), the estimate of the treatment contrast, estimated SE, and p-value based on a z-test with estimate and SE.
`varcov`	The variance-covariance matrix for the treatment contrast estimates.
`settings`	List of model settings used for the contrast.

---

robincar_covhr *Covariate-adjusted estimators for time to event data*

Description

Estimate a covariate-adjusted hazard ratio (‘adj_method="CL"‘), or a covariate-adjusted stratified hazard ratio (‘adj_method="CSL"‘).

Usage

```
robincar_covhr(
  df,
  treat_col,
  response_col,
  event_col,
  car_strata_cols = NULL,
  covariate_cols = NULL,
  p_trt = 0.5,
  ref_arm = NULL,
  car_scheme = "simple",
  adj_method = "CL",
  interval = c(-10, 10)
)
```

Arguments

`df`	A data.frame with the required columns
`treat_col`	Name of column in df with treatment variable
`response_col`	Name of the column in df with response variable
`event_col`	Name of column in df with event indicator (0/FALSE=no event, 1/TRUE=event)
`car_strata_cols`	Names of columns in df with car_strata variables
`covariate_cols`	Names of columns in df with covariate variables
`p_trt`	Treatment allocation ratio for the reference arm.

`ref_arm`	Reference arm of the treatment group, defaults to NULL, which results in using the first element of ‘unique(data[, treat_col])‘.
`car_scheme`	Name of the type of covariate-adaptive randomization scheme. One of: "simple", "pocock-simon", "biased-coin", "permuted-block".
`adj_method`	Adjustment method (one of "CL", "CSL")
`interval`	Interval for uniroot function

Value

An object with attribute named "result", which lists:

`theta_L`	estimate of the hazard ratio
`se_theta_L`	SE estimate of the hazard ratio
`theta_CL`	estimate of the covariate-adjusted hazard ratio
`se_theta_CL`	SE estimate of the covariate-adjusted hazard ratio

Other attributes are the settings used, data attributes, and the original data frame supplied by the user.

`robincar_coxscore` *Robust cox score adjustment*

Description

Robust cox score adjustment

Usage

```
robincar_coxscore(...)
```

Arguments

`...`	Arguments to robincar_tte, other than ‘adj_method‘

Value

A result object with the following attributes:

`result`	A list: "statistic" is the robust Cox score test statistic which can be used to obtain p-values; "U" and "se" are the numerator and denominator of the test statistic, respectively.
`settings`	The covariate adjustment settings used.
`original_df`	The dataset supplied by the user.

`robincar_glm` *Covariate adjustment using generalized linear working model*

Description

Estimate treatment-group-specific response means and (optionally) treatment group contrasts using a generalized linear working model.

Usage

```
robincar_glm(
  df,
  treat_col,
  response_col,
  formula = NULL,
  car_strata_cols = NULL,
  car_scheme = "simple",
  g_family = stats::gaussian,
  g_accuracy = 7,
  contrast_h = NULL,
  contrast_dh = NULL,
  variance_type = 1
)
```

Arguments

`df`	A data.frame with the required columns
`treat_col`	Name of column in df with treatment variable
`response_col`	Name of the column in df with response variable
`formula`	The formula to use for adjustment specified using as.formula("..."). This overrides car_strata_cols and covariate_cols.
`car_strata_cols`	Names of columns in df with car_strata variables
`car_scheme`	Name of the type of covariate-adaptive randomization scheme. One of: "simple", "pocock-simon", "biased-coin", "permuted-block".
`g_family`	Family that would be supplied to glm(...), e.g., binomial. If no link specified, will use default link, like behavior in glm. If you wish to use a negative binomial working model with an unknown dispersion parameter, then use ‘g_family="nb"‘.
`g_accuracy`	Level of accuracy to check prediction un-biasedness.
`contrast_h`	An optional function to specify a desired contrast
`contrast_dh`	An optional jacobian function for the contrast (otherwise use numerical derivative)
`variance_type`	The type of variance estimator to use, type 1, 2, or 3. All three are asymptotically equivalent. See details for more.

Details

The output is the AIPW estimator given by (for each treatment group a):

$$\frac{1}{n}\sum_{i=1}^{n}\hat{\mu}_a(X_i) + \frac{1}{n_a}\sum_{i:A_i=a}\{Y_i - \hat{\mu}(X_i)\}$$

where Y_i is the outcome, A_i is the treatment assignment, X_i are the covariates, $n_a = \sum_{i=1}^{n} A_i = a$, and $\hat{\mu}_a$ is the estimated conditional mean function based on the GLM working model. This working model has treatment a-specific coefficients if ‘adj_method‘ is "heterogeneous". Otherwise, they are shared across the treatment arms. Alternatively, if ‘formula‘ is used, the working model can be specified according to the user.

Importantly, the estimated variance accounts for misspecification of the working model, and for covariate-adaptive randomization. The variance estimator is given by

$$\hat{V} = \hat{V}_{\mathrm{SR}} - \hat{V}_{\Omega}$$

where $\hat{V}_{\mathrm{SR}}$ is the contribution to the variance under simple randomization, and $\hat{V}_{\Omega}$ is a term that only appears when a covariate-adaptive randomization scheme is used. The $\hat{V}_{\Omega}$ is the second line of $\hat{V}$ in (Bannick et al. 2025).

There are three different estimators available for $\hat{V}_{\mathrm{SR}}$, which the user can choose with the argument `variance_type`. We describe these here.

The three variance types are given as follows:

- Type 1 (default):

$$\mathrm{diag}\left[\hat{\pi}_a^{-1}(\mathrm{Var}_a(Y_i) - 2\hat{Q}_{a,a} + \hat{\Sigma}_{a,a}), a = 1,\ldots,K\right] + \hat{Q} + \hat{Q}^T - \hat{\Sigma}$$

- Type 2:

$$\mathrm{diag}\left[\hat{\pi}_a^{-1}(\mathrm{Var}_a(Y_i - \hat{\mu}_a(X_i)) - 2\hat{Q}_{a,a} + \hat{\Sigma}_{a,a}), a = 1,\ldots,K\right] + \hat{Q} + \hat{Q}^T - \hat{\Sigma}$$

- Type 3:

$$\mathrm{diag}\left[\hat{\pi}_a^{-1}E_a([Y_i - \hat{\mu}_a(X_i)]^2), a = 1,\ldots,K\right] + \hat{A}$$

where $\hat{\pi}_a$ is the treatment proportion for group a, $\hat{Q}_{a,b} = \mathrm{Cov}_a(Y_i, \hat{\mu}_b(X_i))$, $\hat{\Sigma}_{a,b} = \mathrm{Cov}(\hat{\mu}_a(X_i), \hat{\mu}_b(X_i))$, and the matrix $\hat{A}$ has diagonal entries for (a,a) given by

$$2\mathrm{Cov}_a(Y_i - \hat{\mu}_a(X_i), \hat{\mu}_a(X_i)) + \mathrm{Var}_a(\hat{\mu}_a(X_i))$$

and off-diagonal entries for (a,b) given by

$$\mathrm{Cov}_a(Y_i, \hat{\mu}_b(X_i)) + \mathrm{Cov}_b(Y_i, \hat{\mu}_a(X_i)) - (1/2)\left[\mathrm{Cov}_a(\hat{\mu}_a(X_i), \hat{\mu}_b(X_i)) + \mathrm{Cov}_b(\hat{\mu}_a(X_i), \hat{\mu}_b(X_i))\right].$$

We use E_a, Var_a, and Cov_a to refer to the empirical expectation, variance, and covariance among observations in group a only, and Cov is the covariance within the entire sample.

Please see the Supplemental Material Sect. H of (Bannick et al. 2025) for a discussion of the merits of each type of variance estimator. Briefly, we recommend variance types 1 generally, and variance type 3 if it is anticipated that the distribution of X varies substantially over treatment groups.

Value

If ‘contrast_h‘ argument is used, outputs a ‘main‘ and a ‘contrast‘ object. The ‘main‘ object has the following structure:

`result`	A dplyr::tibble() with the treatment label, treatment mean estimate using AIPW, estimated SE, and p-value based on a z-test with estimate and SE.
`varcov`	The variance-covariance matrix for the treatment mean estimates.
`settings`	List of model settings used in covariate adjustment.
`original_df`	The original dataset provided by the user.
`mod`	The fit from the glm() working model used for covariate adjustment.
`mu_a`	Predicted potential outcomes for each treatment category (columns) and individual (rows). These are the $\hat{\mu}_a$

.

`g.estimate`	The G-computation estimate based only on $\frac{1}{n}\sum_{i=1}^{n}\hat{\mu}_a(X_i)$. This is equivalent to the AIPW estimate when a canonical link function is used.
`data`	Attributes about the dataset.

The ‘contrast‘ object has a structure that is documented in RobinCar::robincar_contrast().

References

Bannick MS, Shao J, Liu J, Du Y, Yi Y, Ye T (2025). “A General Form of Covariate Adjustment in Clinical Trials under Covariate-Adaptive Randomization.” *Biometrika*, asaf029. ISSN 1464-3510, doi:10.1093/biomet/asaf029.

`robincar_linear` *Covariate adjustment using linear working model*

Description

Estimate treatment-group-specific response means and (optionally) treatment group contrasts using a linear working model for continuous outcomes.

Usage

```
robincar_linear(
  df,
  treat_col,
  response_col,
  car_strata_cols = NULL,
  covariate_cols = NULL,
  car_scheme = "simple",
  adj_method = "ANOVA",
  contrast_h = NULL,
  contrast_dh = NULL
)
```

Arguments

`df`	A data.frame with the required columns
`treat_col`	Name of column in df with treatment variable
`response_col`	Name of the column in df with response variable
`car_strata_cols`	Names of columns in df with car_strata variables
`covariate_cols`	Names of columns in df with covariate variables. **If you want to include the strata variables as covariates also, add them here.**
`car_scheme`	Name of the type of covariate-adaptive randomization scheme. One of: "simple", "pocock-simon", "biased-coin", "permuted-block".
`adj_method`	Name of linear adjustment method to use. One of: "ANOVA", "ANCOVA", "ANHECOVA".
`contrast_h`	An optional function to specify a desired contrast
`contrast_dh`	An optional jacobian function for the contrast (otherwise use numerical derivative)

Details

* Adjustment method "ANOVA" fits a linear model with formula ‘Y ~ A‘ where ‘A‘ is the treatment group indicator and ‘Y‘ is the response. * "ANCOVA" fits a linear model with ‘Y ~ A + X‘ where ‘X‘ are the variables specified in the ‘covariate_cols‘ argument. * "ANHECOVA" fits a linear model with ‘Y ~ A*X‘, the main effects and treatment-by-covariate interactions.

Value

See value of RobinCar::robincar_glm(), this function is a wrapper using a linear link function.

`robincar_logrank` *Robust (potentially stratified) logrank adjustment*

Description

Perform a robust covariate-adjusted logrank test ("CL") that can be stratified ("CSL") if desired.

Usage

```
robincar_logrank(adj_method, ...)
```

Arguments

`adj_method`	Adjustment method, one of "CL", "CSL"
`...`	Additional arguments to ‘robincar_tte‘

Details

Note: Since RobinCar version 0.4.0, the variance of the test statistic has changed to better accommodate tied event times.

$\hat{\sigma}^2_{\text{CL}}$ and $\hat{\sigma}^2_{\text{CSL}}$ for the covariate-adjusted stratified log-rank test are given by Ye, Shao, and Yi (2024) after equation (8) on page 700, with $\hat{\sigma}^2_{\text{SL}}$ replaced by the following estimator, which is the standard denominator of the logrank test:

$$\frac{1}{n}\sum_j \sum_i v_j(t_i)$$

$$v_j(t_i) = \frac{Y_{0,j}(t)Y_{1,j}(t)d_j(t)\left[Y_j(t) - d_j(t)\right]}{Y_j(t)^2\left[Y_j(t) - 1\right]}$$

where t_i are strata-specific unique failure times, $d_j(t)$ is the number of events at time t in strata j, $Y_j(t)$ is the number at risk within strata j at time t, and $Y_{a,j}(t)$ is the number at risk within strata j and treatment a at time t.

Please see Ye, Shao, and Yi (2024)'s "Covariate-adjusted log-rank test: guaranteed efficiency gain and universal applicability" in *Biometrika* for more details about $\hat{\sigma}^2_{\text{CSL}}$.

Value

A result object with the following attributes:

`result`	A list: "statistic" is the adjusted logrank test statistic which can be used to obtain p-values; "U" and "se" are the numerator and denominator of the test statistic, respectively; if the input parameter "return_influence" is set to TRUE, the list will contain an element called "inf_func", which is a data.frame with columns "id" and "inf_func", which is the influence function for each subject for the covariate-adjusted log-rank test (stratified or unstratified). This influence function can be used, e.g., to calculate the correlation between interim and final test statistics in a group sequential design.
`settings`	The covariate adjustment settings used.
`original_df`	The dataset supplied by the user.

Examples

```
library(magrittr)
library(dplyr)
library(forcats)
set.seed(0)
n=100
data.simu0=data_gen(n=n,
                    theta=0,
                    randomization="permuted_block",
                    p_trt=0.5,
                   case="case2") %>% mutate(strata1=sample(letters[1:3],n,replace=TRUE),
                                           strata2=sample(LETTERS[4:5],n,replace=TRUE))

out <- robincar_logrank(df=data.simu0,
```

```
                        treat_col="I1",
                        p_trt=0.5,
                        ref_arm=0,
                        response_col="t",
                        event_col="delta",
                        covariate_cols=c("model_z1", "model_z2"),
                        car_scheme="simple",
                        adj_method=c("CL"))

set.seed(0)
n=100
data.simu0=data_gen(n=n,
                    theta=0,
                    randomization="permuted_block",
                    p_trt=0.5,
                    case="case1")

data.simu <- data.simu0 %>%
  tidyr::pivot_longer(cols=starts_with("car_strata"),
                      names_prefix="car_strata",
                      names_to="strt") %>%
  filter(value==1) %>% select(-value) %>%
  mutate(strt=forcats::as_factor(strt)) %>%
  select(t,strt) %>%
  left_join(data.simu0, .)

out1 <- robincar_logrank(df=data.simu,
                         treat_col="I1",
                         p_trt=0.5,
                         ref_arm=0,
                         response_col="t",
                         event_col="delta",
                         car_strata_cols="strt",
                         covariate_cols=NULL,
                         car_scheme=c("permuted-block"),
                         adj_method=c("CSL")
)
```

`robincar_mh` *Estimate Mantel-Haenszel Risk Difference*

Description

This function estimates Mantel-Haenszel risk difference and average treatment effect.

Usage

```
robincar_mh(
  df,
```

```
  treat_col,
  response_col,
  strata_cols,
  estimand = "ATE",
  ci_type = "mGR"
)
```

Arguments

`df`	A data.frame with the required columns
`treat_col`	Name of column in df with treatment variable. Must be binary
`response_col`	Name of the column in df with response variable
`strata_cols`	Names of columns in df with strata variables
`estimand`	A character string specifying the estimand. One of "MH" or "ATE" (default). See Details
`ci_type`	A character string specifying the type of confidence interval. One of "GR", "mGR" (default), "Sato"

Details

The estimand of interest can be either Mantel-Haenszel risk difference or Average Treatment Effect (ATE). The latter is the default option of ‘estimand‘. When ‘estimand="ATE"‘, ‘ci_type‘ is limited to the modified Greenland variance estimator (mGR). Otherwise, Greenland’s variance estimator (GR) and Sato’s variance estimator are optional.

Examples

```
df <- RobinCar:::data_sim
df$y_bin = ifelse(df$y>2.5, 1, 0)
robincar_mh(df = df[df$A!=2,],
            treat_col = "A",
            response_col = "y_bin",
            strata_cols = c("z1", "z2"),
            estimand = "MH",
            ci_type = "mGR")
```

`robincar_SL` *BETA: Covariate adjustment using working models from the super learner libraries through the AIPW package with cross-fitting.*

Description

Estimate treatment-group-specific response means and (optionally) treatment group contrasts using a generalized linear working model.

Usage

```
robincar_SL(
  df,
  treat_col,
  response_col,
  car_strata_cols = NULL,
  covariate_cols = NULL,
  car_scheme = "simple",
  covariate_to_include_strata = NULL,
  SL_libraries = c(),
  SL_learners = c(),
  k_split = 2,
  g_accuracy = 7,
  contrast_h = NULL,
  contrast_dh = NULL,
  variance_type = 1
)
```

Arguments

`df`	A data.frame with the required columns
`treat_col`	Name of column in df with treatment variable
`response_col`	Name of the column in df with response variable
`car_strata_cols`	Names of columns in df with car_strata variables
`covariate_cols`	Names of columns in df with covariate variables
`car_scheme`	Name of the type of covariate-adaptive randomization scheme. One of: "simple", "pocock-simon", "biased-coin", "permuted-block".
`covariate_to_include_strata`	Whether to include car_strata variables in covariate adjustment. Defaults to F for ANOVA and ANCOVA; defaults to T for ANHECOVA. User may override by passing in this argument.
`SL_libraries`	Vector of super-learner libraries to use for the covariate adjustment (see SuperLearner::listWrappers)
`SL_learners`	Optional list of super-learner "learners" to use for the covariate adjustment (see SuperLearner::create.Learner())
`k_split`	Number of splits to use in cross-fitting
`g_accuracy`	Level of accuracy to check prediction un-biasedness (in digits).
`contrast_h`	An optional function to specify a desired contrast
`contrast_dh`	An optional jacobian function for the contrast (otherwise use numerical derivative)
`variance_type`	The type of variance estimator to use, type 1, 2, or 3. All three are asymptotically equivalent. See details of RobinCar::robincar_glm for more.

Details

WARNING: This function is still under development and has not been extensively tested. This function currently only works for two treatment groups. Before using this function, you must load the SuperLearner library with 'library(SuperLearner)', otherwise the function call will fail.

Value

See value of RobinCar::robincar_glm, but the working model for $\hat{\mu}(X_i)$ is based on the AIPW::AIPW package that uses specified SuperLearner libraries and cross-fitting. Also, 'mod' attribute is an object of class AIPW::AIPW.

Examples

```
library(SuperLearner)
library(ranger)
n <- 1000
set.seed(10)
DATA2 <- data.frame(A=rbinom(n, size=1, prob=0.5),
                    y=rbinom(n, size=1, prob=0.2),
                    x1=rnorm(n),
                    x2=rnorm(n),
                    x3=as.factor(rbinom(n, size=1, prob=0.5)),
                    z1=rbinom(n, size=1, prob=0.5),
                    z2=rbinom(n, size=1, prob=0.5))
DATA2[, "y"] <- NA
As <- DATA2$A == 1
DATA2[DATA2$A == 1, "y"] <- rbinom(
  sum(As),
  size=1,
  prob=exp(DATA2[As,]$x1)/(1+exp(DATA2[As,]$x1)))
DATA2[DATA2$A == 0, "y"] <- rbinom(
  n-sum(As),
  size=1,
  prob=exp(1 +
    5*DATA2[!As,]$x1 + DATA2[!As,]$x2)/
    (1+exp(1 + 5*DATA2[!As,]$x1 + DATA2[!As,]$x2)))
DATA2$A <- as.factor(DATA2$A)

sl.mod <- robincar_SL(
  df=DATA2,
  response_col="y",
  treat_col="A",
  car_strata_cols=c("z1"),
  covariate_cols=c("x1"),
  SL_libraries=c("SL.ranger"),
  car_scheme="permuted-block",
  covariate_to_include_strata=TRUE
)

sl.mod$result
```

`robincar_SL_median` *BETA: Covariate adjustment using working models from the super learner libraries through the AIPW package with cross-fitting, with median adjustment.*

Description

Estimate treatment-group-specific response means and (optionally) treatment group contrasts using a generalized linear working model. Perform median adjustment to limit randomness induced from cross-fitting.

Usage

```
robincar_SL_median(
  n_times,
  seed,
  df,
  treat_col,
  response_col,
  car_strata_cols = NULL,
  covariate_cols = NULL,
  car_scheme = "simple",
  covariate_to_include_strata = NULL,
  SL_libraries = c(),
  SL_learners = c(),
  k_split = 2,
  g_accuracy = 7,
  contrast_h = NULL,
  contrast_dh = NULL
)
```

Arguments

`n_times`	Number of times to run the robincar_SL function
`seed`	Seed to set before running the set of functions
`df`	A data.frame with the required columns
`treat_col`	Name of column in df with treatment variable
`response_col`	Name of the column in df with response variable
`car_strata_cols`	Names of columns in df with car_strata variables
`covariate_cols`	Names of columns in df with covariate variables
`car_scheme`	Name of the type of covariate-adaptive randomization scheme. One of: "simple", "pocock-simon", "biased-coin", "permuted-block".

`covariate_to_include_strata`	Whether to include car_strata variables in covariate adjustment. Defaults to F for ANOVA and ANCOVA; defaults to T for ANHECOVA. User may override by passing in this argument.
`SL_libraries`	Vector of super-learner libraries to use for the covariate adjustment (see SuperLearner::listWrappers)
`SL_learners`	Optional list of super-learner "learners" to use for the covariate adjustment (see SuperLearner::create.Learner())
`k_split`	Number of splits to use in cross-fitting
`g_accuracy`	Level of accuracy to check prediction un-biasedness (in digits).
`contrast_h`	An optional function to specify a desired contrast
`contrast_dh`	An optional jacobian function for the contrast (otherwise use numerical derivative)

Details

WARNING: This function is still under development and has not been extensively tested. This function currently only works for two treatment groups. Before using this function, you must load the SuperLearner library with ‘library(SuperLearner)‘, otherwise the function call will fail.

Value

See value of RobinCar::robincar_SL. Attributes ‘mods‘ and ‘mu_as‘ are lists of ‘mod‘ and ‘mu_a‘ attributes, respectively, for each replicate of ‘robincar_SL‘ used in the median.

`robincar_tte` *Covariate adjustment for time to event data*

Description

Perform a covariate-adjusted logrank test (‘adj_method="CL"‘), covariate-adjusted stratified logrank test (‘adj_method="CSL"‘), or a covariate-adjusted robust Cox score test (‘adj_method="coxscore"‘).

Usage

```
robincar_tte(
  df,
  treat_col,
  response_col,
  event_col,
  adj_method,
  car_strata_cols = NULL,
  covariate_cols = NULL,
  p_trt = 0.5,
  ref_arm = NULL,
  sparse_remove = TRUE,
```

```
  car_scheme = "simple",
  return_influence = FALSE,
  id_col = NULL
)
```

Arguments

`df`	A data.frame with the required columns
`treat_col`	Name of column in df with treatment variable
`response_col`	Name of the column in df with response variable
`event_col`	Name of column in df with event indicator (0/FALSE=no event, 1/TRUE=event)
`adj_method`	Adjustment method (one of "CL", "CSL", or "coxscore")
`car_strata_cols`	Names of columns in df with car_strata variables
`covariate_cols`	Names of columns in df with covariate variables
`p_trt`	Treatment allocation ratio for the treatment arm.
`ref_arm`	Reference arm of the treatment group, defaults to NULL, which results in using the first element of ‘unique(data[, treat_col])‘.
`sparse_remove`	Remove sparse car_strata from calculation
`car_scheme`	Name of the type of covariate-adaptive randomization scheme. One of: "simple", "pocock-simon", "biased-coin", "permuted-block".
`return_influence`	Return the influence function for the covariate-adjusted log-rank test for each individual
`id_col`	Name of the column in df indicating individual ids

Details

‘robincar_coxscore‘ and ‘robincar_logrank‘ are wrapper functions around ‘robincar_tte‘.

Value

For adjustment method "CL" or "CSL", see value of RobinCar::robincar_logrank(); for adjustment method "coxscore" see value of RobinCar::robincar_coxscore().

Index