Package 'maicplus'

February 21, 2025

Title Matching Adjusted Indirect Comparison

Version 0.1.2

Description Facilitates performing matching adjusted indirect comparison (MAIC) analysis where the endpoint of interest is either time-to-event (e.g. overall survival) or binary (e.g. objective tumor response). The method is described by Signorovitch et al (2012) <doi:10.1016/j.jval.2012.05.004>.

License Apache License 2.0

URL https://github.com/hta-pharma/maicplus/, https://hta-pharma.github.io/maicplus/

BugReports https://github.com/hta-pharma/maicplus/issues

Depends R (>= 4.1)

Imports graphics, grDevices, stats, survival, lubridate, matrixStats, MASS, boot, stringr, lmtest, sandwich

Suggests knitr, testthat (>= 2.0), ggplot2, rmarkdown, dplyr, survminer, flexsurv, tibble, vdiffr, checkmate

VignetteBuilder knitr

Encoding UTF-8

Language en-US

LazyData true

RoxygenNote 7.3.2

Config/testthat/edition 3

NeedsCompilation no

Author Gregory Chen [aut], Michael Seo [aut], Isaac Gravestock [aut, cre], Miranta Antoniou [ctb], Chrysostomos Kalyvas [ctb], MSD, Inc. [cph, fnd], F. Hoffmann-La Roche AG [cph, fnd]

Maintainer Isaac Gravestock <isaac.gravestock@roche.com>

Repository CRAN

Date/Publication 2025-02-21 16:00:02 UTC

Contents

adrs_sat *Binary outcome data from single arm trial*

Description

Binary outcome data from single arm trial

Usage

```
adrs_sat
```

Format

A data frame with 500 rows and 5 columns:

USUBJID Unique subject identifiers for patients.

ARM Assigned treatment arm.

AVAL Analysis value, in this dataset an indicator of response.

PARAM Parameter type of `AVAL`.

RESPONSE Indicator of response.

See Also

Other unanchored datasets: `adsl_sat`, `adtte_sat`, `agd`, `centered_ipd_sat`, `pseudo_ipd_sat`, `weighted_sat`

adrs_twt *Binary outcome data from two arm trial*

Description

Binary outcome data from two arm trial

Usage

```
adrs_twt
```

Format

A data frame with 1000 rows and 5 columns:

USUBJID Unique subject identifiers for patients.

ARM Assigned treatment arm, `"A"`, `"C"`.

AVAL Analysis value, in this dataset an indicator of response.

PARAM Parameter type of `AVAL`.

RESPONSE Indicator of response.

See Also

Other anchored datasets: adsl_twt, adtte_twt, agd, centered_ipd_twt, pseudo_ipd_twt, weighted_twt

---

adsl_sat *Patient data from single arm study*

Description

Patient data from single arm study

Usage

```
adsl_sat
```

Format

a data frame with 500 rows and 8 columns:

USUBJID Unique subject identifiers for patients.

ARM Assigned treatment arm.

AGE Age in years at baseline.

SEX Sex of patient recorded as character "Male"/"Female".

SMOKE Smoking status at baseline as integer 1/0.

ECOG0 Indicator of ECOG score = 0 at baseline as integer 1/0.

N_PR_THER Number of prior therapies received as integer 1, 2, 3, 4.

SEX_MALE Indicator of SEX == "Male" as numeric 1/0.

See Also

Other unanchored datasets: adrs_sat, adtte_sat, agd, centered_ipd_sat, pseudo_ipd_sat, weighted_sat

adsl_twt *Patient data from two arm trial*

Description

Patient data from two arm trial

Usage

```
adsl_twt
```

Format

A data frame with 1000 rows and 8 columns:

USUBJID Unique subject identifiers for patients.

ARM Assigned treatment arm.

AGE Age in years at baseline.

SEX Sex of patient recorded as character "Male"/"Female"

SMOKE Smoking status at baseline as integer `1`/`0`.

ECOG0 Indicator of ECOG score = 0 at baseline as integer `1`/`0`.

N_PR_THER Number of prior therapies received as integer `1, 2, 3, 4`.

SEX_MALE Indicator of SEX == "Male" as numeric 1/0

See Also

Other anchored datasets: `adrs_twt`, `adtte_twt`, `agd`, `centered_ipd_twt`, `pseudo_ipd_twt`, `weighted_twt`

adtte_sat *Survival data from single arm trial*

Description

Survival data from single arm trial

Usage

```
adtte_sat
```

Format

A data frame with 500 rows and 10 columns:

USUBJID Unique subject identifiers for patients.
ARM Assigned treatment arm, `"A"`.
AVAL Analysis value which in this dataset overall survival time in days.
AVALU Unit of `AVAL`.
PARAMCD Paramater code of `AVAL`, `"OS"`.
PARAM Parameter name of `AVAL`, `"Overall Survival`.
CNSR Censoring indicator `0`/`1`.
TIME Survival time in days.
EVENT Event indicator `0`/`1`.

See Also

Other unanchored datasets: `adrs_sat`, `adsl_sat`, `agd`, `centered_ipd_sat`, `pseudo_ipd_sat`, `weighted_sat`

`adtte_twt` *Survival data from two arm trial*

Description

Survival data from two arm trial

Usage

```
adtte_twt
```

Format

A data frame with 1000 rows and 10 columns:

USUBJID Unique subject identifiers for patients.
ARM Assigned treatment arm, `"A"`, `"C"`.
AVAL Analysis value which in this dataset overall survival time in days.
AVALU Unit of `AVAL`.
PARAMCD Parameter code of `AVAL`, `"OS"`.
PARAM Parameter name of `AVAL`, `"Overall Survival`.
CNSR Censoring indicator `0`/`1`.
TIME Survival time in days.
EVENT Event indicator `0`/`1`.

See Also

Other anchored datasets: `adrs_twt`, `adsl_twt`, `agd`, `centered_ipd_twt`, `pseudo_ipd_twt`, `weighted_twt`

agd *Aggregate effect modifier data from published study*

Description

This data is formatted to be used in `center_ipd()`.

Usage

```
agd
```

Format

A data frame with 3 rows and 9 columns:

STUDY The study name, Study_XXXX

ARM Study arm name or total

N Number of observations in study arm

AGE_MEAN Mean age in study arm

AGE_MEDIAN Median age in study arm

AGE_SD Standard deviation of age in study arm

SEX_MALE_COUNT Number of male patients

ECOG0_COUNT Number of patients with ECOG score = 0

SMOKE_COUNT Number of smokers

N_PR_THER_MEDIAN Median number of prior therapies

See Also

Other unanchored datasets: `adrs_sat`, `adsl_sat`, `adtte_sat`, `centered_ipd_sat`, `pseudo_ipd_sat`, `weighted_sat`

Other anchored datasets: `adrs_twt`, `adsl_twt`, `adtte_twt`, `centered_ipd_twt`, `pseudo_ipd_twt`, `weighted_twt`

basic_kmplot *Basic Kaplan Meier (KM) plot function*

Description

This function can generate a basic KM plot with or without risk set table appended at the bottom. In a single plot, it can include up to 4 KM curves. This depends on number of levels in 'treatment' column in the input data.frame kmdat

Usage

```
basic_kmplot(
  kmdat,
  endpoint_name = "Time to Event Endpoint",
  time_scale = NULL,
  time_grid = NULL,
  show_risk_set = TRUE,
  main_title = "Kaplan-Meier Curves",
  subplot_heights = NULL,
  suppress_plot_layout = FALSE,
  use_colors = NULL,
  use_line_types = NULL,
  use_pch_cex = 0.65,
  use_pch_alpha = 100
)
```

Arguments

kmdat	a data.frame, must consist treatment, time (unit in days), n.risk, censor, surv, similar to an output from maicplus:::survfit_makeup
endpoint_name	a string, name of time to event endpoint, to be show in the last line of title
time_scale	a string, time unit of median survival time, taking a value of 'years', 'months', 'weeks' or 'days'
time_grid	a numeric vector in the unit of time_scale, risk set table and x axis of the km plot will be defined based on this time grid
show_risk_set	logical, show risk set table or not, TRUE by default
main_title	a string, main title of the KM plot
subplot_heights	a numeric vector, heights argument to graphic::layout(),NULL by default which means user will use the default setting
suppress_plot_layout	logical, suppress the layout setting in this function so that user can specify layout outside of the function, FALSE by default
use_colors	a character vector of length up to 4, colors to the KM curves, it will be passed to col of lines()

use_line_types a numeric vector of length up to 4, line type to the KM curves, it will be passed to lty of lines()

use_pch_cex a scalar between 0 and 1, point size to indicate censored individuals on the KM curves, it will be passed to cex of points()

use_pch_alpha a scalar between 0 and 255, degree of color transparency of points to indicate censored individuals on the KM curves, it will be passed to cex of points()

Value

a KM plot with or without risk set table appended at the bottom, with up to 4 KM curves

Examples

```
library(survival)
data(adtte_sat)
data(pseudo_ipd_sat)

combined_data <- rbind(adtte_sat[, c("TIME", "EVENT", "ARM")], pseudo_ipd_sat)
kmobj <- survfit(Surv(TIME, EVENT) ~ ARM, combined_data, conf.type = "log-log")
kmdat <- do.call(rbind, survfit_makeup(kmobj))
kmdat$treatment <- factor(kmdat$treatment)

# without risk set table
basic_kmplot(kmdat,
  time_scale = "month",
  time_grid = seq(0, 20, by = 2),
  show_risk_set = FALSE,
  main_title = "Kaplan-Meier Curves",
  subplot_heights = NULL,
  suppress_plot_layout = FALSE,
  use_colors = NULL,
  use_line_types = NULL
)

# with risk set table
basic_kmplot(kmdat,
  time_scale = "month",
  time_grid = seq(0, 20, by = 2),
  show_risk_set = TRUE,
  main_title = "Kaplan-Meier Curves",
  subplot_heights = NULL,
  suppress_plot_layout = FALSE,
  use_colors = NULL,
  use_line_types = NULL
)
```

basic_kmplot2 *Basic Kaplan Meier (KM) plot function using ggplot*

Description

This function generates a basic KM plot using ggplot.

Usage

```
basic_kmplot2(
  kmlist,
  kmlist_name,
  endpoint_name = "Time to Event Endpoint",
  show_risk_set = TRUE,
  main_title = "Kaplan-Meier Curves",
  break_x_by = NULL,
  censor = TRUE,
  xlab = "Time",
  xlim = NULL,
  use_colors = NULL,
  use_line_types = NULL
)
```

Arguments

kmlist	a list of survfit object
kmlist_name	a vector indicating the treatment names of each survfit object
endpoint_name	a string, name of time to event endpoint, to be show in the last line of title
show_risk_set	logical, show risk set table or not, TRUE by default
main_title	a string, main title of the KM plot
break_x_by	bin parameter for survminer
censor	indicator to include censor information
xlab	label name for x-axis of the plot
xlim	x limit for the x-axis of the plot
use_colors	a character vector of length up to 4, colors to the KM curves, it will be passed to 'col' of lines()
use_line_types	a numeric vector of length up to 4, line type to the KM curves, it will be passed to lty of lines()

Value

A Kaplan-Meier plot object created with survminer::ggsurvplot().

Examples

```
library(survival)
data(adtte_sat)
data(pseudo_ipd_sat)

kmobj_A <- survfit(Surv(TIME, EVENT) ~ ARM,
  data = adtte_sat,
  conf.type = "log-log"
)

kmobj_B <- survfit(Surv(TIME, EVENT) ~ ARM,
  data = pseudo_ipd_sat,
  conf.type = "log-log"
)

kmlist <- list(kmobj_A = kmobj_A, kmobj_B = kmobj_B)
kmlist_name <- c("A", "B")

basic_kmplot2(kmlist, kmlist_name)
```

---

bucher *Bucher method for combining treatment effects*

Description

Given two treatment effects of A vs. C and B vs. C derive the treatment effects of A vs. B using the Bucher method. Two-sided confidence interval and Z-test p-value are also calculated. Treatment effects and standard errors should be in log scale for hazard ratio, odds ratio, and risk ratio. Treatment effects and standard errors should be in natural scale for risk difference and mean difference.

Usage

```
bucher(trt, com, conf_lv = 0.95)

## S3 method for class 'maicplus_bucher'
print(x, ci_digits = 2, pval_digits = 3, exponentiate = FALSE, ...)
```

Arguments

`trt`	a list of two scalars for the study with the experimental arm. `'est'` is the point estimate and `'se'` is the standard error of the treatment effect. For time-to-event data, `'est'` and `'se'` should be point estimate and standard error of the log hazard ratio. For binary data, `'est'` and `'se'` should be point estimate and standard error of the log odds ratio, log risk ratio, or risk difference. For continuous data, `'est'` and `'se'` should be point estimate and standard error of the mean difference.
`com`	same as `trt`, but for the study with the control arm

conf_lv	a numerical scalar, prescribe confidence level to derive two-sided confidence interval for the treatment effect
x	maicplus_bucher object
ci_digits	an integer, number of decimal places for point estimate and derived confidence limits
pval_digits	an integer, number of decimal places to display Z-test p-value
exponentiate	whether the treatment effect and confidence interval should be exponentiated. This applies to relative treatment effects. Default is set to false.
...	not used

Value

a list with 5 elements,

est a scalar, point estimate of the treatment effect

se a scalar, standard error of the treatment effect

ci_l a scalar, lower confidence limit of a two-sided CI with prescribed nominal level by conf_lv

ci_u a scalar, upper confidence limit of a two-sided CI with prescribed nominal level by conf_lv

pval p-value of Z-test, with null hypothesis that est is zero

Methods (by generic)

- print(maicplus_bucher): Print method for maicplus_bucher objects

Examples

```
trt <- list(est = log(1.1), se = 0.2)
com <- list(est = log(1.3), se = 0.18)
result <- bucher(trt, com, conf_lv = 0.9)
print(result, ci_digits = 3, pval_digits = 3)
```

---

centered_ipd_sat *Centered patient data from single arm trial*

Description

Centered patient data from single arm trial

Usage

```
centered_ipd_sat
```

Format

A data frame with 500 rows and 14 columns:

USUBJID Unique subject identifiers for patients.

ARM Assigned treatment arm.

AGE Age in years at baseline.

SEX Sex of patient recorded as character `"Male"`/`"Female"`.

SMOKE Smoking status at baseline as integer `1`/`0`.

ECOG0 Indicator of ECOG score = 0 at baseline as integer `1`/`0`.

N_PR_THER Number of prior therapies received as integer `1, 2, 3, 4`.

SEX_MALE Indicator of `SEX == "Male"` as numeric `1`/`0`.

AGE_CENTERED Age in years at baseline relative to average in aggregate data agd.

AGE_MEDIAN_CENTERED `AGE` greater/less than `MEDIAN_AGE` in agd coded as `1`/`0` and then centered at 0.5.

AGE_SQUARED_CENTERED `AGE` squared and centered with respect to the `AGE` in agd. The squared age in the aggregate data is derived from the $E(X^2)$ term in the variance formula.

SEX_MALE_CENTERED `SEX_MALE` centered by the proportion of male patients in agd

ECOG0_CENTERED `ECOG0` centered by the proportion of `ECOG0` in agd

SMOKE_CENTERED `SMOKE` centered by the proportion of `SMOKE` in agd

N_PR_THER_MEDIAN_CENTERED `N_PR_THER` centered by the median in agd.

See Also

Other unanchored datasets: `adrs_sat`, `adsl_sat`, `adtte_sat`, `agd`, `pseudo_ipd_sat`, `weighted_sat`

`centered_ipd_twt` *Centered patient data from two arm trial*

Description

Centered patient data from two arm trial

Usage

```
centered_ipd_twt
```

Format

A data frame with 1000 rows and 14 columns:

USUBJID Unique subject identifiers for patients.

ARM Assigned treatment arm.

AGE Age in years at baseline.

SEX Sex of patient recorded as character `"Male"/"Female"`.

SMOKE Smoking status at baseline as integer `1/0`.

ECOG0 Indicator of ECOG score = 0 at baseline as integer `1/0`.

N_PR_THER Number of prior therapies received as integer `1, 2, 3, 4`.

SEX_MALE Indicator of `SEX == "Male"` as numeric `1/0`.

AGE_CENTERED Age in years at baseline relative to average in aggregate data agd.

AGE_MEDIAN_CENTERED `AGE` greater/less than `MEDIAN_AGE` in agd coded as `1/0` and then centered at 0.5.

AGE_SQUARED_CENTERED `AGE` squared and centered with respect to the `AGE` in agd. The squared age in the aggregate data is derived from the $E(X^2)$ term in the variance formula.

SEX_MALE_CENTERED `SEX_MALE` centered by the proportion of male patients in agd

ECOG0_CENTERED `ECOG0` centered by the proportion of `ECOG0` in agd

SMOKE_CENTERED `SMOKE` centered by the proportion of `SMOKE` in agd

N_PR_THER_MEDIAN_CENTERED `N_PR_THER` centered by the median in agd.

See Also

Other anchored datasets: `adrs_twt`, `adsl_twt`, `adtte_twt`, `agd`, `pseudo_ipd_twt`, `weighted_twt`

---

center_ipd *Center individual patient data (IPD) variables using aggregate data averages*

Description

This function subtracts IPD variables (prognostic variables and/or effect modifiers) by the aggregate data averages. This centering is needed in order to calculate weights. IPD and aggregate data variable names should match.

Usage

```
center_ipd(ipd, agd)
```

Arguments

ipd	IPD variable names should match the aggregate data names without the suffix. This would involve either changing the aggregate data name or the ipd name. For instance, if we binarize SEX variable with MALE as a reference using dummize_ipd, function names the new variable as SEX_MALE. In this case, SEX_MALE should also be available in the aggregate data.
agd	pre-processed aggregate data which contain STUDY, ARM, and N. Variable names should be followed by legal suffixes (i.e. MEAN, MEDIAN, SD, or PROP). Note that COUNT suffix is no longer accepted.

Value

centered ipd using aggregate level data averages

Examples

```
data(adsl_sat)
data(agd)
agd <- process_agd(agd)
ipd_centered <- center_ipd(ipd = adsl_sat, agd = agd)
```

---

check_weights *Check to see if weights are optimized correctly*

Description

This function checks to see if the optimization is done properly by checking the covariate averages before and after adjustment. In case of ties when calculating median, we return the mean of the two numbers. For more details, see `ties` parameter in matrixStats::weightedMedian.

Usage

```
check_weights(weighted_data, processed_agd)

## S3 method for class 'maicplus_check_weights'
print(
  x,
  mean_digits = 2,
  prop_digits = 2,
  sd_digits = 3,
  digits = getOption("digits"),
  ...
)
```

Arguments

weighted_data	object returned after calculating weights using estimate_weights
processed_agd	a data frame, object returned after using process_agd or aggregated data following the same naming convention
x	object from check_weights
mean_digits	number of digits for rounding mean columns in the output
prop_digits	number of digits for rounding proportion columns in the output
sd_digits	number of digits for rounding mean columns in the output
digits	minimal number of significant digits, see print.default.
...	further arguments to print.data.frame

Value

data.frame of weighted and unweighted covariate averages of the IPD, average of aggregate data, and sum of inner products of covariate x_i and the weights ($exp(x_i\beta)$)

Methods (by generic)

- print(maicplus_check_weights): Print method for check_weights objects

Examples

```
data(weighted_sat)
data(agd)
check_weights(weighted_sat, process_agd(agd))
```

---

dummize_ipd *Create dummy variables from categorical variables in an individual patient data (ipd)*

Description

This is a convenient function to convert categorical variables into dummy binary variables. This would be especially useful if the variable has more than two factors. Note that the original variable is kept after a variable is dummized.

Usage

```
dummize_ipd(raw_ipd, dummize_cols, dummize_ref_level)
```

Arguments

raw_ipd	ipd data that contains variable to dummize
dummize_cols	vector of column names to binarize
dummize_ref_level	vector of reference level of the variables to binarize

Value

ipd with dummized columns

Examples

```
data(adsl_twt)
dummize_ipd(adsl_twt, dummize_cols = c("SEX"), dummize_ref_level = c("Male"))
```

`estimate_weights` *Derive individual weights in the matching step of MAIC*

Description

Assuming data is properly processed, this function takes individual patient data (IPD) with centered covariates (effect modifiers and/or prognostic variables) as input, and generates weights for each individual in IPD trial to match the covariates in aggregate data.

The plot function displays individuals weights with key summary in top right legend that includes median weight, effective sample size (ESS), and reduction percentage (what percent ESS is reduced from the original sample size). There are two options of plotting: base R plot and `ggplot`. The default for base R plot is to plot unscaled and scaled separately. The default for `ggplot` is to plot unscaled and scaled weights on a same plot.

Usage

```
estimate_weights(
  data,
  centered_colnames = NULL,
  start_val = 0,
  method = "BFGS",
  n_boot_iteration = NULL,
  set_seed_boot = 1234,
  boot_strata = "ARM",
  ...
)

## S3 method for class 'maicplus_estimate_weights'
plot(
  x,
  ggplot = FALSE,
  bin_col = "#6ECEB2",
  vline_col = "#688CE8",
  main_title = NULL,
  scaled_weights = TRUE,
  bins = 50,
  ...
)
```

Arguments

data	a numeric matrix, centered covariates of IPD, no missing value in any cell is allowed
centered_colnames	a character or numeric vector (column indicators) of centered covariates
start_val	a scalar, the starting value for all coefficients of the propensity score regression
method	a string, name of the optimization algorithm (see 'method' argument of base::optim()) The default is "BFGS", other options are "Nelder-Mead", "CG", "L-BFGS-B", "SANN", and "Brent"
n_boot_iteration	an integer, number of bootstrap iterations. By default is NULL which means bootstrapping procedure will not be triggered, and hence the element "boot" of output list object will be NULL.
set_seed_boot	a scalar, the random seed for conducting the bootstrapping, only relevant if n_boot_iteration is not NULL. By default, use seed 1234
boot_strata	a character vector of column names in data that defines the strata for bootstrapping. This ensures that samples are drawn proportionally from each defined stratum. If NULL, no stratification during bootstrapping process. By default, it is "ARM"
...	Additional control parameters passed to stats::optim.
x	object from estimate_weights
ggplot	indicator to print base weights plot or ggplot weights plot
bin_col	a string, color for the bins of histogram
vline_col	a string, color for the vertical line in the histogram
main_title	title of the plot. For ggplot, name of scaled weights plot and unscaled weights plot, respectively.
scaled_weights	(base plot only) an indicator for using scaled weights instead of regular weights
bins	(ggplot only) number of bin parameter to use

Value

a list with the following 4 elements,

data a data.frame, includes the input data with appended column 'weights' and 'scaled_weights'. Scaled weights has a summation to be the number of rows in data that has no missing value in any of the effect modifiers

centered_colnames column names of centered effect modifiers in data

nr_missing number of rows in data that has at least 1 missing value in specified centered effect modifiers

ess effective sample size, square of sum divided by sum of squares

opt R object returned by base::optim(), for assess convergence and other details

boot_strata 'strata' from a boot::boot object

boot_seed column names in data of the stratification factors

boot a n by 2 by k array or NA, where n equals to number of rows in data, and k equals n_boot_iteration. The 2 columns in the second dimension include a column of numeric indexes of the rows in data that are selected at a bootstrapping iteration and a column of weights. boot is NA when argument n_boot_iteration is set as NULL

Methods (by generic)

- plot(maicplus_estimate_weights): Plot method for estimate_weights objects

Examples

```
data(centered_ipd_sat)
centered_colnames <- grep("_CENTERED", colnames(centered_ipd_sat), value = TRUE)
weighted_data <- estimate_weights(data = centered_ipd_sat, centered_colnames = centered_colnames)

# To later estimate bootstrap confidence intervals, we calculate the weights
# for the bootstrap samples:
weighted_data_boot <- estimate_weights(
  data = centered_ipd_sat, centered_colnames = centered_colnames, n_boot_iteration = 100
)

plot(weighted_sat)

if (requireNamespace("ggplot2")) {
  plot(weighted_sat, ggplot = TRUE)
}
```

---

find_SE_from_CI *Calculate standard error from the reported confidence interval.*

Description

Comparator studies often only report confidence interval of the treatment effects. This function calculates standard error of the treatment effect given the reported confidence interval. For relative treatment effect (i.e. hazard ratio, odds ratio, and risk ratio), the function would log the confidence interval. For risk difference and mean difference, we do not log the confidence interval. The option to log the confidence interval is controlled by 'log' parameter.

Usage

```
find_SE_from_CI(CI_lower = NULL, CI_upper = NULL, CI_perc = 0.95, log = TRUE)
```

Arguments

CI_lower	Reported lower percentile value of the treatment effect
CI_upper	Reported upper percentile value of the treatment effect
CI_perc	Percentage of confidence interval reported
log	Whether the confidence interval should be logged. For relative treatment effect, log should be applied because estimated log treatment effect is approximately normally distributed.

Value

Standard error of log relative treatment effect if `'log'` is true and standard error of the treatment effect if `'log'` is false

Examples

```
find_SE_from_CI(CI_lower = 0.55, CI_upper = 0.90, CI_perc = 0.95)
```

`get_pseudo_ipd_binary` *Create pseudo IPD given aggregated binary data*

Description

Create pseudo IPD given aggregated binary data

Usage

```
get_pseudo_ipd_binary(binary_agd, format = c("stacked", "unstacked"))
```

Arguments

`binary_agd`	a data.frame that take different formats depending on `format`
`format`	a string, "stacked" or "unstacked"

Value

a data.frame of pseudo binary IPD, with columns USUBJID, ARM, RESPONSE

Examples

```
# example of unstacked
testdat <- data.frame(Yes = 280, No = 120)
rownames(testdat) <- "B"
get_pseudo_ipd_binary(
  binary_agd = testdat,
  format = "unstacked"
)

# example of stacked
get_pseudo_ipd_binary(
  binary_agd = data.frame(
    ARM = rep("B", 2),
    RESPONSE = c("YES", "NO"),
    COUNT = c(280, 120)
  ),
  format = "stacked"
)
```

get_time_as *Convert Time Values Using Scaling Factors*

Description

Convert Time Values Using Scaling Factors

Usage

```
get_time_as(times, as = NULL)
```

Arguments

times	Numeric time values
as	A time scale to convert to. One of "days", "weeks", "months", "years"

Value

Returns a numeric vector calculated from `times / get_time_conversion(factor = as)`

Examples

```
get_time_as(50, as = "months")
```

glm_makeup *Helper function to summarize outputs from glm fit*

Description

Helper function to summarize outputs from glm fit

Usage

```
glm_makeup(binobj, legend = "before matching", weighted = FALSE)
```

Arguments

binobj	returned object from `stats::glm`
legend	label to indicate the binary fit
weighted	logical flag indicating whether weights have been applied in the glm fit

Value

A `data.frame` containing a summary of the number of events and subjects in a logistic regression model.

Examples

```
data(adrs_sat)
pseudo_adrs <- get_pseudo_ipd_binary(
  binary_agd = data.frame(
    ARM = rep("B", 2),
    RESPONSE = c("YES", "NO"),
    COUNT = c(280, 120)
  ),
  format = "stacked"
)
pseudo_adrs$RESPONSE <- as.numeric(pseudo_adrs$RESPONSE)
combined_data <- rbind(adrs_sat[, c("USUBJID", "ARM", "RESPONSE")], pseudo_adrs)
combined_data$ARM <- as.factor(combined_data$ARM)
binobj_dat <- stats::glm(RESPONSE ~ ARM, combined_data, family = binomial("logit"))
glm_makeup(binobj_dat)
```

---

kmplot *Kaplan Meier (KM) plot function for anchored and unanchored cases*

Description

It is wrapper function of `basic_kmplot`. The argument setting is similar to `maic_anchored` and `maic_unanchored`, and it is used in those two functions.

Usage

```
kmplot(
  weights_object,
  tte_ipd,
  tte_pseudo_ipd,
  trt_ipd,
  trt_agd,
  trt_common = NULL,
  normalize_weights = FALSE,
  trt_var_ipd = "ARM",
  trt_var_agd = "ARM",
  km_conf_type = "log-log",
  km_layout = c("all", "by_trial", "by_arm"),
  ...
)
```

Arguments

`weights_object`	an object returned by `estimate_weight`
`tte_ipd`	a data frame of individual patient data (IPD) of internal trial, contain at least `"USUBJID"`, `"EVENT"`, `"TIME"` columns and a column indicating treatment assignment

tte_pseudo_ipd	a data frame of pseudo IPD by digitized KM curves of external trial (for time-to-event endpoint), contain at least "EVENT", "TIME"
trt_ipd	a string, name of the interested investigation arm in internal trial dat_igd (real IPD)
trt_agd	a string, name of the interested investigation arm in external trial dat_pseudo (pseudo IPD)
trt_common	a string, name of the common comparator in internal and external trial, by default is NULL, indicating unanchored case
normalize_weights	logical, default is FALSE. If TRUE, scaled_weights (normalized weights) in weights_object$data will be used.
trt_var_ipd	a string, column name in tte_ipd that contains the treatment assignment
trt_var_agd	a string, column name in tte_pseudo_ipd that contains the treatment assignment
km_conf_type	a string, pass to conf.type of survfit
km_layout	a string, only applicable for unanchored case (trt_common = NULL), indicated the desired layout of output KM curve.
...	other arguments in basic_kmplot

Value

In unanchored case, a KM plot with risk set table. In anchored case, depending on km_layout,

- if "by_trial", 2 by 1 plot, first all KM curves (incl. weighted) in IPD trial, and then KM curves in AgD trial, with risk set table.
- if "by_arm", 2 by 1 plot, first KM curves of trt_agd and trt_ipd (with and without weights), and then KM curves of trt_common in AgD trial and IPD trial (with and without weights). Risk set table is appended.
- if "all", 2 by 2 plot, all plots in "by_trial" and "by_arm" without risk set table appended.

Examples

```
# unanchored example using kmplot
data(weighted_sat)
data(adtte_sat)
data(pseudo_ipd_sat)

kmplot(
  weights_object = weighted_sat,
  tte_ipd = adtte_sat,
  tte_pseudo_ipd = pseudo_ipd_sat,
  trt_var_ipd = "ARM",
  trt_var_agd = "ARM",
  endpoint_name = "Overall Survival",
  trt_ipd = "A",
  trt_agd = "B",
  trt_common = NULL,
```

```
  km_conf_type = "log-log",
  time_scale = "month",
  time_grid = seq(0, 20, by = 2),
  use_colors = NULL,
  use_line_types = NULL,
  use_pch_cex = 0.65,
  use_pch_alpha = 100
)
# anchored example using kmplot
data(weighted_twt)
data(adtte_twt)
data(pseudo_ipd_twt)

# plot by trial
kmplot(
  weights_object = weighted_twt,
  tte_ipd = adtte_twt,
  tte_pseudo_ipd = pseudo_ipd_twt,
  trt_ipd = "A",
  trt_agd = "B",
  trt_common = "C",
  trt_var_ipd = "ARM",
  trt_var_agd = "ARM",
  endpoint_name = "Overall Survival",
  km_conf_type = "log-log",
  km_layout = "by_trial",
  time_scale = "month",
  time_grid = seq(0, 20, by = 2),
  use_colors = NULL,
  use_line_types = NULL,
  use_pch_cex = 0.65,
  use_pch_alpha = 100
)

# plot by arm
kmplot(
  weights_object = weighted_twt,
  tte_ipd = adtte_twt,
  tte_pseudo_ipd = pseudo_ipd_twt,
  trt_ipd = "A",
  trt_agd = "B",
  trt_common = "C",
  trt_var_ipd = "ARM",
  trt_var_agd = "ARM",
  endpoint_name = "Overall Survival",
  km_conf_type = "log-log",
  km_layout = "by_arm",
  time_scale = "month",
  time_grid = seq(0, 20, by = 2),
  use_colors = NULL,
  use_line_types = NULL,
  use_pch_cex = 0.65,
  use_pch_alpha = 100
```

```
)

# plot all
kmplot(
  weights_object = weighted_twt,
  tte_ipd = adtte_twt,
  tte_pseudo_ipd = pseudo_ipd_twt,
  trt_ipd = "A",
  trt_agd = "B",
  trt_common = "C",
  trt_var_ipd = "ARM",
  trt_var_agd = "ARM",
  endpoint_name = "Overall Survival",
  km_conf_type = "log-log",
  km_layout = "all",
  time_scale = "month",
  time_grid = seq(0, 20, by = 2),
  use_colors = NULL,
  use_line_types = NULL,
  use_pch_cex = 0.65,
  use_pch_alpha = 100
)
```

`kmplot2` *Kaplan-Meier (KM) plot function for anchored and unanchored cases using ggplot*

Description

This is wrapper function of `basic_kmplot2`. The argument setting is similar to `maic_anchored` and `maic_unanchored`, and it is used in those two functions.

Usage

```
kmplot2(
  weights_object,
  tte_ipd,
  tte_pseudo_ipd,
  trt_ipd,
  trt_agd,
  trt_common = NULL,
  normalize_weights = FALSE,
  trt_var_ipd = "ARM",
  trt_var_agd = "ARM",
  km_conf_type = "log-log",
  km_layout = c("all", "by_trial", "by_arm"),
  time_scale,
  ...
)
```

Arguments

weights_object	an object returned by estimate_weight
tte_ipd	a data frame of individual patient data (IPD) of internal trial, contain at least "USUBJID", "EVENT", "TIME" columns and a column indicating treatment assignment
tte_pseudo_ipd	a data frame of pseudo IPD by digitized KM curves of external trial (for time-to-event endpoint), contain at least "EVENT", "TIME"
trt_ipd	a string, name of the interested investigation arm in internal trial dat_igd (real IPD)
trt_agd	a string, name of the interested investigation arm in external trial dat_pseudo (pseudo IPD)
trt_common	a string, name of the common comparator in internal and external trial, by default is NULL, indicating unanchored case
normalize_weights	logical, default is FALSE. If TRUE, scaled_weights (normalized weights) in weights_object$data will be used.
trt_var_ipd	a string, column name in tte_ipd that contains the treatment assignment
trt_var_agd	a string, column name in tte_pseudo_ipd that contains the treatment assignment
km_conf_type	a string, pass to conf.type of survfit
km_layout	a string, only applicable for unanchored case (trt_common = NULL), indicated the desired layout of output KM curve.
time_scale	a string, time unit of median survival time, taking a value of 'years', 'months', weeks' or 'days'
...	other arguments in basic_kmplot2

Value

In unanchored case, a KM plot with risk set table. In anchored case, depending on km_layout,

- if "by_trial", 2 by 1 plot, first all KM curves (incl. weighted) in IPD trial, and then KM curves in AgD trial, with risk set table.
- if "by_arm", 2 by 1 plot, first KM curves of trt_agd and trt_ipd (with and without weights), and then KM curves of trt_common in AgD trial and IPD trial (with and without weights). Risk set table is appended.
- if "all", 2 by 2 plot, all plots in "by_trial" and "by_arm" without risk set table appended.

Examples

```
# unanchored example using kmplot2
data(weighted_sat)
data(adtte_sat)
data(pseudo_ipd_sat)

kmplot2(
```

```
  weights_object = weighted_sat,
  tte_ipd = adtte_sat,
  tte_pseudo_ipd = pseudo_ipd_sat,
  trt_ipd = "A",
  trt_agd = "B",
  trt_common = NULL,
  trt_var_ipd = "ARM",
  trt_var_agd = "ARM",
  endpoint_name = "Overall Survival",
  km_conf_type = "log-log",
  time_scale = "month",
  break_x_by = 2,
  xlim = c(0, 20)
)
# anchored example using kmplot2
data(weighted_twt)
data(adtte_twt)
data(pseudo_ipd_twt)

# plot by trial
kmplot2(
  weights_object = weighted_twt,
  tte_ipd = adtte_twt,
  tte_pseudo_ipd = pseudo_ipd_twt,
  trt_ipd = "A",
  trt_agd = "B",
  trt_common = "C",
  trt_var_ipd = "ARM",
  trt_var_agd = "ARM",
  endpoint_name = "Overall Survival",
  km_conf_type = "log-log",
  km_layout = "by_trial",
  time_scale = "month",
  break_x_by = 2
)

# plot by arm
kmplot2(
  weights_object = weighted_twt,
  tte_ipd = adtte_twt,
  tte_pseudo_ipd = pseudo_ipd_twt,
  trt_ipd = "A",
  trt_agd = "B",
  trt_common = "C",
  trt_var_ipd = "ARM",
  trt_var_agd = "ARM",
  endpoint_name = "Overall Survival",
  km_conf_type = "log-log",
  km_layout = "by_arm",
  time_scale = "month",
  break_x_by = 2
)
```

```
# plot all
kmplot2(
  weights_object = weighted_twt,
  tte_ipd = adtte_twt,
  tte_pseudo_ipd = pseudo_ipd_twt,
  trt_ipd = "A",
  trt_agd = "B",
  trt_common = "C",
  trt_var_ipd = "ARM",
  trt_var_agd = "ARM",
  endpoint_name = "Overall Survival",
  km_conf_type = "log-log",
  km_layout = "all",
  time_scale = "month",
  break_x_by = 2,
  xlim = c(0, 20),
  show_risk_set = FALSE
)
```

`maic_anchored` *Anchored MAIC for binary and time-to-event endpoint*

Description

This is a wrapper function to provide adjusted effect estimates and relevant statistics in anchored case (i.e. there is a common comparator arm in the internal and external trial).

Usage

```
maic_anchored(
  weights_object,
  ipd,
  pseudo_ipd,
  trt_ipd,
  trt_agd,
  trt_common,
  trt_var_ipd = "ARM",
  trt_var_agd = "ARM",
  normalize_weights = FALSE,
  endpoint_type = "tte",
  endpoint_name = "Time to Event Endpoint",
  eff_measure = c("HR", "OR", "RR", "RD"),
  boot_ci_type = c("norm", "basic", "stud", "perc", "bca"),
  time_scale = "months",
  km_conf_type = "log-log",
  binary_robust_cov_type = "HC3"
)
```

Arguments

weights_object	an object returned by estimate_weight
ipd	a data frame that meet format requirements in 'Details', individual patient data (IPD) of internal trial
pseudo_ipd	a data frame, pseudo IPD from digitized KM curve of external trial (for time-to-event endpoint) or from contingency table (for binary endpoint)
trt_ipd	a string, name of the interested investigation arm in internal trial ipd (internal IPD)
trt_agd	a string, name of the interested investigation arm in external trial pseudo_ipd (pseudo IPD)
trt_common	a string, name of the common comparator in internal and external trial
trt_var_ipd	a string, column name in ipd that contains the treatment assignment
trt_var_agd	a string, column name in ipd that contains the treatment assignment
normalize_weights	logical, default is FALSE. If TRUE, scaled_weights (normalized weights) in weights_object$data will be used.
endpoint_type	a string, one out of the following "binary", "tte" (time to event)
endpoint_name	a string, name of time to event endpoint, to be show in the last line of title
eff_measure	a string, "RD" (risk difference), "OR" (odds ratio), "RR" (relative risk) for a binary endpoint; "HR" for a time-to-event endpoint. By default is NULL, "OR" is used for binary case, otherwise "HR" is used.
boot_ci_type	a string, one of c("norm","basic", "stud", "perc", "bca") to select the type of bootstrap confidence interval. See boot::boot.ci for more details.
time_scale	a string, time unit of median survival time, taking a value of 'years', 'months', 'weeks' or 'days'. NOTE: it is assumed that values in TIME column of ipd and pseudo_ipd is in the unit of days
km_conf_type	a string, pass to conf.type of survfit
binary_robust_cov_type	a string to pass to argument type of sandwich::vcovHC, see possible options in the documentation of that function. Default is "HC3"

Details

It is required that input ipd and pseudo_ipd to have the following columns. This function is not sensitive to upper or lower case of letters in column names.

- USUBJID - character, unique subject ID
- ARM - character or factor, treatment indicator, column name does not have to be 'ARM'. User specify in trt_var_ipd and trt_var_agd

For time-to-event analysis, the follow columns are required:

- EVENT - numeric, 1 for censored/death, 0 otherwise
- TIME - numeric column, observation time of the EVENT; unit in days

For binary outcomes:

- RESPONSE - numeric, 1 for event occurred, 0 otherwise

Value

A list, contains 'descriptive' and 'inferential'

Examples

```
# Anchored example using maic_anchored for time-to-event data
data(weighted_twt)
data(adtte_twt)
data(pseudo_ipd_twt)

result_tte <- maic_anchored(
  weights_object = weighted_twt,
  ipd = adtte_twt,
  pseudo_ipd = pseudo_ipd_twt,
  trt_var_ipd = "ARM",
  trt_var_agd = "ARM",
  trt_ipd = "A",
  trt_agd = "B",
  trt_common = "C",
  endpoint_name = "Overall Survival",
  endpoint_type = "tte",
  eff_measure = "HR",
  time_scale = "month",
  km_conf_type = "log-log",
)
result_tte$descriptive$summary
result_tte$inferential$summary
# Anchored example using maic_anchored for binary outcome
data(weighted_twt)
data(adrs_twt)

# Reported summary data
pseudo_adrs <- get_pseudo_ipd_binary(
  binary_agd = data.frame(
    ARM = c("B", "C", "B", "C"),
    RESPONSE = c("YES", "YES", "NO", "NO"),
    COUNT = c(280, 120, 200, 200)
  ),
  format = "stacked"
)

# inferential result
result_binary <- maic_anchored(
  weights_object = weighted_twt,
  ipd = adrs_twt,
  pseudo_ipd = pseudo_adrs,
  trt_var_ipd = "ARM",
  trt_var_agd = "ARM",
  trt_ipd = "A",
  trt_agd = "B",
  trt_common = "C",
  endpoint_name = "Binary Event",
```

```
  endpoint_type = "binary",
  eff_measure = "OR"
)

result_binary$descriptive$summary
result_binary$inferential$summary
```

`maic_unanchored` *Unanchored MAIC for binary and time-to-event endpoint*

Description

This is a wrapper function to provide adjusted effect estimates and relevant statistics in unanchored case (i.e. there is no common comparator arm in the internal and external trial).

Usage

```
maic_unanchored(
  weights_object,
  ipd,
  pseudo_ipd,
  trt_ipd,
  trt_agd,
  trt_var_ipd = "ARM",
  trt_var_agd = "ARM",
  normalize_weights = FALSE,
  endpoint_type = "tte",
  endpoint_name = "Time to Event Endpoint",
  eff_measure = c("HR", "OR", "RR", "RD"),
  boot_ci_type = c("norm", "basic", "stud", "perc", "bca"),
  time_scale = "months",
  km_conf_type = "log-log",
  binary_robust_cov_type = "HC3"
)
```

Arguments

`weights_object`	an object returned by `estimate_weight`
`ipd`	a data frame that meet format requirements in 'Details', individual patient data (IPD) of internal trial
`pseudo_ipd`	a data frame, pseudo IPD from digitized KM curve of external trial (for time-to-event endpoint) or from contingency table (for binary endpoint)
`trt_ipd`	a string, name of the interested investigation arm in internal trial `dat_igd` (real IPD)
`trt_agd`	a string, name of the interested investigation arm in external trial `pseudo_ipd` (pseudo IPD)

`trt_var_ipd` a string, column name in `ipd` that contains the treatment assignment

`trt_var_agd` a string, column name in `ipd` that contains the treatment assignment

`normalize_weights` logical, default is `FALSE`. If `TRUE`, `scaled_weights` (normalized weights) in `weights_object$data` will be used.

`endpoint_type` a string, one out of the following "binary", "tte" (time to event)

`endpoint_name` a string, name of time to event endpoint, to be show in the last line of title

`eff_measure` a string, "RD" (risk difference), "OR" (odds ratio), "RR" (relative risk) for a binary endpoint; "HR" for a time-to-event endpoint. By default is `NULL`, "OR" is used for binary case, otherwise "HR" is used.

`boot_ci_type` a string, one of `c("norm","basic", "stud", "perc", "bca")` to select the type of bootstrap confidence interval. See boot::boot.ci for more details.

`time_scale` a string, time unit of median survival time, taking a value of 'years', 'months', 'weeks' or 'days'. NOTE: it is assumed that values in TIME column of `ipd` and `pseudo_ipd` is in the unit of days

`km_conf_type` a string, pass to `conf.type` of `survfit`

`binary_robust_cov_type` a string to pass to argument `type` of sandwich::vcovHC, see possible options in the documentation of that function. Default is `"HC3"`

Details

For time-to-event analysis, it is required that input `ipd` and `pseudo_ipd` to have the following columns. This function is not sensitive to upper or lower case of letters in column names.

- USUBJID - character, unique subject ID
- ARM - character or factor, treatment indicator, column name does not have to be 'ARM'. User specify in `trt_var_ipd` and `trt_var_agd`
- EVENT - numeric, 1 for censored/death, 0 for otherwise
- TIME - numeric column, observation time of the `EVENT`; unit in days

Value

A list, contains 'descriptive' and 'inferential'

Examples

```
#
# unanchored example using maic_unanchored for time-to-event data
#
data(centered_ipd_sat)
data(adtte_sat)
data(pseudo_ipd_sat)

#### derive weights
weighted_data <- estimate_weights(
  data = centered_ipd_sat,
```

```
  centered_colnames = grep("_CENTERED$", names(centered_ipd_sat)),
  start_val = 0,
  method = "BFGS"
)

weighted_data2 <- estimate_weights(
  data = centered_ipd_sat,
  centered_colnames = grep("_CENTERED$", names(centered_ipd_sat)),
  start_val = 0,
  method = "BFGS",
  n_boot_iteration = 100,
  set_seed_boot = 1234
)

# inferential result
result <- maic_unanchored(
  weights_object = weighted_data,
  ipd = adtte_sat,
  pseudo_ipd = pseudo_ipd_sat,
  trt_var_ipd = "ARM",
  trt_var_agd = "ARM",
  trt_ipd = "A",
  trt_agd = "B",
  endpoint_name = "Overall Survival",
  endpoint_type = "tte",
  eff_measure = "HR",
  time_scale = "month",
  km_conf_type = "log-log"
)
result$descriptive$summary
result$inferential$summary

result_boot <- maic_unanchored(
  weights_object = weighted_data2,
  ipd = adtte_sat,
  pseudo_ipd = pseudo_ipd_sat,
  trt_var_ipd = "ARM",
  trt_var_agd = "ARM",
  trt_ipd = "A",
  trt_agd = "B",
  endpoint_name = "Overall Survival",
  endpoint_type = "tte",
  eff_measure = "HR",
  time_scale = "month",
  km_conf_type = "log-log"
)
result$descriptive$summary
result$inferential$summary
#
# unanchored example using maic_unanchored for binary outcome
#

data(centered_ipd_sat)
```

```
data(adrs_sat)

centered_ipd_sat
centered_colnames <- grep("_CENTERED$", colnames(centered_ipd_sat), value = TRUE)
weighted_data <- estimate_weights(data = centered_ipd_sat, centered_colnames = centered_colnames)
weighted_data2 <- estimate_weights(
  data = centered_ipd_sat, centered_colnames = centered_colnames,
  n_boot_iteration = 100
)

# get dummy binary IPD
pseudo_adrs <- get_pseudo_ipd_binary(
  binary_agd = data.frame(
    ARM = rep("B", 2),
    RESPONSE = c("YES", "NO"),
    COUNT = c(280, 120)
  ),
  format = "stacked"
)

# unanchored binary MAIC, with CI based on sandwich estimator
maic_unanchored(
  weights_object = weighted_data,
  ipd = adrs_sat,
  pseudo_ipd = pseudo_adrs,
  trt_ipd = "A",
  trt_agd = "B",
  trt_var_ipd = "ARM",
  trt_var_agd = "ARM",
  endpoint_type = "binary",
  endpoint_name = "Binary Endpoint",
  eff_measure = "RR",
  # binary specific args
  binary_robust_cov_type = "HC3"
)

# unanchored binary MAIC, with bootstrapped CI
maic_unanchored(
  weights_object = weighted_data2,
  ipd = adrs_sat,
  pseudo_ipd = pseudo_adrs,
  trt_ipd = "A",
  trt_agd = "B",
  trt_var_ipd = "ARM",
  trt_var_agd = "ARM",
  endpoint_type = "binary",
  endpoint_name = "Binary Endpoint",
  eff_measure = "RR",
  # binary specific args
  binary_robust_cov_type = "HC3"
)

#---------------------------------
```

medSurv_makeup *Helper function to retrieve median survival time from a* `survival::survfit` *object*

Description

Extract and display median survival time with confidence interval

Usage

```
medSurv_makeup(km_fit, legend = "before matching", time_scale)
```

Arguments

`km_fit`	returned object from `survival::survfit`
`legend`	a character string, name used in 'type' column in returned data frame
`time_scale`	a character string, 'years', 'months', 'weeks' or 'days', time unit of median survival time

Value

a data frame with a index column 'type', median survival time and confidence interval

Examples

```
data(adtte_sat)
data(pseudo_ipd_sat)
library(survival)
combined_data <- rbind(adtte_sat[, c("TIME", "EVENT", "ARM")], pseudo_ipd_sat)
kmobj <- survfit(Surv(TIME, EVENT) ~ ARM, combined_data, conf.type = "log-log")

# Derive median survival time
medSurv <- medSurv_makeup(kmobj, legend = "before matching", time_scale = "day")
medSurv
```

ph_diagplot *Diagnosis plot of proportional hazard assumption for anchored and unanchored*

Description

Diagnosis plot of proportional hazard assumption for anchored and unanchored

Usage

```
ph_diagplot(
  weights_object,
  tte_ipd,
  tte_pseudo_ipd,
  trt_ipd,
  trt_agd,
  trt_common = NULL,
  trt_var_ipd = "ARM",
  trt_var_agd = "ARM",
  endpoint_name = "Time to Event Endpoint",
  time_scale,
  zph_transform = "log",
  zph_log_hazard = TRUE
)
```

Arguments

weights_object	an object returned by estimate_weight
tte_ipd	a data frame of individual patient data (IPD) of internal trial, contain at least "USUBJID", "EVENT", "TIME" columns and a column indicating treatment assignment
tte_pseudo_ipd	a data frame of pseudo IPD by digitized KM curves of external trial (for time-to-event endpoint), contain at least "EVENT", "TIME"
trt_ipd	a string, name of the interested investigation arm in internal trial tte_ipd (real IPD)
trt_agd	a string, name of the interested investigation arm in external trial tte_pseudo_ipd (pseudo IPD)
trt_common	a string, name of the common comparator in internal and external trial, by default is NULL, indicating unanchored case
trt_var_ipd	a string, column name in tte_ipd that contains the treatment assignment
trt_var_agd	a string, column name in tte_pseudo_ipd that contains the treatment assignment
endpoint_name	a string, name of time to event endpoint, to be show in the last line of title
time_scale	a string, time unit of median survival time, taking a value of 'years', 'months', 'weeks' or 'days'
zph_transform	a string, pass to survival::cox.zph, default is "log"
zph_log_hazard	a logical, if TRUE (default), y axis of the time dependent hazard function is log-hazard, otherwise, hazard.

Value

a 3 by 2 plot, include log-cumulative hazard plot, time dependent hazard function and unscaled Schoenfeld residual plot, before and after matching

Examples

```
# unanchored example using ph_diagplot
data(weighted_sat)
data(adtte_sat)
data(pseudo_ipd_sat)

ph_diagplot(
  weights_object = weighted_sat,
  tte_ipd = adtte_sat,
  tte_pseudo_ipd = pseudo_ipd_sat,
  trt_var_ipd = "ARM",
  trt_var_agd = "ARM",
  trt_ipd = "A",
  trt_agd = "B",
  trt_common = NULL,
  endpoint_name = "Overall Survival",
  time_scale = "week",
  zph_transform = "log",
  zph_log_hazard = TRUE
)
# anchored example using ph_diagplot
data(weighted_twt)
data(adtte_twt)
data(pseudo_ipd_twt)

ph_diagplot(
  weights_object = weighted_twt,
  tte_ipd = adtte_twt,
  tte_pseudo_ipd = pseudo_ipd_twt,
  trt_var_ipd = "ARM",
  trt_var_agd = "ARM",
  trt_ipd = "A",
  trt_agd = "B",
  trt_common = "C",
  endpoint_name = "Overall Survival",
  time_scale = "week",
  zph_transform = "log",
  zph_log_hazard = TRUE
)
```

---

ph_diagplot_lch *PH Diagnosis Plot of Log Cumulative Hazard Rate versus time or log-time*

Description

This plot is also known as log negative log survival rate.

Usage

```
ph_diagplot_lch(
  km_fit,
  time_scale,
  log_time = TRUE,
  endpoint_name = "",
  subtitle = "",
  exclude_censor = TRUE
)
```

Arguments

km_fit	returned object from survival::survfit
time_scale	a character string, 'years', 'months', 'weeks' or 'days', time unit of median survival time
log_time	logical, TRUE (default) or FALSE
endpoint_name	a character string, name of the endpoint
subtitle	a character string, subtitle of the plot
exclude_censor	logical, should censored data point be plotted

Details

a diagnosis plot for proportional hazard assumption, versus log-time (default) or time

Value

a plot of log cumulative hazard rate

Examples

```
library(survival)
data(adtte_sat)
data(pseudo_ipd_sat)
combined_data <- rbind(adtte_sat[, c("TIME", "EVENT", "ARM")], pseudo_ipd_sat)
kmobj <- survfit(Surv(TIME, EVENT) ~ ARM, combined_data, conf.type = "log-log")
ph_diagplot_lch(kmobj,
  time_scale = "month", log_time = TRUE,
  endpoint_name = "OS", subtitle = "(Before Matching)"
)
```

ph_diagplot_schoenfeld *PH Diagnosis Plot of Schoenfeld residuals for a Cox model fit*

Description

PH Diagnosis Plot of Schoenfeld residuals for a Cox model fit

Usage

```
ph_diagplot_schoenfeld(
  coxobj,
  time_scale = "months",
  log_time = TRUE,
  endpoint_name = "",
  subtitle = ""
)
```

Arguments

coxobj	object returned from coxph
time_scale	a character string, 'years', 'months', 'weeks' or 'days', time unit of median survival time
log_time	logical, TRUE (default) or FALSE
endpoint_name	a character string, name of the endpoint
subtitle	a character string, subtitle of the plot

Value

a plot of Schoenfeld residuals

Examples

```
library(survival)
data(adtte_sat)
data(pseudo_ipd_sat)
combined_data <- rbind(adtte_sat[, c("TIME", "EVENT", "ARM")], pseudo_ipd_sat)
unweighted_cox <- coxph(Surv(TIME, EVENT == 1) ~ ARM, data = combined_data)
ph_diagplot_schoenfeld(unweighted_cox,
  time_scale = "month", log_time = TRUE,
  endpoint_name = "OS", subtitle = "(Before Matching)"
)
```

`plot_weights_base` *Plot MAIC weights in a histogram with key statistics in legend*

Description

Generates a base R histogram of weights. Default is to plot either unscaled or scaled weights and not both.

Usage

```
plot_weights_base(
  weighted_data,
  bin_col,
  vline_col,
  main_title,
  scaled_weights
)
```

Arguments

`weighted_data`	object returned after calculating weights using estimate_weights
`bin_col`	a string, color for the bins of histogram
`vline_col`	a string, color for the vertical line in the histogram
`main_title`	title of the plot
`scaled_weights`	an indicator for using scaled weights instead of regular weights

Value

a plot of unscaled or scaled weights

Examples

```
plot_weights_base(weighted_sat,
  bin_col = "#6ECEB2",
  vline_col = "#688CE8",
  main_title = c("Scaled Individual Weights", "Unscaled Individual Weights"),
  scaled_weights = TRUE
)
```

`plot_weights_ggplot` *Plot MAIC weights in a histogram with key statistics in legend using* `ggplot2`

Description

Generates a `ggplot` histogram of weights. Default is to plot both unscaled and scaled weights on a same graph.

Usage

```
plot_weights_ggplot(weighted_data, bin_col, vline_col, main_title, bins)
```

Arguments

`weighted_data`	object returned after calculating weights using estimate_weights
`bin_col`	a string, color for the bins of histogram
`vline_col`	a string, color for the vertical line in the histogram
`main_title`	Name of scaled weights plot and unscaled weights plot, respectively.
`bins`	number of bin parameter to use

Value

a plot of unscaled and scaled weights

Examples

```
if (requireNamespace("ggplot2")) {
  plot_weights_ggplot(weighted_sat,
    bin_col = "#6ECEB2",
    vline_col = "#688CE8",
    main_title = c("Scaled Individual Weights", "Unscaled Individual Weights"),
    bins = 50
  )
}
```

process_agd *Pre-process aggregate data*

Description

This function checks the format of the aggregate data. Data is required to have three columns: STUDY, ARM, and N. Column names that do not have legal suffixes (MEAN, MEDIAN, SD, COUNT, or PROP) are dropped. If a variable is a count variable, it is converted to proportions by dividing the sample size (N). Note, when the count is specified, proportion is always calculated based on the count, that is, specified proportion will be ignored if applicable. If the aggregated data comes from multiple sources (i.e. different analysis population) and sample size differs for each variable, one option is to specify proportion directly instead of count by using suffix _PROP.

Usage

```
process_agd(raw_agd)
```

Arguments

raw_agd	raw aggregate data should contain STUDY, ARM, and N. Variable names should be followed by legal suffixes (i.e. MEAN, MEDIAN, SD, COUNT, or PROP).

Value

pre-processed aggregate level data

Examples

```
data(agd)
agd <- process_agd(agd)
```

pseudo_ipd_sat *Pseudo individual patient survival data from published study*

Description

Pseudo individual patient survival data from published study

Usage

```
pseudo_ipd_sat
```

Format

A data frame with 300 rows and 3 columns:

TIME Survival time in days.

EVENT Event indicator `0`/`1`.

ARM Assigned treatment arm, `"B"`.

See Also

Other unanchored datasets: `adrs_sat`, `adsl_sat`, `adtte_sat`, `agd`, `centered_ipd_sat`, `weighted_sat`

`pseudo_ipd_twt` *Pseudo individual patient survival data from published two arm study*

Description

Pseudo individual patient survival data from published two arm study

Usage

```
pseudo_ipd_twt
```

Format

A data frame with 800 rows and 3 columns:

TIME Survival time in days.

EVENT Event indicator `0`/`1`.

ARM Assigned treatment arm, `"B"`, `"C"`.

See Also

Other anchored datasets: `adrs_twt`, `adsl_twt`, `adtte_twt`, `agd`, `centered_ipd_twt`, `weighted_twt`

`set_time_conversion` *Get and Set Time Conversion Factors*

Description

Get and Set Time Conversion Factors

Usage

```
set_time_conversion(
  default = "days",
  days = 1,
  weeks = 7,
  months = 365.25/12,
  years = 365.25
)

get_time_conversion(factor = c("days", "weeks", "months", "years"))
```

Arguments

`default`	The default time scale, commonly whichever has factor = 1
`days`	Factor to divide data time units to get time in days
`weeks`	Factor to divide data time units to get time in weeks
`months`	Factor to divide data time units to get time in months
`years`	Factor to divide data time units to get time in years
`factor`	Time factor to get.

Value

No value returned. Conversion factors are stored internally and used within functions.

Examples

```
# The default time scale is days:
set_time_conversion(default = "days", days = 1, weeks = 7, months = 365.25 / 12, years = 365.25)

# Set the default time scale to years
set_time_conversion(
  default = "years",
  days = 1 / 365.25,
  weeks = 1 / 52.17857,
  months = 1 / 12,
  years = 1
)

# Get time scale factors:
```

```
get_time_conversion("years")
get_time_conversion("weeks")
```

`survfit_makeup` *Helper function to select set of variables used for Kaplan-Meier plot*

Description

Helper function to select set of variables used for Kaplan-Meier plot

Usage

```
survfit_makeup(km_fit, single_trt_name = "treatment")
```

Arguments

`km_fit`	returned object from `survival::survfit`
`single_trt_name`	name of treatment if no strata are specified in `km_fit`

Value

a list of data frames of variables from `survival::survfit()`. Data frame is divided by treatment.

Examples

```
library(survival)
data(adtte_sat)
data(pseudo_ipd_sat)
combined_data <- rbind(adtte_sat[, c("TIME", "EVENT", "ARM")], pseudo_ipd_sat)
kmobj <- survfit(Surv(TIME, EVENT) ~ ARM, combined_data, conf.type = "log-log")
survfit_makeup(kmobj)
```

`weighted_sat` *Weighted object for single arm trial data*

Description

Weighted object for single arm trial data

Usage

```
weighted_sat
```

Format

A `maicplus_estimate_weights` object created by `estimate_weights()` containing

data patient level data with weights

centered_colnames Columns used in MAIC

nr_missing Number of observations with missing data

ess Expected sample size

opt Information from `optim` from weight calculation

boot Parameters and bootstrap sample weights, `NULL` in this object

See Also

Other unanchored datasets: `adrs_sat`, `adsl_sat`, `adtte_sat`, `agd`, `centered_ipd_sat`, `pseudo_ipd_sat`

---

weighted_twt *Weighted object for two arm trial data*

Description

The weighted patient data for a two arm trial generated from the centered patient data (centered_ipd_twt). It has weights calculated for 100 bootstrap samples.

The object is generated using the following code:

```
estimate_weights(
  data = centered_ipd_twt,
  centered_colnames = c(
    "AGE_CENTERED",
    "AGE_MEDIAN_CENTERED",
    "AGE_SQUARED_CENTERED",
    "SEX_MALE_CENTERED",
    "ECOG0_CENTERED",
    "SMOKE_CENTERED"
    ),
  n_boot_iteration = 100
 )
```

Usage

```
weighted_twt
```

Format

A `maicplus_estimate_weights` object created by `estimate_weights()` containing

data patient level data with weights

centered_colnames Columns used in MAIC

nr_missing Number of observations with missing data

ess Expected sample size

opt Information from `optim` from weight calculation

boot Parameters and bootstrap sample weights for the 100 samples

See Also

Other anchored datasets: `adrs_twt`, `adsl_twt`, `adtte_twt`, `agd`, `centered_ipd_twt`, `pseudo_ipd_twt`

Index