Package ‘robsurvey’

January 17, 2022

Type Package

Title Robust Survey Statistics Estimation

Version 0.2

Description Functions to compute robust (outlier-resistant) estimates of finite population characteristics. The package supports the computations of robust means, totals, ratios, etc. Available methods are regression M- and GM-estimators, trimming, and winsorization. The package robsurvey complements the survey.

License GPL (>= 2)

Classification/MSC-2010 62D05, 62F35

URL https://github.com/tobiasschoch/robsurvey

BugReports https://github.com/tobiasschoch/robsurvey/issues

Encoding UTF-8

LazyData true

Depends R (>= 3.5.0)

Imports grDevices, stats, survey (>= 3.35-1), KernSmooth

Suggests MASS, knitr, rmarkdown, wbacon

VignetteBuilder knitr, rmarkdown

NeedsCompilation yes

Author Beat Hulliger [aut],
Tobias Schoch [aut, cre],
Martin Sterchi [ctr, com]

Maintainer Tobias Schoch <tobias.schoch@fhnw.ch>

Repository CRAN

Date/Publication 2022-01-17 17:12:45 UTC

R topics documented:

`class_svyreg_rob` *Utility Functions for Objects of Class svyreg_rob*

Description

Methods and utility functions for objects of class `svyreg_rob`.

Usage

```
## S3 method for class 'svyreg_rob'
print(x, digits = max(3L, getOption("digits") - 3L), ...)

## S3 method for class 'svyreg_rob'
summary(object, mode = c("design", "model", "compound"),
    digits = max(3L, getOption("digits") - 3L), ...)

## S3 method for class 'svyreg_rob'
coef(object, ...)

## S3 method for class 'svyreg_rob'
vcov(object, mode = c("design", "model", "compound"), ...)

## S3 method for class 'svyreg_rob'
residuals(object, ...)

## S3 method for class 'svyreg_rob'
fitted(object, ...)

## S3 method for class 'svyreg_rob'
robweights(object)

## S3 method for class 'svyreg_rob'
plot(x, which = 1:5, ...)
```

Arguments

x	object of class svyreg_rob.
digits	[integer] minimal number of significant digits.
...	additional arguments passed to the method.
object	object of class svyreg_rob.
mode	mode of variance estimator: "design", "model" or "compound" (default: "design").
which	indicating which plots to be drawn; if a subset of the plots is required, you can specify a subset of the numbers 1:5.

Details

Variance For variance estimation (summary and vcov), three modes are available:

- "design": design-based variance estimator using linearization; see Binder (1983)
- "model": model-based weighted variance estimator (the sampling design is ignored)
- "compound": design-model-based variance estimator; see Godambe and Thompson (2009)

Utility functions

- summary gives a summary of the estimation properties
- robweights extracts the robustness weights (if available)
- coef extracts the estimated regression coefficients

- vcov extracts the (estimated) covariance matrix
- residuals extracts the residuals
- fitted extracts the fitted values

References

Binder, D. A. (1983). On the Variances of Asymptotically Normal Estimators from Complex Surveys. *International Statistical Review*, **51**, 279–292.

Godambe, V.P. and Thompson, M.E. (2009). Estimating Functions and Survey Sampling, in: D. Pfeffermann and C.R. Rao (eds.), *Handbook of Statistics*, **vol. 29B**, Sample Surveys: Inference and Analysis, Chapter 26, 83–101, Amsterdam: Elsevier.

class_svystat_rob *Utility Functions for Objects of Class svystat_rob*

Description

Methods and utility functions for objects of class svystat_rob (e.g., weighted_mean_huber).

Usage

```
## S3 method for class 'svystat_rob'
summary(object, digits = max(3L,
    getOption("digits") - 3L), ...)
## S3 method for class 'svystat_rob'
coef(object, ...)
## S3 method for class 'svystat_rob'
SE(object, ...)
## S3 method for class 'svystat_rob'
vcov(object, ...)
## S3 method for class 'svystat_rob'
scale(x, ...)
## S3 method for class 'svystat_rob'
residuals(object, ...)
## S3 method for class 'svystat_rob'
fitted(object, ...)
robweights(object)
## S3 method for class 'svystat_rob'
robweights(object)
## S3 method for class 'svystat_rob'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
```

Arguments

object	object of class svystat_rob.
digits	[integer] minimal number of significant digits.
...	additional arguments passed to the method.
x	object of class svystat_rob.

Details

Utility functions:

- `summary` gives a summary of the estimation properties
- `robweights` extracts the robustness weights
- `coef` extracts the estimate of location
- `SE` extracts the (estimated) standard error
- `vcov` extracts the (estimated) covariance matrix
- `residuals` extracts the residuals
- `fitted` extracts the fitted values

`counties` *Data on a Simple Random Sample of 100 Counties in the U.S.*

Description

Data from a simple random sample (without replacement) of 100 of the 3141 counties in the United Stated (U.S. Bureau of the Census, 1994).

Usage

```
data(counties)
```

Format

A `data.frame` with 100 observations on the following variables:

`state` state, `[character]`.

`county` county, `[character]`.

`landarea` land area, 1990 (square miles), `[double]`.

`totpop` population total, 1992, `[double]`.

`unemp` number of unemployed persons, 1991, `[double]`.

`farmpop` farm population, 1990, `[double]`.

`numfarm` number of farms, 1987, `[double]`.

`farmacre` acreage in farms, 1987, `[double]`.

`weights` sampling weight, `[double]`.

`fpc` finite population corretion, `[double]`.

Details

The data (and 10 additional variables) are published in Lohr (1999, Appendix C).

Source

Lohr, S.L. (1999). *Sampling: Design and Analysis*, Pacific Grove (CA): Duxbury Press.

Examples

```
data(counties)

## Not run:
# survey design for counties data (pkg survey is required)
library(survey)
dn <- svydesign(ids = ~1, fpc = ~fpc, weights = ~weights, data = counties)

## End(Not run)
```

flour *Measurement of Copper Content in Wholemeal Flour*

Description

Measurement of copper content in wholemeal flour (measured in parts per million).

Usage

```
data(flour)
```

Format

A `data.frame` with 24 observations (sorted in ascending order) on the following variables:

`copper` copper content `[double]`.

`weight` weight `[double]`.

Details

The data are published in Maronna et al. (2006, p. 2).

Source

Maronna, R.A., D. Martin, and V.J. Yohai (2006). Robust Statistics: Theory and Methods, John Wiley and Sons: Chichester.

Examples

```
data(flour)
```

huber2 *Weighted Huber Proposal 2 Estimator*

Description

Weighted Huber Proposal 2 estimator of location and scatter.

Usage

```
huber2(x, w, k = 1.5, na.rm = FALSE, maxit = 50, tol = 1e-04, info = FALSE,
    k_Inf = 1e5, df_cor = TRUE)
```

Arguments

x	[numeric vector] data.
w	[numeric vector] weights (same length as vector x).
k	[double] robustness tuning constant ($0 < k \leq \infty$).
na.rm	[logical] indicating whether NA values should be removed before the computation proceeds (default: FALSE).
maxit	[integer] maximum number of iterations to use (default: 50).
tol	[double] numerical tolerance criterion to stop the iterations (default: 1e-04).
info	[logical] indicating whether additional information should be returned (default: FALSE).
k_Inf	[integer] numerical value that represents Inf (default: 1e+05).
df_cor	[logical] toggle for the adjustment of the degrees of freedom for the estimate of scale (default: TRUE).

Details

The function huber2 computes the weighted Huber (1964) Proposal 2 estimates of location and scale.

The method is initialized by the weighted median (location) and the weighted interquartile range (scale).

Value

The return value depends on info:

info = FALSE**:** estimate of mean or total [double]

info = TRUE**:** a [list] with items:

- characteristic [character],
- estimator [character],
- estimate [double],
- variance (default: NA),

- robust [list],
- residuals [numeric vector],
- model [list],
- design (default: NA),
- [call]

Comparison

The huber2 estimator is initialized by the weighted median and the weighted (scaled) interquartile range. For unweighted data, this estimator *differs* from hubers in **MASS**, which is initialized by mad.

The difference between the estimators is usually negligible (for sufficiently small values of tol). See examples.

References

Huber, P. J. (1964). Robust Estimation of a Location Parameter. *Annals of Mathematical Statistics* 35, pp. 73–101.

Examples

```
data(workplace)

# Weighted "Proposal 2" estimator of the mean
huber2(workplace$employment, workplace$weight, k = 8)

# More information on the estimate
m <- huber2(workplace$employment, workplace$weight, k = 8, info = TRUE)

# Estimate of scale
m$scale

# Comparison with MASS::hubers (without weights). We make a copy of MASS::hubers
library(MASS)
hubers_mod <- hubers

# Then we replace mad by the (scaled) IQR as initial scale estimator
body(hubers_mod)[[7]][[3]][[2]] <- substitute(s0 <- IQR(y, type = 2) * 0.7413)

# Define the numerical tolerance
TOLERANCE <- 1e-8

# Comparison
m1 <- huber2(workplace$payroll, rep(1, 142), tol = TOLERANCE)
m2 <- hubers_mod(workplace$payroll, tol = TOLERANCE)$mu
m1 / m2 - 1

# The absolute relative difference is < 4.0-09 (smaller than TOLERANCE)
```

`losdata` *Length-of-Stay (LOS) Hospital Data*

Description

A simple random sample of 70 patients in inpatient hospital treatment.

Usage

```
data(workplace)
```

Format

A `data.frame` with data on 70 patients.

`los` length of stay (days) `[integer]`.

`weight` sampling weight `[double]`.

`fpc` finite population correction `[double]`.

Details

The `losdata` are a simple random sample without replacement (SRSWOR) of size $n = 70$ patients from the (fictive) population of $N = 2479$ patients in inpatient hospital treatment. We have constructed the `losdata` as a showcase; though, the LOS measurements are real data that we have taken from the 201 observations in Ruffieux et al. (2000). The original LOS data of Ruffieux et al. (2000) are available in the R package **robustbase**; see `robustbase::data(los)`. Our `losdata` are a SRSWOR of size $n = 70$ from the 201 original observations.

Source

Ruffieux et al. (2000) and data.frame `los` in the R package **robustbase**.

References

Ruffieux, C., Paccaud, F. and Marazzi, A. (2000). Comparing rules for truncating hospital length of stay. *Casemix Quarterly*, **2**.

Examples

```
data(losdata)
```

mer *Minimum Estimated Risk (MER) M-Estimator*

Description

mer is an adaptive M-estimator of the weighted mean or total. It is defined as the estimator that minimizes the estimated mean square error of the estimator under consideration.

Usage

```
mer(object, verbose = TRUE, max_k = 1000, optim_args = list())
```

Arguments

object — an M-estimate of the total or mean (object of class svystat_rob).

verbose — [logical] indicating whether additional information is printed to the console (default: TRUE).

max_k — [numeric vector] defines the right boundary of the search interval (default: max_k = 1000)

optim_args — [list]: arguments passed on to optim.

Details

MER-estimators are available for the methods svymean_huber, svytotal_huber, svymean_tukey and svytotal_tukey.

Value

The tuning constant that minimizes the estimated mean square error of the estimator

References

Hulliger, B. (1995). Outlier Robust Horvitz-Thompson Estimators. *Survey Methodology*, **21**, 79–87.

Examples

```
library(survey)
data(losdata)
dn <- svydesign(ids = ~1, fpc = ~fpc, weights = ~weight, data = losdata)

# M-estimator of the total with tuning constant k = 8
m <- svymean_huber(~los, dn, type = "rhj", k = 8)

# mer-estimator
mer(m)
```

`MU284strat` *Stratified Sample from the MU284 Population*

Description

Stratified simple random sample (without replacement) of municipalities from the MU284 population of Särndal et al. (1992); stratification is by geographic region and a take-all stratum (by 1975 population size), which includes the big cities Stockholm, Göteborg, and Malmö.

Usage

```
data(MU284strat)
```

Format

A `data.frame` with 60 observations on the following variables:

`LABEL` identifier variable, `[integer]`.

`P85` 1985 population size (in thousands), `[double]`.

`P75` 1975 population size (in thousands), `[double]`.

`RMT85` Revenues from the 1985 municipal taxation (in millions of kronor), `[double]`.

`CS82` number of Conservative seats in municipal council, `[double]`.

`SS82` number of Social-Democrat seats in municipal council (1982), `[double]`.

`S82` total number of seats in municipal council (1982), `[double]`.

`ME84` number of municipal employees in 1984, `[double]`.

`REV84` real estate values according to 1984 assessment (in millions of kronor), `[double]`.

`CL` cluster indicator (a cluster consists of a set of neighbouring municipalities), `[integer]`.

`REG` geographic region indicator, `[integer]`.

`Stratum` stratum indicator, `[integer]`.

`weights` sampling weights, `[double]`.

`fpc` finite population correction, `[double]`.

Details

The MU284 population of Särndal et al. (1992, Appendix B) is a dataset with observations on the 284 municipalities in Sweden in the late 1970s and early 1980s. The `MU284` *population* data are available in the **sampling** package of Tillé and Matei (2021).

The population is divided into two parts based on 1975 population size (`P75`):

- the MU281 population, which consists of the 281 smallest municipalities;
- the MU3 population of the three biggest municipalities/ cities in Sweden (Stockholm, Göteborg, and Malmö).

The three biggest cities take exceedingly large values (representative outliers) on almost all of the variables. To account for this, a stratified sample has been drawn from the MU284 population using a take-all stratum. The sample data, `MU284strat`, (of size $n = 60$) consists of

- a stratified simple random sample (without replacement) from the MU281 population, where stratification is by geographic region (`REG`) with proportional sample size allocation;
- a take-all stratum that includes the three biggest cities/ municipalities (population M3).

Source

Särndal, C.E., Swensson, B. and Wretman, J. (1992). *Model Assisted Survey Sampling*, New York: Springer-Verlag.

Tillé, Y. and Matei, A. (2021). *sampling: Survey Sampling*. R package version 2.9. https://CRAN.R-project.org/package=sampling

Examples

```
data(MU284strat)

## Not run:
# survey design for counties data (pkg survey is required)
library(survey)
dn <- svydesign(ids = ~LABEL, strata = ~Stratum, fpc = ~fpc,
    weights = ~weights, data = MU284strat)

## End(Not run)
```

robsurvey *Package Overview*

Description

A key *design pattern* of the package is that the majority of the estimating methods is available in two "flavors":

- bare-bone methods
- survey methods

Bare-bone methods are stripped-down versions of the survey methods in terms of functionality and informativeness. These functions may serve users and other package developers as building blocks. In particular, bare-bone functions *cannot compute* variances.

The survey methods are much more capable and depend, for variance estimation, on the **survey** package.

Basic Robust Estimators

Trimming:

- Bare-bone methods: weighted_mean_trimmed and weighted_total_trimmed
- Survey methods: svymean_trimmed and svytotal_trimmed

Winsorization:

- Bare-bone methods:
  - weighted_mean_winsorized and weighted_total_winsorized
  - weighted_mean_k_winsorized and weighted_total_k_winsorized
- Survey methods:
  - svymean_winsorized and svytotal_winsorized
  - svymean_k_winsorized and svytotal_k_winsorized

Dalen's estimators (weight reduction methods):

- Bare-bone methods: weighted_mean_dalen and weighted_total_dalen
- Survey methods: svymean_dalen and svytotal_dalen

M-estimators:

- Bare-bone methods:
  - weighted_mean_huber and weighted_total_huber
  - weighted_mean_tukey and weighted_total_tukey
  - huber2 (weighted Huber proposal 2 estimator)
- Survey methods:
  - svymean_huber and svytotal_huber
  - svymean_tukey and svytotal_tukey
  - mer (minimum estimated risk estimator)

Survey Regression (weighted)

svyreg

Robust Survey Regression (weighted)

- Regression M-estimators: svyreg_huber and svyreg_tukey
- Regression GM-estimators (Mallows and Schweppe): svyreg_huberGM and svyreg_tukeyGM

Utility functions

- weighted_quantile and weighted_median
- weighted_mad and weighted_IQR
- weighted_mean and weighted_total
- weighted_line, weighted_median_line, and weighted_median_ratio

robsvyreg *Internal Function for the Regression GM-Estimator*

Description

Internal function for the robust survey regression GM-estimator; this function is **only** intended for internal use. The function does **not** check or validate the arguments. In particular, missing values in the data may make the function crash.

Usage

```
robsvyreg(x, y, w, k, psi, type, xwgt, var = NULL, verbose = TRUE, ...)
svyreg_control(tol = 1e-5, maxit = 100, k_Inf = 1e5, init = NULL,
    mad_center = TRUE, ...)
```

Arguments

x [numeric matrix] design matrix (NA values not allowed).

y [numeric vector] dependent variable (NA values not allowed).

w [numeric vector] weights (no NA's allowed).

k [double] robustness tuning constant ($0 < k \leq \infty$).

psi [integer] psi-functions: 0: Huber, 1: asymmetric Huber, and 2: Tukey biweight.

type [integer] type of estimator; 0: M-estimator; 1: Mallows and 2: Schweppe type GM-estimator.

xwgt [numeric vector] weights for design space used in GM-estimators (default: NULL, NA values not allowed).

var [numeric vector] heteroscedastic variance (default: NULL).

verbose [logical] indicating whether additional information is printed to the console (default: TRUE).

... additional arguments passed to the method (see svyreg_control).

tol [double] numerical tolerance criterion to stop the iterations (default: 1e-05).

maxit [integer] maximum number of iterations to use (default: 100).

k_Inf [integer] numerical value that represents Inf (default: 1e+05).

init either NULL or [numeric vector], if init = NULL the regression estimator is initialized by weighted least squares; otherwise, init can be specified as the estimate (i.e., p-vector) to initialize the iteratively re-weighted least squares method (default: NULL).

mad_center [logical] toggle to select whether the MAD is centered about the (weighted) median (TRUE) or about zero (default: TRUE).

Details

Not documented

Value

`[list]`

---

summary.formula *Weighted Five-Number Summary of a Variable*

Description

Weighted five-number summary used for `survey.design` and `survey.design2` objects (similar to `base::summary` for `[numeric vectors]`).

Usage

```
## S3 method for class 'formula'
summary(object, design, na.rm = FALSE, ...)
```

Arguments

`object`	one-sided `[formula]` for which a summary is desired, e.g., `~payroll`.
`design`	an object of class `survey.design` or `survey.design2`.
`na.rm`	`[logical]` indicating whether `NA` values should be removed before the computation proceeds (default: `FALSE`).
`...`	additional arguments.

Value

A weighted five-number summary (numeric variable) or a frequency table (factor variable).

Examples

```
data(workplace)

library(survey)
# Survey design for simple random sampling without replacement
dn <- svydesign(ids = ~ID, strata = ~strat, fpc = ~fpc, weights = ~weight,
    data = workplace)

summary(~payroll, dn)
```

svymean_dalen *Dalen's Estimators of the Population Mean and Total*

Description

Dalen's estimators Z2 and Z3 of the population mean and total; see weighted_mean_dalen for further details.

Usage

```
svymean_dalen(x, design, censoring, type = "Z2", na.rm = FALSE,
    verbose = FALSE)
svytotal_dalen(x, design, censoring, type = "Z2", na.rm = FALSE,
    verbose = FALSE)
```

Arguments

x	a one-sided [formula], e.g., ~myVariable.
design	an object of class survey.design; see svydesign.
censoring	[double] cutoff threshold above which the observations are censored.
type	[character] type of estimator; either "Z2" or "Z3" (default: "Z2").
na.rm	[logical] indicating whether NA values should be removed before the computation proceeds (default: FALSE).
verbose	[logical] indicating whether additional information is printed to the console (default: TRUE).

Details

Methods/ types type = "Z2" or type = "Z3"; see weighted_mean_dalen for more details.

Utility functions summary, coef, SE, vcov, residuals, fitted, and robweights.

Bare-bone functions See weighted_mean_dalen and weighted_total_dalen.

Value

Object of class svystat_rob

References

Dalén, J. (1987). Practical Estimators of a Population Total Which Reduce the Impact of Large Observations. R & D Report U/STM 1987:32, Statistics Sweden, Stockholm.

See Also

svymean_trimmed, svytotal_trimmed, svymean_winsorized, svytotal_winsorized, svymean_huber, and svytotal_huber

Examples

```
data(workplace)

library(survey)
# Survey design for simple random sampling without replacement
dn <- svydesign(ids = ~ID, strata = ~strat, fpc = ~fpc, weights = ~weight,
    data = workplace)

# Dalen's estimator Z3 of the population total
svytotal_dalen(~employment, dn, censoring = 20000, type = "Z3")

# Dalen's estimator Z3 of the population mean
m <- svymean_dalen(~employment, dn, censoring = 20000, type = "Z3")

# Summarize
summary(m)

# Extract estimate
coef(m)

# Extract estimated standard error
SE(m)
```

---

svymean_huber — *Weighted Huber Mean and Total - Robust Horvitz-Thompson Estimator*

Description

Weighted Huber M-estimator of the population mean and total (robust Horvitz-Thompson estimator)

Usage

```
svymean_huber(x, design, k, type = "rhj", asym = FALSE, na.rm = FALSE,
    verbose = TRUE, ...)
svytotal_huber(x, design, k, type = "rhj", asym = FALSE, na.rm = FALSE,
    verbose = TRUE, ...)
```

Arguments

`x`	a one-sided `[formula]`, e.g., `~myVariable`.
`design`	an object of class `survey.design`; see `svydesign`.
`k`	`[double]` robustness tuning constant ($0 < k \leq \infty$).
`type`	`[character]` type of method: `"rhj"` or `"rht"`.
`asym`	`[logical]` toggle for asymmetric Huber psi-function (default: `FALSE`).

na.rm [logical] indicating whether NA values should be removed before the computation proceeds (default: FALSE).

verbose [logical] indicating whether additional information is printed to the console (default: TRUE).

... additional arguments passed to the method (e.g., maxit: maxit number of iterations, etc.; see svyreg_control).

Details

Methods/ types type = "rht" or type = "rhj"; see weighted_mean_huber for more details.

Variance estimation. Taylor linearization (residual variance estimator).

Utility functions summary, coef, SE, vcov, residuals, fitted, and robweights.

Bare-bone functions See weighted_mean_huber and weighted_total_huber.

Value

Object of class svystat_rob

Failure of convergence

By default, the method assumes a maximum number of maxit = 100 iterations and a numerical tolerance criterion to stop the iterations of tol = 1e-05. You can run the code with specifications other than the default values by specifying the arguments maxit and/or tol in the function call; see also svyreg_control.

References

Hulliger, B. (1995). Outlier Robust Horvitz-Thompson Estimators. *Survey Methodology*, **21**, 79–87.

See Also

svymean_tukey and svytotal_tukey

Examples

```
data(workplace)

library(survey)
# Survey design for simple random sampling without replacement
dn <- svydesign(ids = ~ID, strata = ~strat, fpc = ~fpc, weights = ~weight,
    data = workplace)

# Robust Horvitz-Thompson M-estimator of the population total
svytotal_huber(~employment, dn, k = 9, type = "rht")

# Robust weighted M-estimator of the population mean
m <- svymean_huber(~employment, dn, k = 12, type = "rhj")
```

```
# Summarize
summary(m)

# Extract estimate
coef(m)

# Extract estimate of scale
scale(m)

# Extract estimated standard error
SE(m)
```

`svymean_trimmed` *Weighted Trimmed Mean and Total*

Description

Weighted trimmed population mean and total.

Usage

```
svymean_trimmed(x, design, LB = 0.05, UB = 1 - LB, na.rm = FALSE)
svytotal_trimmed(x, design, LB = 0.05, UB = 1 - LB, na.rm = FALSE)
```

Arguments

`x`	a one-sided `[formula]`, e.g., `~myVariable`.
`design`	an object of class `survey.design`; see `svydesign`.
`LB`	`[double]` lower bound of trimming such that $0 \leq$ `LB` $<$ `UB` ≤ 1.
`UB`	`[double]` upper bound of trimming such that $0 \leq$ `LB` $<$ `UB` ≤ 1.
`na.rm`	`[logical]` indicating whether `NA` values should be removed before the computation proceeds (default: `FALSE`).

Details

Characteristic. Population mean or total. Let μ denote the estimated trimmed population mean; then, the estimated trimmed total is given by $\hat{N}\mu$ with $\hat{N} = \sum w_i$, where summation is over all observations in the sample.

Trimming. The methods trims the `LB` $\cdot 100\%$ percentage of the smallest observations and the (1 - `UB`) $\cdot 100\%$ percentage of the largest observations from the data.

Variance estimation. Large-sample approximation based on the influence function; see Huber (1981, Chap. 3.3) and Shao (1994).

Utility functions. `summary`, `coef`, `SE`, `vcov`, `residuals`, `fitted`, and `robweights`.

Bare-bone functions. See `weighted_mean_trimmed` and `weighted_total_trimmed`.

Value

Object of class svystat_rob

References

Huber, P. J. (1981). *Robust Statistics*, New York: John Wiley and Sons.

Shao, J. (1994). L-Statistics in Complex Survey Problems. *The Annals of Statistics*, **22**, 976–967.

See Also

weighted_mean_trimmed and weighted_total_trimmed

Examples

```
data(workplace)

library(survey)
# Survey design for simple random sampling without replacement
dn <- svydesign(ids = ~ID, strata = ~strat, fpc = ~fpc, weights = ~weight,
    data = workplace)

# Estimated trimmed population total (5% symmetric trimming)
svytotal_trimmed(~employment, dn, LB = 0.05, UB = 0.95)

# Estimated trimmed population mean (5% trimming at the top of the distr.)
svymean_trimmed(~employment, dn, UB = 0.95)
```

---

svymean_tukey *Weighted Tukey Biweight Mean and Total - Robust Horvitz-Thompson Estimator*

Description

Weighted Tukey biweight M-estimator of the population mean and total (robust Horvitz-Thompson estimator)

Usage

```
svymean_tukey(x, design, k, type = "rhj", na.rm = FALSE, verbose = TRUE, ...)
svytotal_tukey(x, design, k, type = "rhj", na.rm = FALSE, verbose = TRUE, ...)
```

Arguments

x	a one-sided [formula], e.g., ~myVariable.
design	an object of class survey.design; see svydesign.
k	[double] robustness tuning constant ($0 < k \leq \infty$).
type	[character] type of method: "rhj" or "rht".

`na.rm`	`[logical]` indicating whether `NA` values should be removed before the computation proceeds (default: `FALSE`).
`verbose`	`[logical]` indicating whether additional information is printed to the console (default: `TRUE`).
`...`	additional arguments passed to the method (e.g., `maxit`: maxit number of iterations, etc.; see `svyreg_control`).

Details

Methods/ types `type = "rht"` or `type = "rhj"`; see `weighted_mean_tukey` for more details.

Variance estimation. Taylor linearization (residual variance estimator).

Utility functions `summary`, `coef`, `SE`, `vcov`, `residuals`, `fitted`, and `robweights`.

Bare-bone functions See `weighted_mean_tukey` and `weighted_total_tukey`.

Value

Object of class `svystat_rob`

Failure of convergence

By default, the method assumes a maximum number of `maxit = 100` iterations and a numerical tolerance criterion to stop the iterations of `tol = 1e-05`. You can run the code with specifications other than the default values by specifying the arguments `maxit` and/or `tol` in the function call; see also `svyreg_control`.

References

Hulliger, B. (1995). Outlier Robust Horvitz-Thompson Estimators. *Survey Methodology*, **21**, 79–87.

See Also

`svymean_huber` and `svytotal_huber`

Examples

```
data(workplace)

library(survey)
# Survey design for simple random sampling without replacement
dn <- svydesign(ids = ~ID, strata = ~strat, fpc = ~fpc, weights = ~weight,
    data = workplace)

# Robust Horvitz-Thompson M-estimator of the population total
svytotal_tukey(~employment, dn, k = 9, type = "rht")

# Robust weighted M-estimator of the population mean
m <- svymean_tukey(~employment, dn, k = 12, type = "rhj")
```

```
# Summarize
summary(m)

# Extract estimate
coef(m)

# Extract estimate of scale
scale(m)

# Extract estimated standard error
SE(m)
```

svymean_winsorized *Weighted Winsorized Mean and Total*

Description

Weighted winsorized mean and total

Usage

```
svymean_winsorized(x, design, LB = 0.05, UB = 1 - LB, na.rm = FALSE,
    trim_var = FALSE)
svymean_k_winsorized(x, design, k, na.rm = FALSE, trim_var = FALSE)
svytotal_winsorized(x, design, LB = 0.05, UB = 1 - LB, na.rm = FALSE,
    trim_var = FALSE)
svytotal_k_winsorized(x, design, k, na.rm = FALSE, trim_var = FALSE)
```

Arguments

x	a one-sided [formula], e.g., ~myVariable.
design	an object of class survey.design; see svydesign.
LB	[double] lower bound of winsorization such that $0 \leq \mathrm{LB} < \mathrm{UB} \leq 1$.
UB	[double] upper bound of winsorization such that $0 \leq \mathrm{LB} < \mathrm{UB} \leq 1$.
na.rm	[logical] indicating whether NA values should be removed before the computation proceeds (default: FALSE).
trim_var	[logical] logical indicating whether the variance should be approximated by the variance estimator of the trimmed mean/ total (default: FALSE).
k	[integer] number of observations to be winsorized at the top of the distribution.

Details

Characteristic. Population mean or total. Let μ denote the estimated winsorized population mean; then, the estimated winsorized total is given by $\hat{N}\mu$ with $\hat{N} = \sum w_i$, where summation is over all observations in the sample.

Modes of winsorization. The amount of winsorization can be specified in relative or absolute terms:

- *relative:* By specifying `LB` and `UB`, the method winsorizes the `LB` $\cdot 100\%$ percentage of the smallest observations and the (1 - `UB`) $\cdot 100\%$ percentage of the largest observations from the data.
- *absolute:* By specifying argument `k` in the functions with the "infix" `_k_` in their name (e.g., `svymean_k_winsorized`), the largest k observations are winsorized, $0 < k < n$, where n denotes the sample size.

Variance estimation. Large-sample approximation based on the influence function; see Huber (1981, Chap. 3.3) and Shao (1994). Two estimators are available:

`simple_var = FALSE`**:** Variance estimator of the winsorized mean/ total. The estimator depends on the estimated probability density function evaluated at the winsorization thresholds, which can be – depending on the context – numerically unstable. As a remedy, a simplified variance estimator is available by setting `simple_var = TRUE`.

`simple_var = TRUE`**:** Variance is approximated using the variance estimator of the trimmed mean/ total.

Utility functions. `summary`, `coef`, `SE`, `vcov`, `residuals`, `fitted`, and `robweights`.

Bare-bone functions. See:

- `weighted_mean_winsorized`,
- `weighted_mean_k_winsorized`,
- `weighted_total_winsorized`,
- `weighted_total_k_winsorized`.

Value

Object of class `svystat_rob`

References

Huber, P. J. (1981). *Robust Statistics*, New York: John Wiley and Sons.

Shao, J. (1994). L-Statistics in Complex Survey Problems. *The Annals of Statistics*, **22**, 976–967.

See Also

`weighted_mean_winsorized`, `weighted_mean_k_winsorized`, `weighted_total_winsorized`, and `weighted_total_k_winsorized`

Examples

```
data(workplace)

library(survey)
```

```
# Survey design for simple random sampling without replacement
dn <- svydesign(ids = ~ID, strata = ~strat, fpc = ~fpc, weights = ~weight,
    data = workplace)

# Estimated winsorized population mean (5% symmetric winsorization)
svymean_winsorized(~employment, dn, LB = 0.05)

# Estimated one-sided k winsorized population total (2 observations are
# winsorized at the top of the distribution)
svytotal_k_winsorized(~employment, dn, k = 2)
```

svyreg *Survey Regression Estimator*

Description

svyreg is used to fit survey weighted linear models.

Usage

```
svyreg(formula, design, var = NULL, na.rm = FALSE)
```

Arguments

formula	a [formula] object (i.e., symbolic description of the model)
design	an object of class survey.design; see svydesign.
var	[numeric vector] heteroscedastic variance (default: NULL, i.e., homoscedastic variance).
na.rm	[logical] indicating whether NA values should be removed before the computation proceeds (default: FALSE).

Details

svyreg computes the regression coefficients by weighted least squares.

Models for svyreg_rob are specified symbolically. A typical model has the form response ~ terms where response is the (numeric) response vector and terms is a series of terms which specifies a linear predictor for response; see formula and lm.

A formula has an implied intercept term. To remove this use either y ~ x -1 or y ~ 0 + x; see formula for more details of allowed formulae.

Value

Object of class svyreg_rob.

See Also

summary for summaries.

The generic functions coef, residuals, fitted, and vcov.

plot for regression diagnostic plot methods.

Robust estimating methods svyreg_huber, svyreg_huberGM, svyreg_tukey, and svyreg_tukeyGM.

Examples

```
data(workplace)

library(survey)
# Survey design for simple random sampling without replacement
dn <- svydesign(ids = ~ID, strata = ~strat, fpc = ~fpc, weights = ~weight,
    data = workplace)

# Compute the regression estimate
m <- svyreg(payroll ~ employment, dn)

# Regression inference
summary(m)

# Extract the coefficients
coef(m)

# Extract variance/ covariance matrix
vcov(m)
```

---

svyreg_huber *Huber Robust Survey Regression M- and GM-Estimator*

Description

svyreg_huber and svyreg_huberGM compute, respectively, a survey weighted M- and GM-estimator of regression using the Huber psi-function.

Usage

```
svyreg_huber(formula, design, k, var = NULL, na.rm = FALSE, asym = FALSE,
    verbose = TRUE, ...)
svyreg_huberGM(formula, design, k, type = c("Mallows", "Schweppe"),
    xwgt, var = NULL, na.rm = FALSE, asym = FALSE, verbose = TRUE, ...)
```

Arguments

formula	a [formula] object (i.e., symbolic description of the model)
design	an object of class survey.design; see svydesign.
k	[double] robustness tuning constant ($0 < k \leq \infty$).

`var` `[numeric vector]` heteroscedastic variance (default: `NULL`, i.e., homoscedastic variance).

`na.rm` `[logical]` indicating whether `NA` values should be removed before the computation proceeds (default: `FALSE`).

`asym` `[logical]` toggle for asymmetric Huber psi-function (default: `FALSE`).

`verbose` `[logical]` indicating whether additional information is printed to the console (default: `TRUE`).

`...` additional arguments passed to the method (e.g., `maxit`: maxit number of iterations, etc.).

`type` `[character]` `"Mallows"` or `"Schweppe"`.

`xwgt` `[numerical vector]` of weights in the design space (default: `NULL`); `xwgt` is only relevant for `type = "Mallows"` or `type = "Schweppe"`.

Details

`svyreg_huber` and `svyreg_huberGM` compute, respectively, M- and GM-estimates of regression by iteratively re-weighted least squares (IRWLS). The estimate of regression scale is (by default) computed by the (normalized) weighted median of absolute deviations from the weighted median (MAD; see `weighted_mad`) for each IRWLS iteration.

M-estimator The regression M-estimator is robust against residual outliers (granted that the tuning constant `k` is chosen appropriately).

GM-estimator `svyreg_huberGM` implements the Mallows and Schweppe regression GM-estimator (see argument `type`). The regression GM-estimator are robust against residual outliers *and* outliers in the model's design space (leverage observations; see argument `xwgt`).

Numerical optimization See `svyreg_control`.

Models Models for `svyreg_rob` are specified symbolically. A typical model has the form `response ~ terms` where `response` is the (numeric) response vector and `terms` is a series of terms which specifies a linear predictor for `response`; see `formula` and `lm`.

A formula has an implied intercept term. To remove this use either `y ~ x -1` or `y ~ 0 + x`; see `formula` for more details of allowed formulae.

Value

Object of class `svyreg.rob`

Failure of convergence

By default, the method assumes a maximum number of `maxit = 100` iterations and a numerical tolerance criterion to stop the iterations of `tol = 1e-05`. You can run the code with specifications other than the default values by specifying the arguments `maxit` and/or `tol` in the function call; see also `svyreg_control`.

See Also

summary for summaries.

The generic functions coef, residuals, fitted, and vcov.

plot for regression diagnostic plot methods.

Other robust estimating methods svyreg_tukey and svyreg_tukeyGM.

Examples

```
data(workplace)

library(survey)
# Survey design for simple random sampling without replacement
dn <- svydesign(ids = ~ID, strata = ~strat, fpc = ~fpc, weights = ~weight,
    data = workplace)

# Compute the regression estimate
m <- svyreg_huber(payroll ~ employment, dn, k = 8)

# Regression inference
summary(m)

# Extract the coefficients
coef(m)

# Extract variance/ covariance matrix
vcov(m)
```

---

svyreg_tukey *Tukey Biweight Robust Survey Regression M- and GM-Estimator*

Description

svyreg_tukey and svyreg_tukeyGM compute, respectively, a survey weighted M- and GM-estimator of regression using the biweight Tukey psi-function.

Usage

```
svyreg_tukey(formula, design, k, var = NULL, na.rm = FALSE, verbose = TRUE,
    ...)
svyreg_tukeyGM(formula, design, k, type = c("Mallows", "Schweppe"),
    xwgt, var = NULL, na.rm = FALSE, verbose = TRUE, ...)
```

Arguments

formula a [formula] object (i.e., symbolic description of the model)

design an object of class survey.design; see svydesign.

k [double] robustness tuning constant ($0 < k \leq \infty$).

`var` `[numeric vector]` heteroscedastic variance (default: `NULL`, i.e., homoscedastic variance).

`na.rm` `[logical]` indicating whether `NA` values should be removed before the computation proceeds (default: `FALSE`).

`verbose` `[logical]` indicating whether additional information is printed to the console (default: `TRUE`).

`...` additional arguments passed to the method (e.g., `maxit`: maxit number of iterations, etc.).

`type` `[character]` `"Mallows"` or `"Schweppe"`.

`xwgt` `[numerical vector]` of weights in the design space (default: `NULL`); `xwgt` is only relevant for `type = "Mallows"` or `type = "Schweppe"`.

Details

`svyreg_tukey` and `svyreg_tukeyGM` compute, respectively, M- and GM-estimates of regression by iteratively re-weighted least squares (IRWLS). The estimate of regression scale is (by default) computed by the (normalized) weighted median of absolute deviations from the weighted median (MAD; see `weighted_mad`) for each IRWLS iteration.

M-estimator The regression M-estimator is robust against residual outliers (granted that the tuning constant `k` is chosen appropriately).

GM-estimator `svyreg_huberGM` implements the Mallows and Schweppe regression GM-estimator (see argument `type`). The regression GM-estimator are robust against residual outliers *and* outliers in the model's design space (leverage observations; see argument `xwgt`).

Numerical optimization See `svyreg_control`.

Models Models for `svyreg_rob` are specified symbolically. A typical model has the form `response ~ terms` where `response` is the (numeric) response vector and `terms` is a series of terms which specifies a linear predictor for `response`; see `formula` and `lm`.

A formula has an implied intercept term. To remove this use either `y ~ x -1` or `y ~ 0 + x`; see `formula` for more details of allowed formulae.

Value

Object of class `svyreg.rob`

Failure of convergence

By default, the method assumes a maximum number of `maxit = 100` iterations and a numerical tolerance criterion to stop the iterations of `tol = 1e-05`. You can run the code with specifications other than the default values by specifying the arguments `maxit` and/or `tol` in the function call; see also `svyreg_control`.

See Also

`summary` for summaries.

The generic functions `coef`, `residuals`, `fitted`, and `vcov`.

`plot` for regression diagnostic plot methods.

Other robust estimating methods `svyreg_huber` and `svyreg_huberGM`.

Examples

```
data(workplace)

library(survey)
# Survey design for simple random sampling without replacement
dn <- svydesign(ids = ~ID, strata = ~strat, fpc = ~fpc, weights = ~weight,
    data = workplace)

# Compute the regression estimate
m <- svyreg_tukey(payroll ~ employment, dn, k = 8)

# Regression inference
summary(m)

# Extract the coefficients
coef(m)

# Extract variance/ covariance matrix
vcov(m)
```

weighted_IQR *Weighted Interquartile Range (IQR)*

Description

weighted_IQR computes weighted (normalized) interquartile range

Usage

```
weighted_IQR(x, w, na.rm = FALSE, constant = 0.7413)
```

Arguments

x	[numeric vector] data.
w	[numeric vector] weights (same length as vector x).
na.rm	[logical] indicating whether NA values should be removed before the computation proceeds (default: FALSE).
constant	[double] constant scaling factor to make the weighted IQR a consistent estimator of the scale (default: 0.7413).

Details

By default, the weighted IQR is normalized to be an unbiased estimate of scale at the Gaussian core model. If normalization is not wanted, put constant = 1.

Value

Weighted IQR

Examples

```
data(workplace)

# normalized weighted IQR (default)
weighted_IQR(workplace$employment, workplace$weight)

# weighted IQR (without normalization)
weighted_IQR(workplace$employment, workplace$weight, constant = 1)
```

---

weighted_line *Weighted Robust Line Fitting*

Description

`weighted_line` fits a robust line and allows weights.

Usage

```
weighted_line(x, y = NULL, w, na.rm = FALSE, iter = 1)
```

Arguments

x	`[numeric vector]` explanatory variable.
y	`[numeric vector]` response variable (default: `NULL`).
w	`[numeric vector]` weights (same length as vector `x`).
na.rm	`[logical]` indicating whether `NA` values should be removed before the computation proceeds (default: `FALSE`).
iter	`[integer]` number of iterations to use (default: `1`).

Details

`weighted_line` uses different quantiles for splitting the sample than `stats::line()`.

Value

intercept and slope of the fitted line

See Also

`line`

Examples

```
data(cars)

# compute weighted line
weighted_line(cars$speed, cars$dist, w = rep(1, length(cars$speed)))
weighted_line(cars$speed, cars$dist, w = rep(1:10, each = 5))
```

weighted_mad *Weighted Median Absolute Deviation from the Median (MAD)*

Description

weighted_mad computes weighted median of the absolute deviations from the weighted median

Usage

```
weighted_mad(x, w, na.rm = FALSE, constant = 1.482602)
```

Arguments

x	[numeric vector] data.
w	[numeric vector] weights (same length as vector x).
na.rm	[logical] indicating whether NA values should be removed before the computation proceeds (default: FALSE).
constant	[double] constant scaling factor to make the MAD a consistent estimator of the scale (default: 1.4826).

Details

The weighted MAD is computed as the (normalized) weighted median of the absolute deviation from the weighted median; see weighted_median. The weighted MAD is normalized to be an unbiased estimate of scale at the Gaussian core model. If normalization is not wanted, put constant = 1.

Value

Weighted median absolute deviation from the (weighted) median

Examples

```
data(workplace)

# normalized weighted MAD (default)
weighted_mad(workplace$employment, workplace$weight)

# weighted MAD (without normalization)
weighted_mad(workplace$employment, workplace$weight, constant = 1)
```

weighted_mean *Weighted Total and Mean (Horvitz-Thompson and Hajek Estimators)*

Description

Weighted total and mean (Horvitz-Thompson and Hajek estimators)

Usage

```
weighted_mean(x, w, na.rm = FALSE)
weighted_total(x, w, na.rm = FALSE)
```

Arguments

`x`	`[numeric vector]` data.
`w`	`[numeric vector]` weights (same length as vector `x`).
`na.rm`	`[logical]` indicating whether `NA` values should be removed before the computation proceeds (default: `FALSE`).

Details

`weighted_total` and `weighted_mean` compute, respectively, the Horvitz-Thompson estimator of the total and the Hajek estimator of the mean.

Value

Estimated population mean or total

Examples

```
data(workplace)

# Horvitz-Thompson estimator of the total
weighted_total(workplace$employment, workplace$weight)

# Hajek estimator of the mean
weighted_mean(workplace$employment, workplace$weight)
```

`weighted_mean_dalen` *Dalen Estimators of the Mean and Total*

Description

Dalén's estimators of the mean and total (bare-bone functions with limited functionality)

Usage

```
weighted_mean_dalen(x, w, censoring, type = "Z2", info = FALSE,
    na.rm = FALSE, verbose = TRUE)
weighted_total_dalen( x, w, censoring, type = "Z2", info = FALSE,
    na.rm = FALSE, verbose = TRUE)
```

Arguments

`x` `[numeric vector]` data.

`w` `[numeric vector]` weights (same length as vector `x`).

`censoring` `[double]` cutoff threshold above which the observations are censored.

`type` `[character]` type of estimator; either `"Z2"` or `"Z3"` (default: `"Z2"`).

`info` `[logical]` indicating whether additional information should be returned (default: `FALSE`).

`na.rm` `[logical]` indicating whether `NA` values should be removed before the computation proceeds (default: `FALSE`).

`verbose` `[logical]` indicating whether additional information should be printed to the console (default: `FALSE`).

Details

Let $\sum_{i \in s} w_i x_i$ denote the expansion estimator of the x-total (summation is over all elements i in sample s). The estimators Z2 and Z3 of Dalén (1987) are defined as follows.

Estimator Z2 The estimator Z2 of the population total sums over $\min(c, w_i x_i)$; hence, it censors the products $w_i x_i$ to the censoring constant c (`censoring`). The estimator of the population x-mean is is defined similarly.

Estimator Z3 The estimator Z3 of the population total is defined as the sum over the elements z_i, which is equal to $z_i = w_i x_i$ if$w_i y_i \leq c$ and $z_i = c + (y_i - c/w_i)$ otherwise.

Value

The return value depends on `info`:

`info = FALSE`**:** estimate of mean or total `[double]`

`info = TRUE`**:** a `[list]` with items:

- `characteristic [character]`,

- estimator [character],
- estimate [double],
- variance (default: NA),
- robust [list],
- residuals [numeric vector],
- model [list],
- design (default: NA),
- [call]

References

Dalén, J. (1987). Practical Estimators of a Population Total Which Reduce the Impact of Large Observations. R & D Report U/STM 1987:32, Statistics Sweden, Stockholm.

Examples

```
data(workplace)

# Dalen's estimator of the total (with censoring threshold: 100000)
weighted_total_dalen(workplace$employment, workplace$weight, 100000)
```

weighted_mean_huber *Weighted Huber Mean and Total (bare-bone functions)*

Description

Weighted Huber M-estimator of the mean and total (bare-bone functions with limited functionality; see svymean_huber and svytotal_huber for more capable methods)

Usage

```
weighted_mean_huber(x, w, k, type = "rhj", asym = FALSE, info = FALSE,
    na.rm = FALSE, verbose = TRUE, ...)
weighted_total_huber(x, w, k, type = "rhj", asym = FALSE, info = FALSE,
    na.rm = FALSE, verbose = TRUE, ...)
```

Arguments

x	[numeric vector] data.
w	[numeric vector] weights (same length as vector x).
k	[double] robustness tuning constant ($0 < k \leq \infty$).
type	[character] type of method: "rhj" or "rht"; see below (default: "rhj").
asym	[logical] toggle for asymmetric Huber psi-function (default: FALSE).
info	[logical] indicating whether additional information should be returned (default: FALSE).

na.rm	[logical] indicating whether NA values should be removed before the computation proceeds (default: FALSE).
verbose	[logical] indicating whether additional information is printed to the console (default: TRUE).
...	additional arguments passed to the method (e.g., maxit: maxit number of iterations, etc.).

Details

Characteristic. Population mean or total. Let μ denote the estimated population mean; then, the estimated total is given by $\hat{N}\mu$ with $\hat{N} = \sum w_i$, where summation is over all observations in the sample.

Type. Two methods/types are available for estimating the location μ (and the scale σ; see models, below):

type = "rhj" (default): robust Hajek *M*-estimator of the population mean and total, respectively. This estimator is recommended for sampling designs the inclusion probabilities of which are *not* proportional to some measure of size.

type = "rht": robust Horvitz-Thompson *M*-estimator of the population mean and total, respectively. This estimator is recommended for proportional-to-size designs.

Variance estimation. See the related but more capable functions:

- svymean_huber,
- svytotal_huber.

Psi-function. By default, the Huber psi-function is used in the specification of the M-estimator. An asymmetric version of the Huber psi-function can be used by setting asym = TRUE.

Value

The return value depends on info:

info = FALSE: estimate of mean or total [double]

info = TRUE: a [list] with items:

- characteristic [character],
- estimator [character],
- estimate [double],
- variance (default: NA),
- robust [list],
- residuals [numeric vector],
- model [list],
- design (default: NA),
- [call]

Failure of convergence

By default, the method assumes a maximum number of maxit = 100 iterations and a numerical tolerance criterion to stop the iterations of tol = 1e-05. You can run the code with specifications other than the default values by specifying the arguments maxit and/or tol in the function call; see also svyreg_control.

References

Hulliger, B. (1995). Outlier Robust Horvitz-Thompson Estimators. *Survey Methodology*, **21**, 79–87.

See Also

weighted_mean_tukey and weighted_total_tukey

Examples

```
data(workplace)

# Robust Horvitz-Thompson M-estimator of the population total
weighted_total_huber(workplace$employment, workplace$weight, k = 9,
    type = "rht")

# Robust weighted M-estimator of the population mean
weighted_mean_huber(workplace$employment, workplace$weight, k = 12,
    type = "rhj")
```

weighted_mean_trimmed *Weighted Trimmed Mean and Total (bare-bone functions)*

Description

Weighted trimmed mean and total (bare-bone functions with limited functionality; see svymean_trimmed and svytotal_trimmed for more capable methods)

Usage

```
weighted_mean_trimmed(x, w, LB = 0.05, UB = 1 - LB, info = FALSE,
    na.rm = FALSE)
weighted_total_trimmed(x, w, LB = 0.05, UB = 1 - LB, info = FALSE,
    na.rm = FALSE)
```

Arguments

x	[numeric vector] data.
w	[numeric vector] weights (same length as vector x).
LB	[double] lower bound of trimming such that $0 \leq \text{LB} < \text{UB} \leq 1$.
UB	[double] upper bound of trimming such that $0 \leq \text{LB} < \text{UB} \leq 1$.
info	[logical] indicating whether additional information should be returned (default: FALSE).
na.rm	[logical] indicating whether NA values should be removed before the computation proceeds (default: FALSE).

Details

Characteristic. Population mean or total. Let μ denote the estimated trimmed population mean; then, the estimated trimmed population total is given by $\hat{N}\mu$ with $\hat{N} = \sum w_i$, where summation is over all observations in the sample.

Trimming. The methods trims the `LB` $\cdot 100\%$ percentage of the smallest observations and the (1 - `UB`) $\cdot 100\%$ percentage of the largest observations from the data.

Variance estimation. See survey methods:

- `svymean_trimmed`,
- `svytotal_trimmed`.

Value

The return value depends on `info`:

`info = FALSE`**:** estimate of mean or total `[double]`

`info = TRUE`**:** a `[list]` with items:

- `characteristic [character]`,
- `estimator [character]`,
- `estimate [double]`,
- `variance` (default: `NA`),
- `robust [list]`,
- `residuals [numeric vector]`,
- `model [list]`,
- `design` (default: `NA`),
- `[call]`

See Also

`svymean_trimmed` and `svytotal_trimmed`

Examples

```
data(workplace)

# Estimated trimmed population total (5% symmetric trimming)
weighted_total_trimmed(workplace$employment, workplace$weight, LB = 0.05,
    UB = 0.95)

# Estimated trimmed population mean (5% trimming at the top of the distr.)
weighted_mean_trimmed(workplace$employment, workplace$weight, UB = 0.95)
```

weighted_mean_tukey *Weighted Tukey Biweight Mean and Total (bare-bone functions)*

Description

Weighted Tukey biweight M-estimator of the mean and total (bare-bone functions with limited functionality; see svymean_tukey and svytotal_tukey for more capable methods)

Usage

```
weighted_mean_tukey(x, w, k, type = "rhj", info = FALSE, na.rm = FALSE,
    verbose = TRUE, ...)
weighted_total_tukey(x, w, k, type = "rhj", info = FALSE, na.rm = FALSE,
    verbose = TRUE, ...)
```

Arguments

x [numeric vector] data.

w [numeric vector] weights (same length as vector x).

k [double] robustness tuning constant ($0 < k \leq \infty$).

type [character] type of method: "rhj" or "rht"; see below (default: "rhj").

info [logical] indicating whether additional information should be returned (default: FALSE).

na.rm [logical] indicating whether NA values should be removed before the computation proceeds (default: FALSE).

verbose [logical] indicating whether additional information is printed to the console (default: TRUE).

... additional arguments passed to the method (e.g., maxit: maxit number of iterations, etc.).

Details

Characteristic. Population mean or total. Let μ denote the estimated population mean; then, the estimated total is given by $\hat{N}\mu$ with $\hat{N} = \sum w_i$, where summation is over all observations in the sample.

Type. Two methods/types are available for estimating the location μ (and the scale σ; see models, below):

type = "rhj" (default): robust Hajek *M*-estimator of the population mean and total, respectively. This estimator is recommended for sampling designs the inclusion probabilities of which are *not* proportional to some measure of size.

type = "rht": robust Horvitz-Thompson *M*-estimator of the population mean and total, respectively. This estimator is recommended for proportional-to-size designs.

Variance estimation. See survey methods:

- svymean_tukey,
- svytotal_tukey.

Psi-function. Tukey biweight psi-function with tuning parameter k

Value

The return value depends on info:

info = FALSE: estimate of mean or total [double]

info = TRUE: a [list] with items:

- characteristic [character],
- estimator [character],
- estimate [double],
- variance (default: NA),
- robust [list],
- residuals [numeric vector],
- model [list],
- design (default: NA),
- [call]

Failure of convergence

By default, the method assumes a maximum number of maxit = 100 iterations and a numerical tolerance criterion to stop the iterations of tol = 1e-05. You can run the code with specifications other than the default values by specifying the arguments maxit and/or tol in the function call; see also svyreg_control.

References

Hulliger, B. (1995). Outlier Robust Horvitz-Thompson Estimators. *Survey Methodology*, **21**, 79–87.

See Also

weighted_mean_huber and weighted_total_huber

Examples

```
data(workplace)

# Robust Horvitz-Thompson M-estimator of the population total
weighted_total_tukey(workplace$employment, workplace$weight, k = 9,
    type = "rht")

# Robust weighted M-estimator of the population mean
weighted_mean_tukey(workplace$employment, workplace$weight, k = 12,
    type = "rhj")
```

`weighted_mean_winsorized` *Weighted Winsorized Mean and Total (bare-bone functions)*

Description

Weighted winsorized mean and total (bare-bone functions with limited functionality; see `svymean_winsorized` and `svytotal_winsorized` for more capable methods)

Usage

```
weighted_mean_winsorized(x, w, LB = 0.05, UB = 1 - LB, info = FALSE,
    na.rm = FALSE)
weighted_mean_k_winsorized(x, w, k, info = FALSE, na.rm = FALSE)
weighted_total_winsorized(x, w, LB = 0.05, UB = 1 - LB, info = FALSE,
    na.rm = FALSE)
weighted_total_k_winsorized(x, w, k, info = FALSE, na.rm = FALSE)
```

Arguments

`x` `[numeric vector]` data.

`w` `[numeric vector]` weights (same length as vector `x`).

`LB` `[double]` lower bound of winsorization such that $0 \leq$ `LB` $<$ `UB` ≤ 1.

`UB` `[double]` upper bound of winsorization such that $0 \leq$ `LB` $<$ `UB` ≤ 1.

`info` `[logical]` indicating whether additional information should be returned (default: `FALSE`).

`na.rm` `[logical]` indicating whether `NA` values should be removed before the computation proceeds (default: `FALSE`).

`k` `[integer]` number of observations to be winsorized at the top of the distribution.

Details

Characteristic. Population mean or total. Let μ denote the estimated winsorized population mean; then, the estimated population total is given by $\hat{N}\mu$ with $\hat{N} = \sum w_i$, where summation is over all observations in the sample.

Modes of winsorization. The amount of winsorization can be specified in relative or absolute terms:

- *relative:* By specifying `LB` and `UB`, the methods winsorizes the `LB` $\cdot 100\%$ percentage of the smallest observations and the (1 - `UB`) $\cdot 100\%$ percentage of the largest observations from the data.
- *absolute:* By specifying argument `k` in the functions with the "infix" `_k_` in their name, the largest k observations are winsorized, $0 < k < n$, where n denotes the sample size.

Variance estimation. See survey methods:

- svymean_winsorized,
- svytotal_winsorized,
- svymean_k_winsorized,
- svytotal_k_winsorized.

Value

The return value depends on info:

info = FALSE**:** estimate of mean or total [double]

info = TRUE**:** a [list] with items:

- characteristic [character],
- estimator [character],
- estimate [double],
- variance (default: NA),
- robust [list],
- residuals [numeric vector],
- model [list],
- design (default: NA),
- [call]

See Also

svymean_winsorized, svymean_k_winsorized, svytotal_winsorized, and svytotal_k_winsorized

Examples

```
data(workplace)

# Estimated winsorized population mean (5% symmetric winsorization)
weighted_mean_winsorized(workplace$employment, workplace$weight, LB = 0.05)

# Estimated one-sided k winsorized population total (2 observations are
# winsorized at the top of the distribution)
weighted_total_k_winsorized(workplace$employment, workplace$weight, k = 2)
```

---

weighted_median *Weighted Median*

Description

weighted_median computes the weighted population median.

Usage

```
weighted_median(x, w, na.rm = FALSE)
```

Arguments

x [numeric vector] data.

w [numeric vector] weights (same length as vector x).

na.rm [logical] indicating whether NA values should be removed before the computation proceeds (default: FALSE).

Details

Weighted sample median; see weighted_quantile for more information.

Value

Weighted estimate of the population median

See Also

weighted_quantile

Examples

```
data(workplace)

weighted_median(workplace$employment, workplace$weight)
```

weighted_median_line *Robust Simple Linear Regression Based on Medians*

Description

Robust simple linear regression based on medians: two methods are available: "slopes" and "product".

Usage

```
weighted_median_line(x, y = NULL, w, type = "slopes", na.rm = FALSE)
```

Arguments

x [numeric vector] explanatory variable.

y [numeric vector] response variable (default: NULL).

w [numeric vector] weights (same length as vector x).

type [character] "slopes" or "products" (default: "slopes").

na.rm [logical] indicating whether NA values should be removed before the computation proceeds (default: FALSE).

Details

Overview. Robust simple linear regression based on medians

Type. Two methods/ types are available. Let $m(x, w)$ denote the weighted median of variable `x` with weights `w`:

`type = "slopes"`**:** The slope is computed as

$$b1 = m\left(\frac{y - m(y, w)}{x - m(x, w)}, w\right).$$

`type = "products"`**:** The slope is computed as

$$b1 = \frac{m\big([y - m(y, w)][x - m(x, w)], w\big)}{m\big([x - m(x, w)]^2, w\big)}.$$

Value

A vector with two components: intercept and slope

See Also

`line`, `weighted_line`, `weighted_median_ratio`

Examples

```
x <- c(1, 2, 4, 5)
y <- c(3, 2, 7, 4)
weighted_line(y~x, w=rep(1, length(x)))
weighted_median_line(y~x, w = rep(1, length(x)))
weighted_median_line(y~x, w = rep(1, length(x)), type = "prod")

data(cars)
with(cars, weighted_median_line(dist ~ speed, w = rep(1, length(dist))))
with(cars, weighted_median_line(dist ~ speed, w = rep(1, length(dist)),
type = "prod"))

# weighted
w <- c(rep(1,20), rep(2,20), rep(5, 10))
with(cars, weighted_median_line(dist ~ speed, w = w))
with(cars, weighted_median_line(dist ~ speed, w = w, type = "prod"))

# outlier in y
cars$dist[49] <- 360
with(cars, weighted_median_line(dist ~ speed, w = w))
with(cars, weighted_median_line(dist ~ speed, w = w, type = "prod"))

# outlier in x
data(cars)
cars$speed[49] <- 72
with(cars, weighted_median_line(dist ~ speed, w = w))
with(cars, weighted_median_line(dist ~ speed, w = w, type = "prod"))
```

weighted_median_ratio *Weighted Robust Ratio Estimator Based on Median*

Description

A weighted median of the ratios y/x determines the slope of a regression through the origin.

Usage

```
weighted_median_ratio(x, y = NULL, w, na.rm = FALSE)
```

Arguments

x [numeric vector] explanatory variable.

y [numeric vector] response variable (default: NULL).

w [numeric vector] weights (same length as vector x).

na.rm [logical] indicating whether NA values should be removed before the computation proceeds (default: FALSE).

Value

A vector with two components: intercept and slope

See Also

line, weighted_line, and weighted_median_line

Examples

```
x <- c(1,2,4,5)
y <- c(1,0,5,2)
weighted_median_ratio(y~x, w = rep(1, length(y)))
```

weighted_quantile *Weighted Quantile*

Description

weighted_quantile computes weighted population quantiles.

Usage

```
weighted_quantile(x, w, probs, na.rm = FALSE)
```

Arguments

`x` `[numeric vector]` data.

`w` `[numeric vector]` weights (same length as vector `x`).

`probs` `[numeric vector]` vector of probabilities with values in `[0,1]`.

`na.rm` `[logical]` indicating whether `NA` values should be removed before the computation proceeds (default: `FALSE`).

Details

Overview. `weighted_quantile` computes the weighted sample quantiles; argument `probs` allows vector inputs.

Implementation. The function is based on a weighted version of the quickselect algorithm with the Bentley and McIlroy (1993) 3-way partitioning scheme. For very small arrays, we use insertion sort.

Compatibility. For equal weighting, i.e. when all elements in `w` are equal, `weighted_quantile` computes quantiles that are identical with `type = 2` in `stats::quantile`; see also Hyndman and Fan (1996).

Value

Weighted estimate of the population quantiles

References

Bentley, J. L. and McIlroy, D. M. (1993). Engineering a Sort Function, *Software - Practice and Experience*, **23**, 1249–1265.

Hyndman, R.J. and Fan, Y. (1996). Sample Quantiles in Statistical Packages, *The American Statistician*, **50**, 361–365.

See Also

`weighted_median`

Examples

```
data(workplace)

# Weighted 25% quantile (1st quartile)
weighted_quantile(workplace$employment, workplace$weight, 0.25)
```

wgt_functions — *Weight Functions (for the M- and GM-Estimators)*

Description

Weight functions associated with the Huber and the Tukey biweight psi-functions; and the weight function of Simpson et al. (1992) for GM-estimators.

Usage

```
huberWgt(x, k = 1.345)
tukeyWgt(x, k = 4.685)
simpsonWgt(x, a, b)
```

Arguments

x	`[numeric vector]` data.
k	`[double]` robustness tuning constant ($0 < k \leq \infty$).
a	`[double]` robustness tuning constant ($0 \leq a \leq \infty$); see details below.
b	`[double]` robustness tuning constant ($0 < b \leq \infty$; see details below.

Details

The functions `huberWgt` and `tukeyWgt` return the weights associated with the respective psi-function.

The function `simpsonWgt` is used (in regression GM-estimators) to downweight leverage observations (i.e., outliers in the model's design space). Let d_i denote the (robust) squared Mahalanobis distance of the i-th observation. The Simpson et al. (1992) type of weight is defined as $\min\{1, (b/d_i)^{a/2}\}$, where `a` and `b` are tuning constants.

- By default, `a = 1`; this choice implies that the weights are computed on the basis of the robust Mahalanobis distances. Alternative: `a = Inf` implies a weight of zero for all observations whose (robust) squared Mahalanobis is larger than `b`.
- The tuning constants `b` is a threshold on the distances.

Value

Numerical vector of weights

References

Simpson, D. G., Ruppert, D. and Carroll, R.J. (1992). On One-Step GM Estimates and Stability of Inferences in Linear Regression. *Journal of the American Statistical Association*, **87**, 439–450.

See Also

`svyreg_huber`, `svyreg_huberGM`, `svyreg_tukey`, and `svyreg_tukeyGM`

workplace *(Modified) Canadian Workplace and Employee Survey*

Description

The workplace data are from Fuller (2009, pp. 366–367).

Usage

```
data(workplace)
```

Format

A data.frame with a sample of 142 workplaces on the following variables

ID identifier variable [integer].

weight sampling weight [double].

employment (total) employment [double].

payroll payroll [double].

fpc finite population correction [integer].

Details

The workplace data represent a sample of workplaces in the retail sector in a Canadian province. The data are *not* those collected by Statistics Canada, but have been generated by Fuller (2009, Example 3.1.1) to display similar characteristics to the original 1999 Canadian Workplace and Employee Survey (WES).

Sampling design of the 1999 WES: The WES target population is defined as all workplaces operating in Canada with paid employees. The sampling frame is stratified by industry, geographic region, and size (size is defined using estimated employment). A sample of workplaces has been drawn independently in each stratum using simple random sample without replacement (sample size is determined by Neyman allocation). Several strata containing very large workplaces were sampled exhaustively; see Patak et al (1998). The original sampling weights were adjusted for nonresponse.

Remarks by Fuller (2009, p. 365): The original weights of WES were about 2200 for the stratum of small workplaces, about 750 for medium-sized, and about 35 for large workspaces.

Source

The data workplace is from Table 6.3 in Fuller (2009, pp. 366–367).

References

Fuller, W. A. (2009). *Sampling Statistics*, Hoboken (NJ): John Wiley & Sons.

Patak, Z., Hidiroglou, M. and Lavallée, P. (1998). The methodology of the Workplace and Employee Survey. *Proceedings of the Survey Research Methods Section, American Statistical Association*, 83–91.

Examples

```
data(workplace)
```

Index