Package ‘robsurvey’

June 11, 2019

Type Package

Title Robust Survey Statistics Estimation

Version 0.1.1

Description Multiple functions to compute robust survey statistics. The package supports the computations of robust means, totals, and ratios. Available methods are Huber M-estimators, trimming, and winsorization. The package 'robsurvey' complements the 'survey' package. The package additionally includes a weighted version of the resistant line function of base R (line()), as well as two median based simple regression estimators. The methods are described in Hulliger (1995) <https://www150.statcan.gc.ca/n1/en/catalogue/12-001-X199500114407/>.

License MIT + file LICENSE

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

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

Encoding UTF-8

LazyData true

Depends R (>= 3.5.0)

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

RoxygenNote 6.1.1

Suggests knitr, rmarkdown, testthat

VignetteBuilder knitr

NeedsCompilation yes

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

Maintainer Martin Sterchi <martin.sterchi@fhnw.ch>

Repository CRAN

Date/Publication 2019-06-11 07:40:04 UTC
\textbf{R topics documented:}

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>huberwgt</td>
<td>2</td>
</tr>
<tr>
<td>rht_control</td>
<td>4</td>
</tr>
<tr>
<td>robsurvey</td>
<td>5</td>
</tr>
<tr>
<td>robweights</td>
<td>5</td>
</tr>
<tr>
<td>trimwgt</td>
<td>6</td>
</tr>
<tr>
<td>weighted_line</td>
<td>7</td>
</tr>
<tr>
<td>weighted_mad</td>
<td>8</td>
</tr>
<tr>
<td>weighted_median</td>
<td>9</td>
</tr>
<tr>
<td>weighted_median_line</td>
<td>10</td>
</tr>
<tr>
<td>weighted_median_ratio</td>
<td>11</td>
</tr>
<tr>
<td>weighted_quantile</td>
<td>12</td>
</tr>
<tr>
<td>wgtmeantotal</td>
<td>13</td>
</tr>
<tr>
<td>winswgt</td>
<td>14</td>
</tr>
</tbody>
</table>

\textbf{Index} 16

\begin{itemize}
  \item \texttt{huberwgt} \hspace{1cm} \emph{Huber M-estimators of the weighted mean and weighted total}
\end{itemize}

\textbf{Description}

Weighted Huber M-estimators of the mean and total are available in two forms:

- \textbf{bare-bone} functions: \texttt{weighted_mean_huber} and \texttt{weighted_total_huber},
- \textbf{estimation methods}: \texttt{svymean_huber} and \texttt{svytotal_huber} (incl. variance estimation based on the functionality of the \texttt{survey} package).

\textbf{Usage}

```r
weighted_mean_huber(x, w, k = 1.5, type = "rht", info = FALSE, na.rm = FALSE, ...)
weighted_total_huber(x, w, k = 1.5, type = "rht", info = FALSE, na.rm = FALSE, ...)
svymean_huber(x, design, k = 1.5, type = "rht", ...)
svytotal_huber(x, design, k = 1.5, ...)
```

\textbf{Arguments}

- \texttt{x} \hspace{1cm} a numeric vector (\texttt{weighted.[total/mean].huber} or \texttt{weighted.[total/mean].huber}); a formula object or variable name (\texttt{svymean_huber} or \texttt{svytotal_huber})
- \texttt{w} \hspace{1cm} a numeric vector of weights
- \texttt{k} \hspace{1cm} a robustness tuning constant, \(k\) in \([0, \infty)\)
The `huberwgt` function provides a robust M-estimator of the Horvitz–Thompson total or the Hajek mean. It supports two types of estimation methods:

- `rht`: robust Horvitz-Thompson M-estimator of the total/mean
- `rwm`: robust weighted mean estimator of a Hajek-type estimator of the mean.

If the study variable $x$ is positively correlated with the inclusion probabilities, `rht` tends to be superior.

### Scale

M-estimators of location are not scale invariant. The unknown scale is estimated simultaneously with the estimate of location (mean or total) as the weighted median absolute deviation from the weighted median (MAD, see `weighted_mad`).

### Variance

Variance estimates of the mean or total estimator are computed as first-order linearization using the design-based-estimation capabilities available in the package `survey`.

### Tuning

Additional arguments can be passed (via `...`) to specify the control parameters (e.g., number of iterations, psi-function, etc.); see `rht_control` for details.

### Domain estimation

Estimates for domains can be obtained using the `svyby` wrapper in the `survey` package (see examples).

### Value

- An estimate (scalar) for `weighted.[total/mean].huber` (unless `info=TRUE`)
- An object of class `svystat.rob` for functions of the type `msvy[total/mean]`, i.e., a list including the following components: characteristic, estimator, estimate, variance, robust, optim, residuals, model, design, and call.

### Utility functions

For the methods `svymean_huber` and `svytotal_huber`, the following utility functions can be used:

- `summary` gives a summary of the estimation properties
- `robweights` retrieves the robustness weights
- `coef, vcov, residuals, and fitted` retrieve the estimate, variance, residuals and fitted values, respectively

### Note

`huberwgt` is a generic name for the functions documented.
rht_control

References

See Also
svymean_trimmed, svytotal_trimmed, svymean_winsorized, svytotal_winsorized, weighted_mean_trimmed, weighted_total_trimmed weighted_mean_winsorized, weighted_total_winsorized

Examples
library(survey)
data(api)
dstrat <- svydesign(id=-1, strata=-stype, weights=-pw, data=apistrat, fpc=-fpc)
svymean_huber(~api00, dstrat, k = 2)
# Domain estimates
svyby(~api00, by = ~stype, design = dstrat, svymean_huber, k = 1.34)

---

rht_control  
Control function for M-estimation (tuning parameters etc.)

Description
This function is called internally.

Usage
rht_control(acc = 1e-05, maxit = 100, psi = "Huber", ...)

Arguments
acc  
numeric tolerance, stoping rule in the iterative updating scheme (default: 1e-5)
maxit
maximum number of updating iterations
psi
psi-function (Huber or asymHuber)
...  
additional arguments

Details
Tuning parameters for weighted_mean_huber, weighted_total_huber, svymean_huber, svytotal_huber.

Value
List
Description

The package robsurvey is a collection of functions for robust survey statistics.

robsurvey functions

- robust Horvitz-Thompson M-estimator of mean and total in `svymean_huber()` and `svytotal_huber()`,
- robust trimmed Horvitz-Thompson estimator of mean and total in `svymean_trimmed()` and `svytotal_trimmed()`,
- robust winsorized Horvitz-Thompson estimator of mean and total in `svymean_winsorized()` and `svytotal_winsorized()`,
- weighted median estimator in `weighted_median()`,
- weighted quantile estimator in `weighted_quantile()`,
- weighted median absolute deviation in `weighted_mad()`,
- weighted mean and total estimators in `weighted_mean()` and `weighted_total()`.

References


Description

robweights retrieves the robustness weights from an M-estimator of class svystat.rob.

Usage

```r
robweights(object)
```

Arguments

- `object` class of type svystat.rob

Details

Extracts the robustness weights.

Value

Vector of robustness weights
Weighted trimmed mean and trimmed total

Description

Weighted trimmed estimators of the mean and total are available in two forms:

- **bare-bone functions**: `weighted_mean_trimmed` and `weighted_total_trimmed`,
- **estimation methods**: `svymean_trimmed` and `svytotal_trimmed` (incl. variance estimation based on the functionality of the `survey` package).

Usage

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

Arguments

- `x`: numeric vector (`weighted_mean_trimmed` or `weighted_total_trimmed`); a formula object or variable name (`svymean_trimmed` or `svytotal_trimmed`)
- `w`: numeric vector of weights
- `LB`: lower bound of trimming, such that \(0 \leq LB < UB \leq 1\)
- `UB`: upper bound of trimming, such that \(0 \leq LB < UB \leq 1\)
- `na.rm`: a logical value indicating whether NA values should be stripped before the computation proceeds.
- `design`: a `survey::design` object (see `svydesign` in `survey`)
- `...`: additional arguments (not used)

Details

**Overview** Robust trimmed Horvitz–Thompson total or Hajek mean

- bare-bone functions: return the estimate (no variance estimation)
- estimation methods on the basis of the `survey` package.

**Variance** Variance estimates of the mean or total estimator are computed as first-order linearization using the design-based-estimation capabilities available in package `survey`.

**Domain estimation** Estimates for domains can be obtained using the `svyby` wrapper in the `survey` package (see examples).
weighted_line

Value

Estimate (scalar) or object of class svystat.rob

Utility functions

For the methods svymean_trimmed and svytotal_trimmed, the following utility functions can be used

- `summary` gives a summary of the estimation properties
- `robweights` retrieves the robustness weights
- `coef`, `vcov`, `residuals`, and `fitted` retrieve, respectively, the estimate, variance, residuals and fitted values

Note

trimwgt is a generic name for the functions documented.

See Also

svymean_huber, svytotal_huber, svymean_winsorized, svytotal_winsorized, weighted_mean_huber, weighted_total_huber, weighted_mean_winsorized, weighted_total_winsorized

Examples

```r
library(survey)
data(api)
dstrat <- svydesign(id=~1, strata=~stype, weights=~pw, data=apistrat, fpc=~fpc)
svymean Trimmed(~api00, dstrat, LB = 0.05)
# Domain estimates
svyby(~api00, by = ~stype, design = dstrat, svymean Trimmed, LB = 0.1)
```

---

**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)`
weighted_mad

Arguments

- **x**: a numeric vector (explanatory variable)
- **y**: a numeric vector (response variable)
- **w**: a numeric vector of weights
- **na.rm**: a logical value indicating whether rows with NA values should be stripped before the computation proceeds
- **iter**: number of iterations for enhancing the slope

Details

Uses different quantiles for splitting the sample than `line()`. Is based on `weighted_median()`.

Value

intercept and slope of the fitted line

See Also

`line`

Examples

```r
data(cars)
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 absolute deviation from the weighted median

Usage

```r
weighted_mad(x, w, na.rm = FALSE, constant = 1.4826)
```

Arguments

- **x**: a numeric vector
- **w**: a numeric vector of weights
- **na.rm**: a logical value indicating whether NA values should be stripped before the computation proceeds.
- **constant**: (scale factor, default: 1.4826)
The weighted MAD is computed as the (normalized) weighted median of the absolute deviation from the weighted median; the median is computed as the weighted lower sample median (see \texttt{weighted_median}); the MAD is normalized to be an unbiased estimate of scale at the Gaussian core model.

**Value**

Weighted median absolute deviation from the (weighted) median

**See Also**

\texttt{weighted_median}

**Examples**

```r
x <- c(0.1, 0.35, 0.05, 0.1, 0.15, 0.05, 0.2)
weighted_mad(x, x)
```

---

**Description**

\texttt{weighted_median} computes a weighted median where the exact location corresponds exactly to a cumulative weight of 0.5. This yields a symmetric median.

**Usage**

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

**Arguments**

- \texttt{x} a numeric vector whose weighted sample median is wanted
- \texttt{w} a numeric vector of weights
- \texttt{na.rm} a logical value indicating whether NA values should be stripped before the computation proceeds.

**Details**

Note that the \texttt{weighted_median} function delivers a symmetric median while the \texttt{weighted_quantile} function with probability 0.5 delivers the lower median. Hence, the results of these two functions will generally differ.

**Value**

weighted sample median
See Also

weighted_quantile

Examples

```r
x <- c(0.1, 0.35, 0.05, 0.1, 0.15, 0.05, 0.2)
weighted_median(x, x)
```

weighted_median_line

Robust simple linear regression based on medians

Description

For type med slopes the median individual ratios response/explanatory is used as estimator of the slope. For version ratiomeds the ratio of the median crossproduct to the median of squares of the explanatory variable is used as the estimator of the slope. Survey weights may be used. Missing values are neglected.

Usage

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

Arguments

- `x` : a numeric vector (explanatory variable)
- `y` : a numeric vector (response variable)
- `w` : a numeric vector of weights
- `type` : either "slopes" (default) or "products"
- `na.rm` : a logical value indicating whether rows with NA values should be stripped before the computation proceeds

Details

Uses weighted_median(). The median of slopes (type="slopes") uses $b_1 = M((y - M(y, w))/(x - M(x, w)), w)$. The median of crossproducts by median of squares (type="products") uses $b_1 = M(((y - M(y, w))(x - M(x, w)), w)/M((x - M(x, w)^2), w)$, where $M(x, w)$ is shorthand for the function weighted_median(x, w). The function allows weights and missing values.

Value

a vector with two components: intercept and slope

See Also

line, weighted_line, weighted_median_ratio
weighted_median_ratio  

**Examples**

```r
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 based on median*

**Description**

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

**Usage**

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

**Arguments**

- `x`  
a numeric vector (explanatory variable)
- `y`  
a numeric vector (response variable)
- `w`  
a numeric vector of (optional) weights
- `na.rm`  
a logical value indicating whether rows with NA values should be stripped before the computation proceeds

**Value**

a vector with two components: intercept and slope
See Also

`line, weighted_line, weighted_median_line`

Examples

```r
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 lower sample quantiles

**Description**

`weighted_quantile` computes the weighted lower sample quantile

**Usage**

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

**Arguments**

- `x`: a numeric vector whose weighted sample quantiles are wanted
- `w`: a numeric vector of weights
- `probs`: a numeric vector of probabilities with values in [0,1]
- `na.rm`: a logical value indicating whether NA values should be stripped before the computation proceeds.

**Details**

Weighted lower quantiles are computed using an algorithm with $O(n \times \log(n))$ in worst-case time. There exist superior algorithms; see Cormen et al. (2009, Problem 9.2).

**Value**

Weighted sample quantiles

**References**


**See Also**

`weighted_median`
wgtmeantotal

Examples

x <- c(0.1, 0.35, 0.05, 0.1, 0.15, 0.05, 0.2)
weighted_quantile(x, x, probs = c(0.25, 0.5, 0.75))

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

Description

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

Usage

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

Arguments

x  a numeric vector
w  a numeric vector of weights
na.rm  a logical value indicating whether NA values should be stripped before the com-
      putation proceeds.

Details

-

Value

Estimate (scalar)

Note

wgtmeantotal is a generic name for the functions documented.

Examples

x <- c(0.1, 0.35, 0.05, 0.1, 0.15, 0.05, 0.2)
weighted_total(x, x)
x <- c(0.1, 0.35, 0.05, 0.1, 0.15, 0.05, 0.2)
weighted_mean(x, x)
Description

Weighted winsorized estimators of the mean and total are available in two forms:

- **bare-bone functions**: `weighted_mean_winsorized` and `weighted_total_winsorized`,
- **estimation methods**: `svymean_winsorized` and `svytotal_winsorized` (incl. variance estimation based on the functionality of the `survey` package).

Usage

```r
weighted_mean_winsorized(x, w, LB = 0.05, UB = 1 - LB, na.rm = FALSE)
weighted_total_winsorized(x, w, LB = 0.05, UB = 1 - LB, na.rm = FALSE)
svymean_winsorized(x, design, LB = 0.05, UB = 1 - LB, ...)
svytotal_winsorized(x, design, LB = 0.05, UB = 1 - LB, ...)
```

Arguments

- `x` numeric vector (`weighted_mean_winsorized` or `weighted_total_winsorized`); a formula object or variable name (`svymean_winsorized` or `svytotal_winsorized`)
- `w` numeric vector of weights
- `LB` lower bound of winsorizing, such that $0 \leq LB < UB \leq 1$
- `UB` upper bound of winsorizing, such that $0 \leq LB < UB \leq 1$
- `na.rm` a logical value indicating whether NA values should be stripped before the computation proceeds.
- `design` a survey.design object (see `svydesign` in `survey`)
- `...` additional arguments (not used)

Details

**Overview**  Robust winsorized Horvitz–Thompson total or Hajek mean

- bare-bone functions: return the estimate (no variance estimation)
- estimation methods on the basis of `survey` (incl. variance estimation)

**Variance**  Variance estimates of the mean or total estimator are computed as first-order linearization using the design-based-estimation capabilities available in package `survey`.

**Domain estimation**  Estimates for domains can be obtained using the `svyby` wrapper in the `survey` package (see examples).
Value

Estimate (scalar) or object of class svystat.rob

Utility functions

For the methods svymean_winsorized and svytotal_winsorized, the following utility functions can be used:

- summary gives a summary of the estimation properties
- robweights retrieves the robustness weights
- coef, vcov, residuals, and fitted retrieve, respectively, the estimate, variance, residuals and fitted values

Note

winswgt is a generic name for the functions documented.

See Also

svymean_hubber, svytotal_hubber, svymean_trimmed, svytotal_trimmed, weighted_mean_hubber, weighted_total_hubber, weighted_mean_trimmed, weighted_total_trimmed

Examples

library(survey)
data(api)
dstrat <- svydesign(id=~1, strata=~stype, weights=~pw, data=apistrat, fpc=~fpc)
svymean_winsorized(~api00, dstrat, LB = 0.05)
# Domain estimates
svyby(~api00, by = ~stype, design = dstrat, svymean_winsorized, LB = 0.1)
Index

huberwgt, 2
line, 8, 10, 12
rht_control, 3, 4
robsurvey, 5
robsurvey-package (robsurvey), 5
robweights, 3, 5, 7, 15
svyby, 3, 6, 14
svydesign, 3, 6, 14
svymean_hub, 4, 7, 15
svymean_hub (huberwgt), 2
svymean_trimmed, 4, 15
svymean_trimmed (trimwgt), 6
svymean_winsorized, 4, 7
svymean_winsorized (winwgt), 14
svytotal_hub, 4, 7, 15
svytotal_hub (huberwgt), 2
svytotal_trimmed, 4, 15
svytotal_trimmed (trimwgt), 6
svytotal_winsorized, 4, 7
svytotal_winsorized (winwgt), 14
trimwgt, 6
weighted_line, 7, 10, 12
weighted_mad, 3, 8
weighted_mean (wgtmeantotal), 13
weighted_mean_hub, 4, 7, 15
weighted_mean_hub (huberwgt), 2
weighted_mean_trimmed, 4, 15
weighted_mean_trimmed (trimwgt), 6
weighted_mean_winsorized, 4, 7
weighted_mean_winsorized (winwgt), 14
weighted_median, 9, 9, 12
weighted_median_line, 10, 12
weighted_median_ratio, 10, 11
weighted_quantile, 10, 12
weighted_total (wgtmeantotal), 13
weighted_total_hub, 4, 7, 15
weighted_total_trimated (trimwgt), 6
weighted_total_winsorized, 4, 7
weighted_total_winsorized (winwgt), 14
wgtmeantotal, 13
winwgt, 14