Package 'csranks'

March 22, 2023

Type Package

Title Confidence Sets for Ranks

Version 1.0.1

Description Construct confidence sets for positions of populations in a ranking based on values of a certain feature and their estimation errors. Both simultaneous and marginal confidence sets are available, as well as confidence sets with populations occupying top-n positions in the ranking. Theory based on Mogstad, Romano, Shaikh, and Wilhelm (2023)<doi:10.1093/restud/rdad006>.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

URL https://github.com/danielwilhelm/R-CS-ranks, https://danielwilhelm.github.io/R-CS-ranks/

BugReports https://github.com/danielwilhelm/R-CS-ranks/issues

RoxygenNote 7.2.1

Depends R (>= 3.5.0)

Imports stats, ggplot2, scales, MASS, cli

Suggests spelling, testthat (>= 2.1.0), grid, knitr, rmarkdown

VignetteBuilder knitr

Language en-US

NeedsCompilation no

Author Daniel Wilhelm [aut, cre], Pawel Morgen [aut]

Maintainer Daniel Wilhelm <d.wilhelm@ucl.ac.uk>

Repository CRAN

Date/Publication 2023-03-22 19:40:13 UTC

R topics documented:

`csranks` *Confidence sets for ranks*

Description

Given estimates and their covariance matrix of a certain feature for a set of populations, calculate confidence sets for the ranks of populations, where populations are ranked by the feature values.

Usage

```
csranks(
  x,
  Sigma,
  coverage = 0.95,
  cstype = "two-sided",
  stepdown = TRUE,
  R = 1000,
  simul = TRUE,
  indices = NA,
  na.rm = FALSE,
  seed = NA
)
```

Arguments

`x`	vector of estimates.
`Sigma`	covariance matrix of `x`. Note, that it must be covariance matrix of feature **means**, not features themselves.
`coverage`	nominal coverage of the confidence set. Default is 0.95.
`cstype`	type of confidence set (`two-sided`, `upper`, `lower`). Default is `two-sided`.
`stepdown`	logical; if `TRUE` (default), stepwise procedure is used, otherwise single step procedure is used. See Details section for more.
`R`	number of bootstrap replications. Default is 1000.
`simul`	logical; if `TRUE` (default), then simultaneous confidence sets are computed, which jointly cover all populations indicated by `indices`. Otherwise, for each population indicated in `indices` a marginal confidence set is computed.

indices	vector of indices of x for whose ranks the confidence sets are computed. indices=NA (default) means computation for all ranks.
na.rm	logical; if TRUE, then NA's are removed from x and Sigma (if any).
seed	seed for bootstrap random variable draws. If set to NA (default), then seed is not set.

Value

A csranks object, which is a list with three items:

L Lower bounds of the confidence sets for ranks indicated in indices

rank Raw rank estimates using irank with default parameters

U Upper bounds of the confidence sets.

Details

IMPORTANT: make sure, that the Sigma is a (perhaps estimated) covariance matrix of estimates of feature means across populations, not of features themselves. For example, sample of size n of a feature following a standard normal distribution has variance $\sigma^2 = 1$, but mean from such sample has variance $1/n$. We refer to the latter.

The command implements the procedure for construction of confidence sets for ranks described in the referenced paper below. Generally, it consists of verification of a large set of hypotheses. After rejection of certain set of hypotheses, one can terminate the procedure or keep verifying a smaller set of hypotheses that were not rejected so far. The former corresponds to stepdown=FALSE; the latter to stepdown=TRUE.

From a practical point of view, stepdown=TRUE takes more time, but usually results in tighter (better) confidence sets.

Parametric bootstrap used to calculate distribution for confidence sets based on the multivariate normal distribution.

References

Mogstad, Romano, Shaikh, and Wilhelm (2023), "Inference for Ranks with Applications to Mobility across Neighborhoods and Academic Achievements across Countries", forthcoming at Review of Economic Studies

pdf link doi:10.1093/restud/rdad006

Examples

```
# Setup example data
n <- 10
x <- seq(1, 3, length = n)
Sigma <- matrix(0.001, nrow = n, ncol = n)
diag(Sigma) <- 0.04

# Run csranks to get confidence sets for ranks of features
csranks(x, Sigma)
```

```
# If you assume that the feature measurements are independent
# (or have access only to variances / standard errors estimates),
# then pass a diagonal covariance matrix.
Sigma <- diag(rep(0.04, 10))
csranks(x, Sigma)
```

`csranks_multinom` *Confidence sets for ranks based on multinomial data*

Description

Given data on counts of successes for each category, calculate confidence sets for the ranks of categories, where categories are ranked by their success probabilities.

Usage

```
csranks_multinom(
  x,
  coverage = 0.95,
  cstype = "two-sided",
  simul = TRUE,
  multcorr = "Holm",
  indices = NA,
  na.rm = FALSE
)
```

Arguments

`x`	vector of counts of successes for each category
`coverage`	nominal coverage of the confidence set. Default is 0.95.
`cstype`	type of confidence set (`two-sided`, `upper`, `lower`). Default is `two-sided`.
`simul`	logical; if `TRUE` (default), then simultaneous confidence sets are computed, which jointly cover all populations indicated by `indices`. Otherwise, for each population indicated in `indices` a marginal confidence set is computed.
`multcorr`	multiplicity correction to be used: `Holm` (default) or `Bonferroni`. See Details section for more.
`indices`	vector of indices of `x` for whose ranks the confidence sets are computed. `indices=NA` (default) means computation for all ranks.
`na.rm`	logical; if `TRUE`, then `NA`'s are removed from `x` and `Sigma` (if any).

Value

A `csranks` object, which is a list with three items:

`L` Lower bounds of the confidence sets for ranks indicated in `indices`

`rank` Raw rank estimates using `irank` with default parameters

`U` Upper bounds of the confidence sets.

Details

The command implements the procedure for construction of confidence sets for ranks described in the referenced paper below.

It involves testing multiple hypotheses. The ‘`multcorr`‘ states, how the p-values should be corrected to control the Family Wise Error Rate (FWER).

From a practical point of view, `multcorr=Holm` takes more time, but usually results in tighter (better) confidence sets than `multcorr=Bonferroni`.

References

Bazylik, Mogstad, Romano, Shaikh, and Wilhelm. "Finite-and large-sample inference for ranks using multinomial data with an application to ranking political parties".

Examples

```
x <- c(rmultinom(1, 1000, 1:10))
csranks_multinom(x)
```

`cstaubest` *Projection confidence sets for the tau-best*

Description

Find a set of populations, which belong to tau-best populations according to some feature with given confidence.

Usage

```
cstaubest(
  x,
  Sigma,
  tau = 2,
  coverage = 0.95,
  stepdown = TRUE,
  R = 1000,
  na.rm = FALSE,
  seed = NA
)

cstauworst(
  x,
  Sigma,
  tau = 2,
  coverage = 0.95,
  stepdown = TRUE,
  R = 1000,
```

```
  na.rm = FALSE,
  seed = NA
)
```

Arguments

x	vector of estimates.
Sigma	covariance matrix of x. Note, that it must be covariance matrix of feature **means**, not features themselves.
tau	the confidence set contains indicators for the elements in x whose rank is less than or equal to tau.
coverage	nominal coverage of the confidence set. Default is 0.95.
stepdown	logical; if TRUE (default), stepwise procedure is used, otherwise single step procedure is used. See Details section for more.
R	number of bootstrap replications. Default is 1000.
na.rm	logical; if TRUE, then NA's are removed from x and Sigma (if any).
seed	seed for bootstrap random variable draws. If set to NA (default), then seed is not set.

Value

logical vector indicating which of the elements of x are in the confidence set for the tau-best.

Functions

- cstauworst(): Projection confidence sets for the tau-worst

  Similar method, but for populations, which are tau-worst. Equivalent to calling cstaubest with -x.

Details

The confidence set contains indicators for the elements in x whose rank is less than or equal to tau with probability approximately equal to the coverage indicated in coverage. Parametric bootstrap based on the multivariate normal distribution.

If na.rm=TRUE and NAs are present, then results are returned for tau-best (worst) populations among those without NA values, i.e. after NA removal.

References

Mogstad, Romano, Shaikh, and Wilhelm (2023), "Inference for Ranks with Applications to Mobility across Neighborhoods and Academic Achievements across Countries", forthcoming at Review of Economic Studies

pdf link doi:10.1093/restud/rdad006

Examples

```
# Setup example data
n <- 10
x <- seq(1, 3, length = n)
Sigma <- matrix(0.001, nrow = n, ncol = n)
diag(Sigma) <- 0.04

# Run csranks to get confidence sets for top 3 populations
cstaubest(x, Sigma, tau = 3)
cstauworst(x, Sigma, tau = 3)

# If you assume that the feature measurements are independent,
# (or just have access to variances / standard errors)
# then pass a diagonal covariance matrix.
Sigma <- diag(rep(0.04, 10))
cstaubest(x, Sigma, tau = 3)
cstauworst(x, Sigma, tau = 3)
```

---

irank *Compute ranks from feature values*

Description

Given estimates of a certain feature for a set of populations, calculate the integer ranks of populations, i.e. places in ranking done by feature values. The larger (or smaller) feature value, the higher the place and the lower the integer rank (lowest, 1, is the best place).

Usage

```
irank(x, omega = 0, increasing = FALSE, na.rm = FALSE)

frank(x, omega = 0, increasing = FALSE, na.rm = FALSE)
```

Arguments

x	vector of values to be ranked
omega	numeric; numeric value in [0,1], each corresponding to a different definition of the rank; default is 0. See Details.
increasing	logical; if TRUE, then large elements in x receive a large rank. In other words, larger values in x are lower in ranking. Otherwise, large elements receive small ranks.
na.rm	logical; if TRUE, then NA's are removed from x. In other case the output for NAs is NA and for other values it's for extreme possibilities that NA values are actually in first or last positions of ranking.

Details

`omega` (ω) value determines, how equal entries in `x` should be ranked; in other words how to handle ex aequo cases. If there are none, then the parameter does not affect the output of this function. For example, let's say, that n largest entries in `x` are equal. Those entries could receive (minimum) rank 1 or (maximum) rank n or some value in between.

Suppose, that we want to assign rank to n equal values in an array. Denote their minimum rank as r and maximum as $R = r + n - 1$. Then the assigned rank is an average of minimum and maximum rank, weighted by ω:

$$r(1 - \omega) + R\omega$$

Value

vector of the same length as `x` containing the ranks

Functions

- `frank()`: Compute fractional ranks

  This method returns ranks in form of fractions from [0-1] interval. Smaller values (closer to 0) indicate higher rank.

Examples

```
irank(c(4,3,1,10,7))
irank(c(4,3,1,10,7), omega=1) # equal to previous ranks because there are no ties
irank(c(4,3,1,10,7), omega=0.5) # equal to previous ranks because there are no ties
irank(c(4,4,4,3,1,10,7,7))
irank(c(4,4,4,3,1,10,7,7), omega=1)
irank(c(4,4,4,3,1,10,7,7), omega=0.5)
frank(c(4,3,1,10,7))
frank(c(4,3,1,10,7), omega=1) # equal to previous ranks because there are no ties
frank(c(4,3,1,10,7), omega=0.5) # mid-ranks, equal to previous ranks because there are no ties
frank(c(4,4,4,3,1,10,7,7))
frank(c(4,4,4,3,1,10,7,7), omega=1)
frank(c(4,4,4,3,1,10,7,7), omega=0.5) # mid-ranks
```

---

pisa *Cross-country comparison of students' achievement*

Description

A dataset containing average scores on math, reading, and science together with standard errors for all OECD countries. These are from the 2018 Program for International Student Assessment (PISA) study by the Organization for Economic Cooperation and Development (OECD). The average scores are over all 15-year-old students in the study.

Usage

```
pisa
```

Format

A data frame with 37 rows and 6 variables:

jurisdiction country, from which data was collected

math_score average score in math

math_se standard error for the average score in math

reading_score average score in reading

reading_se standard error for the average score in reading

science_score average score in science

science_se standard error for the average score in science

Source

https://www.oecd.org/pisa/data/

`plot.csranks` *Plot ranking with confidence sets*

Description

Display ranks together with their confidence set bounds.

Usage

```
## S3 method for class 'csranks'
plot(x, ...)

plotranking(
  ranks,
  L,
  U,
  popnames = NULL,
  title = NULL,
  subtitle = NULL,
  caption = NULL,
  colorbins = 1,
  horizontal = TRUE
)
```

Arguments

`x`	An `csranks` object, likely produced by `csranks`.
`...`	Other arguments, passed to `plotranking`.
`ranks`	vector of ranks

L	vector of lower bounds of confidence sets for the ranks
U	vector of lower bounds of confidence sets for the ranks
popnames	vector containing names of the populations whose ranks are in ranks. If popnames=NULL (default), then populations are automatically numbered.
title	character string containing the main title of the graph. title=NULL (default) means no title.
subtitle	character string containing the subtitle of the graph. subtitle=NULL (default) means no subtitle.
caption	character string containing the caption of the graph. caption=NULL (default) means no caption.
colorbins	integer indicating the number of quantile bins into which populations are grouped and color-coded. Value has to lie between 1 (default) and the number of populations.
horizontal	logical. Should be the bars displayed horizontally, or vertically?

Value

A ggplot plot displaying confidence sets.

Functions

- plot(csranks): Plot csranks output

Examples

```
x <- seq(1, 3, length = 10)
V <- diag(rep(0.04, 10))
CS <- csranks(x, V)
grid::current.viewport()
plot(CS)
# Equivalent:
plotranking(CS$rank, CS$L, CS$U)

# plotranking returns a ggplot object. It can be customized further:
library(ggplot2)
pl <- plot(CS)
pl + xlab("position in ranking") + ylab("population label") + theme_gray()

# horizontal = FALSE uses ggplot2::coord_flip underneath. The x and y axes swap places.
pl <- plot(CS, horizontal = FALSE)
pl + xlab("position in ranking") + # Note, that xlab refers to vertical axis now
  ylab("population label") + theme_gray()
```

Index