Package 'ordinalsimr'

January 29, 2025

Title Compare Ordinal Endpoints Using Simulations

Version 0.1.3

Description Simultaneously evaluate multiple ordinal outcome measures. Applied data analysts in particular are faced with uncertainty in choosing appropriate statistical tests for ordinal data. The included 'shiny' application allows users to simulate outcomes given different ordinal data distributions.

License MIT + file LICENSE

Imports assertthat, bslib, coin, config (>= 0.3.1), dplyr, DT, ggplot2, golem (>= 0.4.0), magrittr, rhandsontable, rlang, rms, shiny (>= 1.7.4), shinycssloaders, shinydashboard, shinyWidgets, stats, tidyr, utils, withr

Encoding UTF-8

LazyData true

LazyDataCompression xz

RoxygenNote 7.3.2

URL https://github.com/NeuroShepherd/ordinalsimr, https://neuroshepherd.github.io/ordinalsimr/

BugReports https://github.com/NeuroShepherd/ordinalsimr/issues

Suggests callr, knitr, pkgload, purrr, rmarkdown, testthat (>= 3.0.0), writexl

Config/testthat/edition 3

Depends R (>= 4.4.0)

VignetteBuilder knitr

NeedsCompilation no

Author Pat Callahan [aut, cre, cph] (<https://orcid.org/0000-0003-1769-7580>)

Maintainer Pat Callahan <patricktcallahan18@gmail.com>

Repository CRAN

Date/Publication 2025-01-29 18:40:02 UTC

Contents

`assign_groups` *Randomly assign groups*

Description

(Brief description of the function here.)

Usage

```
assign_groups(
  sample_size,
  sample_prob,
  prob0,
  prob1,
  seed,
  .rng_kind = NULL,
  .rng_normal_kind = NULL,
  .rng_sample_kind = NULL
)
```

Arguments

`sample_size`	total number of people under observation.
`sample_prob`	a vector of probability weights for obtaining the elements of the vector being sampled.
`prob0`	vector probability of each possible outcome for the null group
`prob1`	vector probability of each possible outcome for the intervention group
`seed`	integer specifying the seed number
`.rng_kind`	seeding info passed to withr::with_seed

`.rng_normal_kind`
: seeding info passed to withr::with_seed

`.rng_sample_kind`
: seeding info passed to withr::with_seed

Value

list of group assignments

`calculate_power_t2error` *Calculate Hypothesis Test Parameters*

Description

This function calculates the power, Type II error, and Type I error of tests given p-values. Power, Type II error, and confidence intervals calculated using ‘stats::binom.test()‘ which implements the Newcombe method.

Usage

```
calculate_power_t2error(
  df,
  alpha = 0.05,
  power_confidence_int = 95,
  n = NA_real_
)
```

Arguments

`df`	Data frame where each column is a vector of p-values from a statistical test
`alpha`	Numeric significance level; defaults to 0.05
`power_confidence_int`	confidence interval
`n`	Numeric value of sample size; repeated for convenience

Value

A data frame with columns for Type 1 error, Type 2 error, and power as well as rows for each test

`calculate_t1_error` *Calculate Type 1 Error*

Description

Calculate Type 1 error for a distribution, and the confidence interval around this estimate. Type I error and confidence intervals calculated using ‘stats::binom.test()‘ which implements the Newcombe method.

Usage

```
calculate_t1_error(
  df,
  alpha = 0.05,
  t1_error_confidence_int = 95,
  n = NA_real_
)
```

Arguments

`df`	data frame
`alpha`	significance level
`t1_error_confidence_int`	confidence interval
`n`	optional numeric input of

Value

data frame

`get_ordinalsimr_options` *Get ordinalsimr options*

Description

Returns all of the ordinalsimr options to the console.

Usage

```
get_ordinalsimr_options()
```

Value

list of ordinalsimr options

Examples

```
get_ordinalsimr_options()
```

`ordinal_tests` *Ordinal outcome tests*

Description

A metafunction that runs the statistical tests listed below, and returns the p-values as a named vector.

Usage

```
ordinal_tests(x, y, included = "all", ...)
```

Arguments

`x`	Group one
`y`	Group two
`included`	a character vector of the tests to be included. Default is "all"
`...`	Placeholder for additional arguments to functions

Details

- stats::wilcox.test()
- stats::fisher.test()
- stats::chisq.test(correct = FALSE)
- stats::chisq.test(correct = TRUE)
- rms::lrm()
- coin::independence_test(ytrafo = coin::rank_trafo)

Value

A named matrix of probabilities for each test

The function is designed to run all 6 tests by default. If you want to run only a subset of the tests, you can specify them in the ‘included‘ argument. The following values are possible:

- "Wilcoxon"
- "Fisher"
- "Chi Squared (No Correction)"
- "Chi Squared (Correction)"
- "Prop. Odds"
- "Coin Indep. Test"

This option is primarily for use in the Shiny application.

`parse_ratio_text` *Parse Ratio Text*

Description

This function parses text from ratios which are written in the format of 1-2 digit numbers separated by a colon and trailing with another 1-2 digit number. The text is processed into a numeric vector of length 2 containing the two numbers.

Usage

```
parse_ratio_text(text)
```

Arguments

`text`	A string of in the form of e.g. 5:95 or 70:30

Value

Numeric vector of length 2

Examples

```
parse_ratio_text("70:30")
```

`plot_distribution_results` *Plot Distribution*

Description

This function takes a wide table of p-values (i.e. one column for each statistical test), converts it to long format, and creates a density plot of the p-values by each test.

Usage

```
plot_distribution_results(df, alpha = 0.05, outlier_removal = 0.1)
```

Arguments

`df`	data frame where each column is a set of p-values for a different statistical test
`alpha`	numeric. significance level
`outlier_removal`	numeric. set x-axis scale maximum by proportion

Value

ggplot object

`plot_power` *Plot Test Power*

Description

Plot Test Power

Usage

```
plot_power(df, power_threshold = 0.8)
```

Arguments

`df`	a dataframe with p-values and a sample_size column
`power_threshold`	numeric. desired power threshold

Value

ggplot object

`run_app` *Run the Shiny Application*

Description

Run the Shiny Application

Usage

```
run_app(
  onStart = NULL,
  options = list(),
  enableBookmarking = NULL,
  uiPattern = "/",
  ...
)
```

Arguments

onStart	A function that will be called before the app is actually run. This is only needed for shinyAppObj, since in the shinyAppDir case, a global.R file can be used for this purpose.
options	Named options that should be passed to the runApp call (these can be any of the following: "port", "launch.browser", "host", "quiet", "display.mode" and "test.mode"). You can also specify width and height parameters which provide a hint to the embedding environment about the ideal height/width for the app.
enableBookmarking	Can be one of "url", "server", or "disable". The default value, NULL, will respect the setting from any previous calls to enableBookmarking(). See enableBookmarking() for more information on bookmarking your app.
uiPattern	A regular expression that will be applied to each GET request to determine whether the ui should be used to handle the request. Note that the entire request path must match the regular expression in order for the match to be considered successful.
...	arguments to pass to golem_opts. See '?golem::get_golem_options' for more details.

Value

NULL, the function is called for its side effects

`run_simulations` *Run Simulations*

Description

Run Simulations

Usage

```
run_simulations(
  sample_size,
  sample_prob,
  prob0,
  prob1,
  niter,
  included = "all",
  .rng_kind = NULL,
  .rng_normal_kind = NULL,
  .rng_sample_kind = NULL
)
```

Arguments

sample_size	Total number of trial participants
sample_prob	a vector of probability weights for obtaining the elements of the vector being sampled.
prob0	Vector of probabilities for control group
prob1	Vector of probabilities for intervention group
niter	Number of simulation iterations to complete#'
included	a character vector of the tests to be included. Default is "all"
.rng_kind	seeding info passed to withr::with_seed
.rng_normal_kind	seeding info passed to withr::with_seed
.rng_sample_kind	seeding info passed to withr::with_seed

Value

a list of lists; sub-list elements include ‘p_values‘ which is a matrix of p values for tests at each iteration, and ‘initial_groups‘ which is the group assignment information for each iteration

Examples

```
run_simulations(
  sample_size = c(40, 50, 60),
  sample_prob = c(0.5, 0.5),
  prob0 = c(0.1, 0.2, 0.3, 0.4),
  prob1 = c(0.6, 0.2, 0.1, 0.1),
  niter = 40
)
```

---

set_ordinalsimr_options

Set ordinalsimr Shiny App Default Values

Description

Set ordinalsimr Shiny App Default Values

Usage

```
set_ordinalsimr_options(
  default_iterations,
  default_size_min,
  default_size_max,
  default_ratio,
  default_distributions,
  default_entry_rows
)
```

Arguments

`default_iterations`
: number of iterations to run

`default_size_min`
: number for the small end of the sample size range

`default_size_max`
: number for the large end of the sample size range

`default_ratio`
: text ratio of the number of levels in the two groups, format of "50:50"

`default_distributions`
: data frame of the distributions of the levels in the two groups

`default_entry_rows`
: number of rows to initialize the (empty) data frame with

Value

invisible

Examples

```
# Set the default values for the ordinalsimr Shiny app

set_ordinalsimr_options(
  default_iterations = 1000,
  default_size_min = 10,
  default_size_max = 100,
  default_ratio = "50:50",
  default_distributions = data.frame(c(0.4, 0.3, 0.3), c(0.8, 0.1, 0.1))
)

# Values can be either overwritten or unset by setting them to NULL. The Shiny
# app still has backup values if these options are not set. Not all arguments
# need to be provided

set_ordinalsimr_options(
  default_iterations = 500, # Ex: update argument
  default_size_min = NULL, # Ex: unset argument
  default_size_max = NULL, # Ex: unset argument
  # default_ratio = NULL, # Ex: arg not provided (by commenting out)
  default_distributions = NULL
)
```

---

simulation_data_one_group

Simulation Data for One Group

Description

Simulated p-values and metadata for a two group comparison. Useful for Type I error calculations.

Usage

```
simulation_data_one_group
```

Format

'simulation_data_one_group' A list

p_values A data frame of p-values from each run of each test

initial_groups A nested list with information for each simulation run

---

`simulation_data_two_groups`

Simulation Data for Two Groups

Description

Simulated p-values and metadata for a two group comparison. Useful for Type II error and power calculations.

Usage

```
simulation_data_two_groups
```

Format

'simulation_data_two_groups' A list

p_values A data frame of p-values from each run of each test

initial_groups A nested list with information for each simulation run

Index