Package 'Counternull'

January 20, 2025

Type Package

Title Randomization-Based Inference

Version 0.2.12

Description Randomization-Based Inference for customized experiments. Computes Fisher-Exact P-Values alongside null randomization distributions. Retrieves counternull sets and generates counternull distributions. Computes Fisher Intervals and Fisher-Adjusted P-Values. Package includes visualization of randomization distributions and Fisher Intervals. Users can input custom test statistics and their own methods for randomization.
Rosenthal and Rubin (1994) <doi:10.1111/j.1467-9280.1994.tb00281.x>.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Depends R (>= 2.10)

RoxygenNote 7.2.3

Imports stats, effsize, ggplot2, randomizr, dplyr, tidyr

URL https://github.com/ymabene/Counternull

BugReports https://github.com/ymabene/Counternull/issues

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

VignetteBuilder knitr

NeedsCompilation no

Author Mabene Yasmine [aut, cre],
Bind Marie [aut],
Harvard University [cph]

Maintainer Mabene Yasmine <ymabene@stanford.edu>

Repository CRAN

Date/Publication 2024-02-24 21:20:12 UTC

Contents

`adjust_pvalues` *Compute Fisher-Adjusted P-Values for Multiple Testing*

Description

Adjusts p-values obtained from multiple comparisons. Computes Fisher-Adjusted P-Values utilizing randomization-based method (Lee et al., 2017).

Usage

```
adjust_pvalues(ls, bw = NULL)
```

Arguments

`ls`	List of "null_rand" objects
`bw`	Histogram bin width (optional)

Details

Argument "ls" must have a "null_rand" object for each p-value that needs to be adjusted.

Function plots joint p-value distribution.

Value

Vector with adjusted p-values

References

doi:10.5705/ss.202016.0116

Examples

```
y = sample_data$turn_angle
w = sample_data$w
n_one = create_null_rand(y, w, sample_matrix, test_stat = c("t"))
y = sample_data$turn_angle
w = sample_data$w
fun = function(x,y){
  return(invisible(ks.test(x,y)$statistic))
}
n_two = create_null_rand(y, w, sample_matrix, fun = fun,
alternative = c("greater"))
adjust_pvalues(list(n_one,n_two))
```

`create_fisher_interval` *Compute Fisher Interval*

Description

Computes Fisher (Fidicual) Interval and returns object of "fisher_interval" class.

Usage

```
create_fisher_interval(null_r, alpha = NULL, width = NULL)
```

Arguments

`null_r`	"null_rand" object corresponding to data used for interval
`alpha`	Significance level for Fisher Interval (default = .05 for 95% confidence)
`width`	Integer indicating the number of values to search for to construct Fisher Interval. Default value = 10000. (Increasing this argument may result in increased accuracy of interval.) (Optional)

Details

Call summary on "fisher_interval" class to retrieve information on the Fisher Interval. Call plot on "fisher_interval" class for visualization of Fisher Interval.

Use "create_null_rand" function to produce "null_rand" object for first argument.

Note: The warning 'Fisher Interval coverage is smaller than specified' indicates that there are no effect sizes found that match the p-value bounds for the specified significance level (ie. .025 and .975 for alpha = .05). In this case, the largest possible interval found under the significance level alpha will be returned. Check "pvalue_lower" and "pvalue_upper" parameters to see which p-value bounds are used.

Value

Class "fisher_interval" with 4 entries:

lower_bound Lower bound of Fisher Interval

upper_bound Upper bound of Fisher Interval

alpha Specified significance value

pvalue_lower P-value corresponding to lower bound of interval

pvalue_upper P-value corresponding to upper bound of interval

range Range of effect values tested

null_r Specified "null_rand" object

References

doi:10.48550/arXiv.2105.03996

Examples

```
y = sample_data$turn_angle
w = sample_data$w
n_r = create_null_rand(y,w, sample_matrix, test_stat = c("diffmeans"))
f= create_fisher_interval(n_r)
summary(f)
plot(f)
```

`create_null_rand` *Create Null Randomization Distribution*

Description

Generates null randomization distribution for a given test statistic.

Usage

```
create_null_rand(
  y,
  w,
  rand_matrix,
  test_stat = NULL,
  fun = NULL,
  alternative = NULL,
  bw = NULL
)
```

Arguments

y	Vector of observed outcomes
w	Vector indicating treatment assignments
rand_matrix	Matrix with permutations for experiment assignments
test_stat	Name of built in test statistic function. Provide "diffmeans" for difference of means, "t" for t test, "paired-t" for paired t test, and "cohens-d" for cohen's d test (optional).
fun	Test statistic function (optional).
alternative	Character string specifying alternative hypothesis. Must be one of "two-sided" (default), "greater", or "less".
bw	Bin width for histogram (optional)

Details

Call summary on "null_rand" class to retrieve information on the null randomization distribution. Call plot on "null_rand" class for visualization of null randomization distribution.

Assignments must be indicated in arguments "w" and "rand_matrix" using numeric 1 or 0.

Argument "rand_matrix" must have assignment permutations in each column and must have the same number of rows as there are entries in "w".

One of either argument "test_stat" or "fun" must be specified.

Argument "fun" must take in two parameters (treated outcomes and control outcomes) and returns a numeric test statistic value (scalar).

Value

Class "null_rand" with 11 entries:

null_dist Vector of permuted test statistics under the null hypothesis

t_obs Observed test statistic

counts Number of test statistics more extreme than observed test statistic

pvalue Fisher-Exact P-value

alternative Specified alternative

rand_matrix Randomization matrix used to generate null distribution

bin_width Specified bin width

y Observed outcomes

w Vector indicating treatment assignments

test_stat Name of built in test statistic function

fun Test statistic function

Examples

```
y = sample_data$turn_angle
w = sample_data$w
n_r = create_null_rand(y, w, sample_matrix, test_stat = c("t"))
summary(n_r)
plot(n_r)
```

`create_randomization_matrix` *Create Randomization Matrix*

Description

Creates randomization matrix of assignments for given number of units and permutations.Returns matrix with unique randomized permutations.

Usage

```
create_randomization_matrix(units, n, block = NULL)
```

Arguments

`units`	Number of units in dataset
`n`	Number of permutations
`block`	Numeric vector with length equal to "units" indicating block assignments for each unit (optional)

Details

Note, if the number of specified permutations exceeds the maximum number of unique permutations, the matrix returned will contain the maximum number of permutations.

Value

Matrix with unique randomized permutations

Examples

```
create_randomization_matrix(14,128,rep(1:7, each = 2))
```

`find_counternull_values` *Find Counternull Values*

Description

Retrieves counternull value set and returns object of "counternull" class.

Usage

```
find_counternull_values(null_r, counts = NULL, width = NULL, bw = NULL)
```

Arguments

`null_r`	"null_rand" object corresponding to data
`counts`	Vector containing lower and upper bounds for number of test statistics more extreme than observed test statistic in counternull randomization distribution (optional)
`width`	Integer indicating the number of values to search for to retrieve counternull set. Default value = 10000. (Increasing this argument may result in additional counternull values being found.) (optional)
`bw`	Histogram bin width (optional)

Details

Call summary on "counternull" class to retrieve range of counternull values. Call plot on "counternull" class for visualization of counternull distribution.

Argument "counts" must contain whole numbers for bounds.Lower bound must be smaller than upper bound. If argument is not specified, counternull values will be obtained using the "counts" argument from the specified "null_rand" argument.

If no counternull values are found, all entries in class are set to null. If only one set of counternull values are found, "perm_two", low_two" and "high_two" are set to null.

Value

Class "counternull" with 6 entries:

counternull_perm Counternull test statistics for first counternull set

low Counternull test statistics for second counternull set

high Lower bound of counternull set

counternull_perm_two Upper bound of counternull set

low_two Lower bound of second counternull set

high_two Upper bound of second counternull set

null_rand Specified "null_rand" object

bw Specified bin width

References

doi:10.1111/j.14679280.1994.tb00281.x

Examples

```
n_r = create_null_rand(sample_data$turn_angle, sample_data$w,
sample_matrix, test_stat = c("diffmeans"))
c = find_counternull_values(n_r)
summary(c)
plot(c)
c = find_counternull_values(n_r, c(56,60))
summary(c)
```

`find_test_stat` *Calculate Observed Test Statistic*

Description

Finds observed test statistic using treatment and control outcomes

Usage

```
find_test_stat(y, w, test_stat = NULL, fun = NULL)
```

Arguments

`y`	Vector of observed outcomes
`w`	Vector indicating treatment assignments
`test_stat`	Name of built in test statistic function. Provide "diffmeans" for difference of means, "t" for t test, "paired-t" for paired t test, and "cohens-d" for cohen's d test (optional)
`fun`	Test statistic function (optional)

Details

Assignments must be indicated in argument "w" using numeric 1 or 0.

One of either argument "test_stat" or "fun" must be specified.

Argument "fun" must take in two parameters (treated outcomes and control outcomes) and returns a numeric test statistic value (scalar).

Value

Observed test statistic (numeric)

Examples

```
find_test_stat(sample_data$turn_angle, sample_data$w,
test_stat = c("diffmeans"))

find_test_stat(sample_data$turn_angle, sample_data$w,
 test_stat = c("t"))
```

`sample_data` *Sample Data*

Description

This dataset is for an experiment measuring the effect of flashing lights on the movement of fish. It includes the treatment assignments of 156 fish and the turn angles of each fish when swimming.

Usage

```
sample_data
```

Format

A table with fish treatment assignments and turn angles:

w Treatment Assignment: 1 indicates exposure to flashing light and 0 indicates no exposure

turn_angle Angle at which the fish swim

`sample_matrix` *Sample Randomization Matrix*

Description

This is a randomization matrix for an experiment conducted on 156 fish. This matrix contains 1,000 possible treatment assignments for each fish.

Usage

```
sample_matrix
```

Format

A matrix with 1,000 columns:

1 Fish is Exposed to Flashing Light

0 Fish is Not Exposed to Flashing Light

Index