Package ‘doseSens’

October 10, 2025

Title Conduct Sensitivity Analysis with Continuous Exposures and Binary or Continuous Outcomes

Version 1.0.0

Description Performs sensitivity analysis for the sharp null, attributable effects, and weak nulls in matched studies with continuous exposures and binary or continuous outcomes as described in Zhang, Small, Heng (2024) <doi:10.48550/arXiv.2401.06909> and Zhang, Heng (2024) <doi:10.48550/arXiv.240
tions require installation of the 'Gurobi' optimizer. Please see <https://docs.gurobi.com/current/#refman/ins_the_r_package.html> for guidance.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.2.1

LazyData true

Imports dplyr, gtools, stats, utils, nloptr, lpSolve

Suggests gurobi

Depends R (>= 4.1.0)

NeedsCompilation no

Author Jeffrey Zhang [aut, cre]

Maintainer Jeffrey Zhang <jeffreyzhang1226@gmail.com>

Repository CRAN

Date/Publication 2025-10-10 08:30:07 UTC

Contents

apply_permutation_to_matrix

A function that returns a permuted vector according to a matrix and permutation vector.

Description

A function that returns a permuted vector according to a matrix and permutation vector.

Usage

```
apply_permutation_to_matrix(M, p)
```

Arguments

`M`	a square matrix of dimension n
`p`	a permuted version of the vector from 1:n

Value

a length n vector with ith entry corresponding to the i, p_i entry in M

Examples

```
mat <- matrix(1:9, nrow = 3, ncol = 3)
perm <- c(2, 1, 3)
permuted <- apply_permutation_to_matrix(M = mat, p = perm)
```

---

binary_thresh_attribute

Separable algorithm for threshold attributable effect in a sensitivity analysis with at most one over-exposed unit in each matched set. For a greater than alternative, finds the 'a' matched sets that most decrease the mean and/or variance.

Description

Separable algorithm for threshold attributable effect in a sensitivity analysis with at most one over-exposed unit in each matched set. For a greater than alternative, finds the 'a' matched sets that most decrease the mean and/or variance.

Usage

```
binary_thresh_attribute(
  Z,
  Q,
  index,
  gamma,
  thresh = 0,
  a = 1,
  trans = identity,
  mc = 50000
)
```

Arguments

Z	A length N vector of (nonnegative) observed doses.
Q	A length N vector of observed binary outcomes.
index	A length N vector of indices indicating matched set membership.
gamma	The nonnegative sensitivity parameter; gamma = 0 means no unmeasured confounding.
thresh	The dose threshold for the TAE.
a	The number of attributable effects to test for.
trans	The transformation of the doses to use for the test statistic. Default is the identity function.
mc	Number of monte-carlo samples if testing the sharp null, i.e. a = 0.

Value

Either "reject" if the value a is deemed not plausible/compatible, "feasible" if the value a is deemed so, else a list containing a p-value and dataframe of matched sets that have contribution to the test statistic sorted in order of smallest mean reduction followed by smallest variance reduction.

Examples

```
# Load the data
data <- lead_crime
# Solve by the separable algorithm
solution <- binary_thresh_attribute(data$log_lead, data$complain, data$matched_sets,
gamma = 0, thresh = log(3.5), a = 5, trans = identity)
```

change_Delta *A helper that takes a gurobi model object outputted from dose_attributable_general or dose_thresh_attributable_one_sided and changes the Delta parameter. Saves computation time by directly editing the constraint involving Delta without having to reinitialize the other constraints that are kept the same. Outputs a list analogous to output from dose_attributable_general or dose_thresh_attributable_one_sided.*

Description

A helper that takes a gurobi model object outputted from dose_attributable_general or dose_thresh_attributable_one_sided and changes the Delta parameter. Saves computation time by directly editing the constraint involving Delta without having to reinitialize the other constraints that are kept the same. Outputs a list analogous to output from dose_attributable_general or dose_thresh_attributable_one_sided.

Usage

```
change_Delta(model, Delta, direction = "equal", TT)
```

Arguments

model	A gurobi model object outputted from dose_attributable_general.
Delta	The new Delta to test for.
direction	The new direction to test
TT	The observed test statistic.

Value

A gurobi model and solution.

`constant_effects_test` *Asymptotic sensitivity analysis for weak nulls with continuous exposures assuming constant effects across matched sets.*

Description

Asymptotic sensitivity analysis for weak nulls with continuous exposures assuming constant effects across matched sets.

Usage

```
constant_effects_test(
  Z,
  R,
  index,
  gamma = 0,
  theta = 0,
  X = NA,
  estimand_function = extract_OLS,
  gamma_star_vec = NULL
)
```

Arguments

`Z` A length N vector of observed doses.

`R` A length N vector of observed outcomes.

`index` A length N vector of indices indicating matched set membership.

`gamma` The nonnegative sensitivity parameter; gamma = 0 means no unmeasured confounding.

`theta` The value at which to test the weak null.

`X` A matrix with I rows and less than I columns that contains covariate information.

`estimand_function`
A function that takes in values z and r and outputs a scalar; this function governs the causal estimand to estimate

`gamma_star_vec` that contains the maximum ratio of any two probabilities of permutations for each matched set.

Value

A list containing the deviate, one-sided p-value, observed value of the test statistic in each matched set, and conservative standard deviation estimate.

Examples

```
 # Load the data
data <- lead_bmd
# prepare data
threshold <- log(0.74675)
match_info = data |> dplyr::group_by(matched_sets) |>
dplyr::summarise(below = sum(log_lead < threshold) > 0, disc = var(log_lead) > 0,
above = sum(log_lead > threshold) > 0)
below_indices <- match_info$matched_sets[match_info$below]
disc_indices <- match_info$matched_sets[match_info$disc]
above_indices <- match_info$matched_sets[match_info$above]
# outcome analysis using the stochastic intervention statistic, weak null
below_nbp <- data |> dplyr::filter(matched_sets %in% below_indices & matched_sets
%in% disc_indices)
above_below <- below_nbp |> dplyr::filter(matched_sets %in% above_indices)
extract_below_threshold_vs_baseline_function <- function(z, r) {
  extract_below_threshold_vs_baseline(z, r, threshold)
}
# one-sided test that estimand defined by estimand_function is 0 at gamma = 0
result <- constant_effects_test(Z = above_below$log_lead,
R = above_below$lumbar_spine_bmd,
index = above_below$matched_sets, gamma = 0, theta = 0,
estimand_function = extract_below_threshold_vs_baseline_function)
```

dev_TV

Computes deviation from uniform distribution in total variation distance for a given amount of unmeasured confounding and a greater than alternative with a binary outcome.

Description

Computes deviation from uniform distribution in total variation distance for a given amount of unmeasured confounding and a greater than alternative with a binary outcome.

Usage

```
dev_TV(Z, Q, index, gamma, direct = "upper")
```

Arguments

Z	A length N vector of (nonnegative) observed doses.
Q	A length N vector of observed binary outcomes.
index	A length N vector of indices indicating matched set membership.
gamma	The nonnegative sensitivity parameter; gamma = 0 means no unmeasured confounding.
direct	The direction of the test - "upper" or "lower"; default is upper.

Value

A vector of length equaling the number of matched sets consisting of the TV distance from the uniform for each matched set at gamma level of unmeasured confounding for the worst-case.

Examples

```
# Load the data
data <- lead_crime
# compute total variation distances.
total_variation <- dev_TV(data$log_lead, data$complain,
data$matched_sets, gamma = log(1.5))
```

---

dose_attributable_general

Inference for general attributable effects in sensitivity analysis with continuous exposures and binary outcomes. Gurobi must be installed to use this function.

Description

Inference for general attributable effects in sensitivity analysis with continuous exposures and binary outcomes. Gurobi must be installed to use this function.

Usage

```
dose_attributable_general(
  Z,
  Q,
  index,
  gamma,
  alpha = 0.05,
  monotone = TRUE,
  Delta,
  sign = "positive",
  trans = identity,
  direction = "equal",
  M = 10000,
  eps = 0.01,
  alternative = "both",
  mv_threshold = NA,
  baseline = 0,
  relax = FALSE,
  feasible = TRUE,
  MIPgap = 0.01,
  WorkLimit = 1000,
  OutputFlag = 0
)
```

Arguments

`Z` A length N vector of (nonnegative) observed doses.

`Q` A length N vector of observed binary outcomes.

`index` A length N vector of indices indicating matched set membership.

`gamma` The nonnegative sensitivity parameter; gamma = 0 means no unmeasured confounding.

`alpha` Level of the test.

`monotone` Whether to impose a monotonicity constraint on the potential outcomes

`Delta` A numeric for the attributable effect to be tested for.

`sign` The sign of monotonicity - "positive" or "negative".

`trans` The transformation of the doses to use for the test statistic. Default is the identity function.

`direction` A string indicating the direction of testing the attributable effect with respect to Delta; "greater", "equal", or "less"

`M` The numeric penalty parameter to ensure correct sign relationship.

`eps` precision parameter for the objective function if solving feasible = "Yes"

`alternative` For constraining the rejection region using the test statistic with baseline potential outcomes, should it be constructed with "lower" alternative, "upper" alternative, or "both."

`mv_threshold` The number of allowed monotonicity violations.

`baseline` The baseline dose level for defining the threshold attributable effect.

`relax` Whether to solve the continuous relaxation.

`feasible` Whether to look for a feasible solution or to find the optimal solution.

`MIPgap` Gurobi parameter specifying the precision of the mixed integer program solution; default is 0.01.

`WorkLimit` Gurobi parameter specifying the maximum work units before stopping; default is 1000.

`OutputFlag` 0 if Gurobi print output is desired, 1 otherwise; default 0.

Value

A list containing the following:

`sol` A gurobi object containing the solution to the optimization. If feasible is TRUE, then failing to find a solution indiciates rejection. If feasible is FALSE, then optimal value greater than zero indiciates rejection.

`attribut` The attributable effect attained by the least rejectable allocation of potential outcomes and unmeasured confounders.

`search` A list of length the number of matched sets containing the matrix of compatible baseline potential outcomes in each matched set.

`null_exp` The null expectation of the pivot attained by the least rejectable allocation of potential outcomes and unmeasured confounders.

`obsT` The value of the pivot attained by the least rejectable allocation of potential outcomes and unmeasured confounders.

`nps` A vector with length the number of matched sets containing the size of each matched set.

`start.ind` A vector with length the number of matched sets containing the starting index of the decision variables pertaining to each matched set in the optimization program.

`exp_upper` The expectation of the pivot attained by the unmeasured confounders equalling the baseline potential outcome.

`exp_lower` The expectation of the pivot attained by the unmeasured confounders equalling 1 minus the baseline potential outcome.

`var_upper` The variance of the pivot attained by the unmeasured confounders equalling the baseline potential outcome.

`var_lower` The variance of the pivot attained by the unmeasured confounders equalling 1 minus the baseline potential outcome.

`dose` A list of length the number of matched sets containing the vector of doses observed in each matched set.

`model` The initialized gurobi model.

Examples

```
# To run the following example, Gurobi must be installed.
# Load the data
data <- lead_crime
# Make a threshold at log(3.5) transformation function.
above = function(Z) { return(Z > log(3.5)) }
# Solve the mixed-integer program.

solution = tryCatch(dose_attributable_general(data$log_lead,
data$complain, data$matched_sets, gamma=log(1),
alpha = 0.1, monotone = TRUE, trans = above,
Delta = 5, direction = ”less”, M = 10000, eps = 0.0001,
alternative = ”upper”, relax = FALSE, feasible = FALSE),
error = function(e) NULL
  )
```

dose_sensitivity_mc_gen

Sharp null monte-carlo sensitivity analysis for continuous exposures and binary outcomes.

Description

Sharp null monte-carlo sensitivity analysis for continuous exposures and binary outcomes.

Usage

```
dose_sensitivity_mc_gen(
  Z,
  Q,
  index,
  mc,
  gamma,
  weights = NA,
  obsT = NULL,
  trans = identity,
  direct = "upper",
  seed = 1,
  verbose = FALSE
)
```

Arguments

`Z`	A length N vector of (nonnegative) observed doses.
`Q`	A length N vector of observed binary outcomes.
`index`	A length N vector of indices indicating matched set membership.
`mc`	An integer for the total number of Monte-Carlo samples desired.
`gamma`	The nonnegative sensitivity parameter; gamma = 0 means no unmeasured confounding.
`weights`	Weights for each stratum to apply for the test statistic
`obsT`	The observed value of the test statistic; default is NULL
`trans`	The transformation of the doses to use for the test statistic. Default is the identity function.
`direct`	The direction of the test - "upper" or "lower"; default is upper.
`seed`	seed for random number generation.
`verbose`	Whether to print status updates or not; default is FALSE.

Value

A list containing two objects:

`mc`	A length mc vector containing the monte-carlo samples of the test statistic.
`p`	The monte-carlo p-value.

Examples

```
# Load the data
data <- lead_crime
# Make a threshold at log(3.5) transformation function.
above = function(Z) { return(Z > log(3.5)) }
# Conduct randomization test.
solution <- dose_sensitivity_mc_gen(data$log_lead, data$complain, data$matched_sets,
mc = 250, gamma = 0, trans = above)
```

dose_thresh_attributable_one_sided

Inference for threshold attributable effects in sensitivity analysis with continuous exposures and binary outcomes. Gurobi must be installed to use this function.

Description

Inference for threshold attributable effects in sensitivity analysis with continuous exposures and binary outcomes. Gurobi must be installed to use this function.

Usage

```
dose_thresh_attributable_one_sided(
  Z,
  Q,
  index,
  gamma,
  alpha = 0.05,
  monotone = TRUE,
  Delta,
  sign = "positive",
  direction = "equal",
  threshold = 0,
  M = 10000,
  eps = 0.01,
  mv_threshold = NA,
  baseline = 0,
  relax = FALSE,
  feasible = TRUE,
  MIPgap = 0.01,
  WorkLimit = 1000,
  OutputFlag = 0
)
```

Arguments

Z	A length N vector of (nonnegative) observed doses.
Q	A length N vector of observed binary outcomes.
index	A length N vector of indices indicating matched set membership.
gamma	The nonnegative sensitivity parameter; gamma = 0 means no unmeasured confounding.
alpha	Level of the test.

`monotone` Whether to impose a monotonicity constraint on the potential outcomes

`Delta` A numeric for the attributable effect to be tested for.

`sign` The sign of monotonicity - "positive" or "negative".

`direction` A string indicating the direction of testing the attributable effect with respect to Delta; "greater", "equal", or "less"

`threshold` The threshold for the TAE.

`M` The numeric penalty parameter to ensure correct sign relationship.

`eps` precision parameter for the objective function if solving feasible = "Yes"

`mv_threshold` The number of allowed monotonicity violations.

`baseline` The baseline dose level for defining the threshold attributable effect.

`relax` Whether to solve the continuous relaxation.

`feasible` Whether to look for a feasible solution or to find the optima.

`MIPgap` Gurobi parameter specifying the precision of the mixed integer program solution; default is 0.01.

`WorkLimit` Gurobi parameter specifying the maximum work units before stopping; default is 1000.

`OutputFlag` 0 if Gurobi print output is desired, 1 otherwise; default 0.

Value

A list containing the following:

`sol` A gurobi object containing the solution to the optimization. If feasible is TRUE, then failing to find a solution indiciates rejection. If feasible is FALSE, then optimal value greater than zero indiciates rejection.

`attribut` The attributable effect attained by the least rejectable allocation of potential outcomes and unmeasured confounders.

`search` A list of length the number of matched sets containing the matrix of compatible baseline potential outcomes in each matched set.

`null_exp` The null expectation of the pivot attained by the least rejectable allocation of potential outcomes and unmeasured confounders.

`obsT` The value of the pivot attained by the least rejectable allocation of potential outcomes and unmeasured confounders.

`nps` A vector with length the number of matched sets containing the size of each matched set.

`start.ind` A vector with length the number of matched sets containing the starting index of the decision variables pertaining to each matched set in the optimization program.

`exp_upper` The expectation of the pivot attained by the unmeasured confounders equalling the baseline potential outcome.

`var_upper` The variance of the pivot attained by the unmeasured confounders equalling the baseline potential outcome.

`dose` A list of length the number of matched sets containing the vector of doses observed in each matched set.

`model` The initialized gurobi model.

Examples

```
# To run the following example, Gurobi must be installed.
# Load the data
data <- lead_crime
# Solve the mixed-integer program.

solution = tryCatch(dose_thresh_attributable_one_sided(data$log_lead,
data$complain, data$matched_sets,
gamma=log(1), alpha = 0.1, monotone = TRUE, Delta = 5,
 direction = "less", threshold = log(3.5),M = 10000,
 eps = 0.0001,relax = FALSE, feasible = FALSE),
    error = function(e) NULL
 )
```

extract_above_threshold_vs_baseline

Compute average of outcomes above dose threshold minus average of outcomes.

Description

Compute average of outcomes above dose threshold minus average of outcomes.

Usage

```
extract_above_threshold_vs_baseline(z, r, threshold)
```

Arguments

z	a vector of doses
r	a vector of outcomes
threshold	a dose threshold

Value

the average of the outcomes with dose z above threshold c minus the average of the outcomes r.

Examples

```
# dose vector
dose <- c(0, 0.1, 0.4)
# outcome vector
outcome <- c(1, 1.1, 1.5)
theta <- extract_above_threshold_vs_baseline(z = dose, r = outcome, threshold = 0.3)
```

extract_below_threshold_vs_baseline

Compute average of outcomes below dose threshold minus average of outcomes.

Description

Compute average of outcomes below dose threshold minus average of outcomes.

Usage

```
extract_below_threshold_vs_baseline(z, r, threshold)
```

Arguments

`z`	a vector of doses
`r`	a vector of outcomes
`threshold`	a dose threshold

Value

the average of the outcomes with dose z below threshold c minus the average of the outcomes r.

Examples

```
# dose vector
dose <- c(0, 0.1, 0.4)
# outcome vector
outcome <- c(1, 1.1, 1.5)
theta <- extract_below_threshold_vs_baseline(z = dose, r = outcome, threshold = 0.3)
```

extract_max_vs_baseline

Compute largest dose outcome minus average of other outcomes.

Description

Compute largest dose outcome minus average of other outcomes.

Usage

```
extract_max_vs_baseline(z, r)
```

Arguments

`z`	a vector of doses
`r`	a vector of outcomes

Value

the outcome r corresponding to the largest dose z minus the average of the outcomes r.

Examples

```
# dose vector
dose <- c(0, 0.1, 0.4)
# outcome vector
outcome <- c(1, 1.1, 1.5)
theta <- extract_max_vs_baseline(z = dose, r = outcome)
```

extract_min_vs_baseline *Compute smallest dose outcome minus average of other outcomes.*

Description

Compute smallest dose outcome minus average of other outcomes.

Usage

```
extract_min_vs_baseline(z, r)
```

Arguments

z	a vector of doses
r	a vector of outcomes

Value

the outcome r corresponding to the smallest dose z minus the average of the outcomes r.

Examples

```
# dose vector
dose <- c(0, 0.1, 0.4)
# outcome vector
outcome <- c(1, 1.1, 1.5)
theta <- extract_min_vs_baseline(z = dose, r = outcome)
```

extract_OLS *A function that returns the coefficient from regressing an outcome vector on a dose vector.*

Description

A function that returns the coefficient from regressing an outcome vector on a dose vector.

Usage

```
extract_OLS(z, r)
```

Arguments

z	a vector of doses
r	a vector of outcomes

Value

the OLS regression coefficient from regressing r on z.

Examples

```
# dose vector
dose <- c(0, 0.1, 0.4)
# outcome vector
outcome <- c(1, 1.1, 1.5)
beta <- extract_OLS(z = dose, r = outcome)
```

extract_stochastic_intervention *Compute weighted sum of outcomes.*

Description

Compute weighted sum of outcomes.

Usage

```
extract_stochastic_intervention(z, r, s)
```

Arguments

z	a vector of doses
r	a vector of outcomes
s	a set of weights, summing to 1

Value

the inner product of s and r

Examples

```
# dose vector
dose <- c(0, 0.1, 0.4)
# outcome vector
outcome <- c(1, 1.1, 1.5)
# weight vector
weight = c(0.3, 0.4, 0.3)
theta <- extract_stochastic_intervention(z = dose, r = outcome, s = weight)
```

extract_threshold_effect

Compute difference in average outcomes above and below a dose threshold.

Description

Compute difference in average outcomes above and below a dose threshold.

Usage

```
extract_threshold_effect(z, r, threshold)
```

Arguments

z	a vector of doses
r	a vector of outcomes
threshold	a dose threshold

Value

the average of the outcomes with dose z above threshold c minus the average of the outcomes with dose z below the threshold c.

Examples

```
# dose vector
dose <- c(0, 0.1, 0.4)
# outcome vector
outcome <- c(1, 1.1, 1.5)
theta <- extract_threshold_effect(z = dose, r = outcome, threshold = 0.3)
```

`extract_threshold_effect_function` *Function factory for extract_threshold_effect.*

Description

Function factory for extract_threshold_effect.

Usage

```
extract_threshold_effect_function(threshold = 0)
```

Arguments

`threshold`	a dose threshold

Value

A function that corresponds to extract_threshold_effect with the given threshold

Examples

```
threshold_function <- extract_threshold_effect_function(threshold = 0.3)
```

`lead_bmd` *Matched lead bone mineral density dataset*

Description

A matched, trimmed dataset of lead exposure and lumbar bone mineral density. The data comes from NHANES 2011-18. There are 711 matched sets.

Usage

```
lead_bmd
```

Format

`lead_bmd`**:**

A data frame with 1,436 rows and 23 columns:

log_lead The log of lead exposure level measured in micrograms per deciliter.

lumbar_spine_bmd Bone mineral density in the lumbar spine in g/cm^2

matched_sets Matched set membership. ...

Source

https://wwwn.cdc.gov/nchs/nhanes/default.aspx

`lead_crime` *Matched lead crime dataset*

Description

A matched, trimmed dataset of early life lead exposure and juvenile delinquency from a public dataset. There are 2007 matched sets.

Usage

```
lead_crime
```

Format

`lead_crime`**:**

A data frame with 4,134 rows and 17 columns:

log_lead The log of lead exposure level measured in micrograms per deciliter.

serious Whether the juvenile comitted a serious offense.

complain Whether the juvenile comitted an offense worthy of complaint.

matched_sets Matched set membership. ...

Source

https://scholarworks.iu.edu/dspace/handle/2022/25638

`max_expectation` *A function to compute a conservative upper bound on the worst-case expectation under the sharp null*

Description

A function to compute a conservative upper bound on the worst-case expectation under the sharp null

Usage

```
max_expectation(z, gamma, f_pi, with_variance = FALSE)
```

Arguments

`z` vector of doses of length n

`gamma` The nonnegative sensitivity parameter; gamma = 0 means no unmeasured confounding.

`f_pi` a vector of length n! that contains the value of the test statistic under each of the n! permutations of z, with order of f_pi determined by first sorting z into increasing order and calling gtools::permutations on z.

`with_variance` whether to return the variance along with the worst-case expectation, default is FALSE.

Value

a list containing the worst-case expectation, and/or variance and the solution to the optimization problem.

Examples

```
# A vector of observed doses
doses <- c(0, 0.1, 0.4, 0.8)
# values of test statistic under 4! permutations
values <- c(1, 0.5, 0.3, 0.8, 1, 0.7)
upper_bound <- max_expectation(z = doses, gamma = 1, f_pi = values)
```

max_ratio *Find the max ratio of probabilities between two permutations.*

Description

Find the max ratio of probabilities between two permutations.

Usage

```
max_ratio(z, gamma)
```

Arguments

`z` vector of doses

`gamma` level of unmeasured confounding

Value

the maximum ratio between the probability of two different permutations under the Rosenbaum model with doses z and unmeasured confounding level gamma.

Examples

```
# A vector of observed doses
doses <- c(0, 0.1, 0.4, 0.8)
ratio <- max_ratio(z = doses, gamma = 1)
```

max_ratios_summary *Find the max ratio of probabilities between two permutations for each matched set.*

Description

Find the max ratio of probabilities between two permutations for each matched set.

Usage

```
max_ratios_summary(Z, index, gamma)
```

Arguments

Z	A length N vector of observed doses.
index	A length N vector of indices indicating matched set membership.
gamma	The nonnegative sensitivity parameter; gamma = 0 means no unmeasured confounding.

Value

A vector of length equaling the number of unique indices that contains the maximum ratio between any two permutations for each of the matched sets.

Examples

```
# A vector of observed doses
doses <- c(0, 0.1, 0.4, 0.8, 1)
matched_set <- c(1, 1, 1, 2, 2)
ratios <- max_ratios_summary(Z = doses, index = matched_set, gamma = 1)
```

max_ratio_new *Find the max ratio of probabilities between two permutations.*

Description

Find the max ratio of probabilities between two permutations.

Usage

```
max_ratio_new(z, gamma)
```

Arguments

z	vector of doses
gamma	The nonnegative sensitivity parameter; gamma = 0 means no unmeasured confounding.

Value

the maximum ratio between the probability of two different permutations under the Rosenbaum model with doses z and unmeasured confounding level gamma.

Examples

```
# A vector of observed doses
doses <- c(0, 0.1, 0.4, 0.8)
ratio <- max_ratio_new(z = doses, gamma = 1)
```

---

normal_test_gen *Sharp null sensitivity analysis for continuous exposures and binary outcomes using normal approximation.*

Description

Sharp null sensitivity analysis for continuous exposures and binary outcomes using normal approximation.

Usage

```
normal_test_gen(
  Z,
  Q,
  index,
  gamma,
  trans = identity,
  weights = NA,
  obsT = NULL,
  direct = "upper"
)
```

Arguments

Z	A length N vector of (nonnegative) observed doses.
Q	A length N vector of observed binary outcomes.
index	A length N vector of indices indicating matched set membership.
gamma	The nonnegative sensitivity parameter; gamma = 0 means no unmeasured confounding.
trans	The transformation of the doses to use for the test statistic. Default is the identity function.
weights	Weights to apply for the test statistic
obsT	The observed value of the test statistic; default is NULL.
direct	The direction of the test - "upper" or "lower"; default is upper.

Value

A list containing the following:

`obsT`	The observed value of the test statistic
`exp`	The worst-case expectation
`var`	The worst-case variance.
`deviate`	The normal approximation deviate.

Examples

```
# Load the data
data <- lead_crime
# Make a threshold at log(3.5) transformation function.
above = function(Z) { return(Z > log(3.5)) }
# Conduct randomization test using normal approximation.
solution <- normal_test_gen(data$log_lead, data$complain, data$matched_sets,
gamma = 0, trans = above)
```

prob_bounds *A function to find the maximum and minimum probability of a permutation.*

Description

A function to find the maximum and minimum probability of a permutation.

Usage

```
prob_bounds(z, gamma)
```

Arguments

`z`	vector of doses
`gamma`	The nonnegative sensitivity parameter; gamma = 0 means no unmeasured confounding.

Value

a list containing the maximum and minimum probability of a permutation under the Rosenbaum model with doses z and unmeasured confounding level gamma.

Examples

```
# A vector of observed doses
doses <- c(0, 0.1, 0.4, 0.8)
bounds <- prob_bounds(z = doses, gamma = 1)
```

sharp_double_statistic

Statistic based on inner product between transformations of dose and outcome.

Description

Statistic based on inner product between transformations of dose and outcome.

Usage

```
sharp_double_statistic(z, r, q1, q2)
```

Arguments

z	a vector of doses
r	a vector of outcomes
q1	a function that transforms the doses z
q2	a function that transforms the outcomes r

Value

a vector with values corresponding to the inner product of transformed by q1 permutations of z with transformed by q2 versions of r.

Examples

```
# dose vector
dose <- c(0, 0.1, 0.4)
# outcome vector
outcome <- c(1, 1.1, 1.5)
# transforms
transform1 <- function(x) x
transform2 <- function (x) x
theta <- sharp_double_statistic(z = dose, r = outcome, q1 = transform1,
q2 = transform2)
```

sharp_null_double_test

Asymptotic sharp null sensitivity analysis for a class of test statistics accommodating continuous exposures and any scalar outcome.

Description

Asymptotic sharp null sensitivity analysis for a class of test statistics accommodating continuous exposures and any scalar outcome.

Usage

```
sharp_null_double_test(
  Z,
  R,
  index,
  gamma = 0,
  q1 = NA,
  q2 = NA,
  X = NA,
  stratum_weights = rep(NA, nostratum),
  conservative_variance = TRUE,
  double_rank = TRUE
)
```

Arguments

`Z` A length N vector of observed doses.

`R` A length N vector of observed outcomes.

`index` A length N vector of indices indicating matched set membership.

`gamma` The nonnegative sensitivity parameter; gamma = 0 means no unmeasured confounding.

`q1` A transformation to apply to the doses.

`q2` A transformation to apply to the outcomes

`X` A matrix with I rows and less than I columns that contains covariate information.

`stratum_weights`
A weight vector.

`conservative_variance`
Whether to use the conservative variance or not; default is TRUE.

`double_rank` Whether to use the ranks of the transformed doses and outcomes; default is TRUE.

Value

A list containing the deviate, one-sided p-value, observed value of the test statistic in each matched set, and conservative standard deviation estimate.

Examples

```
# Load the data
data <- lead_bmd
# conduct sharp null test at gamma = 0.
result <- sharp_null_double_test(Z = data$log_lead,
R = -data$lumbar_spine_bmd, index = data$matched_sets, gamma = 0)
```

var_est *A function for variance estimation*

Description

A function for variance estimation

Usage

```
var_est(y, W, H_Q)
```

Arguments

y	a vector of length I containing the value of test statistics from each matched set
W	weight vector of length I.
H_Q	hat matrix corresponding to a matrix Q with dimension I by L.

Value

a conservative estimate of the standard deviation of the test statistic.

Examples

```
test_stat <- c(1, 2, 1.5)
weight <- rep(1, 3)
Q <- matrix(1:9, nrow = 3, ncol = 2)
hat <- Q %*% solve(t(Q) %*% Q) %*% t(Q)
```

weak_null_test *Asymptotic sensitivity analysis for weak nulls with continuous exposures.*

Description

Asymptotic sensitivity analysis for weak nulls with continuous exposures.

Usage

```
weak_null_test(
  Z,
  R,
  index,
  gamma = 0,
  theta = 0,
  X = NA,
  estimand_function = extract_OLS,
  gamma_star_vec = NULL,
  kappa_inv_vec = NULL
)
```

Arguments

`Z` A length N vector of observed doses.

`R` A length N vector of observed outcomes.

`index` A length N vector of indices indicating matched set membership.

`gamma` The nonnegative sensitivity parameter; gamma = 0 means no unmeasured confounding.

`theta` The value at which to test the weak null.

`X` A matrix with I rows and less than I columns that contains covariate information.

`estimand_function`
A function that takes in values z and r and outputs a scalar; this function governs the causal estimand to estimate

`gamma_star_vec` A vector that contains the minimum probability of a permutation for each matched set; default is NULL.

`kappa_inv_vec` A vector that contains the ratio of the maximum probability and minimum probability of a permutation for each matched set; default is NULL

Value

A list containing the deviate, one-sided p-value, observed value of the test statistic in each matched set, and conservative standard deviation estimate.

Examples

```
# Load the data
data <- lead_bmd
# prepare data
threshold <- log(0.74675)
match_info = data |> dplyr::group_by(matched_sets) |>
dplyr::summarise(below = sum(log_lead < threshold) > 0, disc = var(log_lead) > 0,
above = sum(log_lead > threshold) > 0)
below_indices <- match_info$matched_sets[match_info$below]
disc_indices <- match_info$matched_sets[match_info$disc]
above_indices <- match_info$matched_sets[match_info$above]
# outcome analysis using the stochastic intervention statistic, weak null
below_nbp <- data |> dplyr::filter(matched_sets %in% below_indices & matched_sets
%in% disc_indices)
above_below <- below_nbp |> dplyr::filter(matched_sets %in% above_indices)
extract_below_threshold_vs_baseline_function <- function(z, r) {
  extract_below_threshold_vs_baseline(z, r, threshold)
}
# one-sided test that estimand defined by estimand_function is 0 at gamma = 0
result <- weak_null_test(Z = above_below$log_lead,
R = above_below$lumbar_spine_bmd,
index = above_below$matched_sets, gamma = 0, theta = 0,
    estimand_function = extract_below_threshold_vs_baseline_function)
```

Index