Package ‘sensitivityIxJ’

October 16, 2025

Type Package

Title Exact Nonparametric Sensitivity Analysis for I by J Contingency Tables

Version 0.1.5

Description Implements exact, normally approximated, and sampling-based sensitivity analysis for observational studies with contingency tables. Includes exact (kernel-based), normal approximation, and sequential importance sampling (SIS) methods using 'Rcpp' for computational efficiency. The methods build upon the framework introduced in Rosenbaum (2002) <doi:10.1007/978-1-4757-3692-2> and the generalized design sensitivity framework developed by Chiu (2025) <doi:10.48550/arXiv.2507.17207>.

License GPL-3

Encoding UTF-8

Imports Rcpp

LinkingTo Rcpp

Suggests rbounds

RoxygenNote 7.3.3

NeedsCompilation yes

Author Elaine Chiu [aut, cre]

Maintainer Elaine Chiu <`kchiu4@wisc.edu`>

Repository CRAN

Date/Publication 2025-10-16 12:20:02 UTC

Contents

`estimate_tables` *Estimate the Total Number of Tables*

Description

This function estimates the total number of tables satisfying the margin constraints. We recommend the readers to use this before determining the number of Monte Carlo simulations for the sampling-based p-value calculation.

Usage

```
estimate_tables(treatment.margins, outcome.margins)
```

Arguments

`treatment.margins`	A vector specifying the total number of subjects receiving each treatment level. The first entry of this vector represents the number of subjects receiving the first treatment level in the study.
`outcome.margins`	A vector specifying the total number of subjects showing each outcome level. The first entry of this vector represents the number of subjects showing the first outcome level in the study.

Value

this function returns the estimated total number of tables satisfying the margin constraints.

`exact.general.sen.IxJ` *Exact Sensitivity Analysis for General Test Statistics in I by J Tables*

Description

This function computes exact p-values for sensitivity analysis in I by J contingency tables under the generic bias model. It enumerates all tables in the reference set and calculates the maximum p-value over the sensitivity parameter space (u allocations). The function is designed for general permutation-invariant test statistics.

Usage

```
exact.general.sen.IxJ(
  u_space = NULL,
  obs.table,
  table_space,
  gamma,
  delta,
  row = "treatment",
  verbose = FALSE
)
```

Arguments

`u_space`	A matrix where each row represents a unique allocation of `u_i=1` across outcomes. If `NULL`, the function generates a default set of corner allocations based on the number of outcomes. Defaults to `NULL`.
`obs.table`	A matrix or table object representing the observed contingency table.
`table_space`	A list of matrices or table objects representing the space of contingency tables to consider (typically all tables with test statistic >= observed).
`gamma`	A scalar
`delta`	A binary vector with no more than two unique values, corresponding to treatment levels. The length must match the number of treatments (rows of `obs.table` if `row = "treatment"`, or columns if `row = "outcome"`).
`row`	A string indicating whether rows represent "outcome" or "treatment". Must be either "outcome" or "treatment". Default is "treatment".
`verbose`	A logical flag indicating whether to print progress messages showing the current maximizer and probability at each step. Default is `FALSE`.

Details

The function performs exact sensitivity analysis by:

1. Enumerating all possible u allocations (or using provided u_space)
2. Computing the p-value for each allocation by summing probabilities over table_space
3. Finding the allocation that maximizes the p-value

For computational efficiency, the function only supports certain table dimensions. If u_space is not provided, default corner allocations are generated for J <= 5.

Value

A list containing:

rct.prob Probability under Randomized Controlled Trial (RCT) with `u_allocation` set to zero.

max.prob Maximum probability found across all allocations in `u_space`.

maximizer The `u_allocation` vector that yields `max.prob`.

gamma Extracted gamma value from the generic bias model.

delta remind the practitioners of their delta

Examples

```
## Example 1: 3 by 3 table with custom test statistic
# Create an observed table (example data)
obs.table <- matrix(c(5, 3, 2, 6, 11, 7, 3, 0, 3), ncol = 3, byrow = TRUE)

# Define a test statistic emphasizing certain cells
transform.fun <- function(tb){
  test.stat <- 4 * tb[3, 3] + 3 * tb[2, 3]
  return(test.stat)
}
obs.stat <- transform.fun(obs.table)

# Find the reference set (tables with test statistic >= observed)
table.set <- possible.table(
  threshold = obs.stat,
  table = obs.table,
  direction = "greater than",
  transform.fun = transform.fun
)

# Perform sensitivity analysis
sen.result <- exact.general.sen.IxJ(
  obs.table = obs.table,
  table_space = table.set,
  gamma = 0.5,
  delta = c(0, 1, 1)
)
sen.result
```

`exact.score.sen.IxJ` *Exact Sensitivity Analysis for Sum Score (Ordinal) Tests in I by J Tables*

Description

This function computes exact p-values for score-based sensitivity analysis in I by J contingency tables under the generic bias model. It is specifically designed for ordinal data where both treatment levels and outcomes have a natural ordering, and tests for trend using assigned scores.

Usage

```
exact.score.sen.IxJ(
  obs.table,
  gamma,
  delta,
  row = "treatment",
  verbose = FALSE,
```

```
  treatment.scores,
  outcome.scores
)
```

Arguments

`obs.table`	A matrix or table object representing the observed contingency table.
`gamma`	A nonnegative scalar.
`delta`	A binary vector with no more than two unique values, corresponding to treatment levels. The length must match the number of treatments (rows of `obs.table` if `row = "treatment"`, or columns if `row = "outcome"`). Must have a monotone trend matching the treatment ordering.
`row`	A string indicating whether rows represent "outcome" or "treatment". Must be either "outcome" or "treatment". Default is "treatment".
`verbose`	A logical flag indicating whether to print progress messages. Default is `FALSE`.
`treatment.scores`	A numeric vector specifying the scores for each treatment level. Must have the same length as the number of treatments and exhibit a monotone trend.
`outcome.scores`	A numeric vector specifying the scores for each outcome level. Must have the same length as the number of outcomes and exhibit a monotone trend.

Details

The score test assumes both treatments and outcomes are ordinal with monotone trends. The test statistic is computed as the sum of products of cell counts with their corresponding treatment and outcome scores. The function automatically generates an appropriate u-space for score tests, focusing on allocations that respect the ordinal nature of the data.

Value

A list containing:

rct.prob Probability under Randomized Controlled Trial (RCT) with `u_allocation` set to zero.

max.prob Maximum probability found across all allocations in `u_space`.

maximizer The `u_allocation` vector that yields `max.prob`.

gamma Extracted gamma value from the generic bias model.

delta The delta vector

obs.stat The observed test statistic based on the observed table and the assigned scores.

obs.table The observed table.

See Also

`exact.general.sen.IxJ` for general test statistics, `possible.table` for generating reference sets

Examples

```
## Example 1: Binary outcome table (2 by 2)
obs.table <- matrix(c(12, 18, 17, 3), ncol = 2, byrow = TRUE,
                   dimnames = list(treatment = c("control", "treated"),
                                   outcome = c("failure", "success")))

# Perform score-based sensitivity analysis
result_2x2 <- exact.score.sen.IxJ(obs.table = obs.table,
                                  gamma = 0.5,
                                  delta = c(0, 1),
                                  treatment.scores = c(0, 1),
                                  outcome.scores = c(0, 1))
result_2x2

## Example 2: Three-level ordinal outcome (3 by 3)
obs.table <- matrix(c(12, 18, 17, 3, 12, 25, 0, 3, 4),
                   ncol = 3, byrow = FALSE,
                   dimnames = list(treatment = c("low", "medium", "high"),
                                   outcome = c("poor", "fair", "good")))

# Test for trend with ordinal scores
result_3x3 <- exact.score.sen.IxJ(obs.table = obs.table,
                                  gamma = 0.5,
                                  delta = c(0, 1, 1),
                                  treatment.scores = c(0, 1, 2),
                                  outcome.scores = c(1, 2, 3))
result_3x3
```

`generic.I.by.J.sensitivity.point.probability`

Compute the exact Probability of a Single Table for the Generic Bias Model

Description

This function computes the probability of a single contingency table under the generic bias model given an unmeasured confounder. This is an auxiliary function for the p-value computation.

Usage

```
generic.I.by.J.sensitivity.point.probability(
  table,
  row = "treatment",
  u_allocation,
  gamma,
  delta,
  shared_divisor = 1e+06
)
```

Arguments

`table`	A matrix or table object representing the observed contingency table.
`row`	A string indicating whether rows represent "outcome" or "treatment". Must be either "outcome" or "treatment". Default is "treatment".
`u_allocation`	A vector where each entry represents the number of `u_i=1` in an outcome level. The first entry represents the number of `u_i=1` among the subjects with outcome as one.
`gamma`	A scalar
`delta`	A binary vector with no more than two unique values, corresponding to treatment levels. The length must match the number of treatments (rows of `obs.table` if `row = "treatment"`, or columns if `row = "outcome"`).
`shared_divisor`	A scalar to rescale the numerator and the denominator of the probability mass function to prevent overflow. Defaulted to 1000000.

Value

This function returns the probability mass of this table given a unmeasured confounder.

`norm.score.sen.IxJ` *Normal Approximation Sensitivity Analysis for I by J Tables*

Description

This function implements normal approximation methods for sensitivity analysis in I by J contingency tables under the generic bias model. It computes asymptotically valid p-values for score test statistics based on the product of treatment and outcome scores, providing rapid analysis for large tables.

Usage

```
norm.score.sen.IxJ(
  obs.table,
  gamma,
  delta,
  row = "treatment",
  treatment.scores,
  outcome.scores,
  shared_divisor = 1e+06,
  u_space = NULL,
  verbose = FALSE
)
```

Arguments

`obs.table`	A matrix or table object representing the observed contingency table.
`gamma`	a nonnegative scalar.
`delta`	a binary vector to treatment levels. Its length must match the number of treatments (rows of `obs.table` if `row = "treatment"`, or columns if `row = "outcome"`).
`row`	A string indicating whether rows represent "outcome" or "treatment". Must be either "outcome" or "treatment". Default is "treatment".
`treatment.scores`	A numeric vector of scores for treatments. Must be monotone (either increasing or decreasing). Higher scores typically indicate more intense treatments. Length must equal the number of treatments.
`outcome.scores`	A numeric vector of scores for outcomes. Must be monotone (either increasing or decreasing). Higher scores typically indicate better outcomes. Length must equal the number of outcomes.
`shared_divisor`	Numeric value used for numerical stability in calculations. Default is 1e6.
`u_space`	A numeric matrix where each row is a candidate `u_allocation`. If `NULL` (default), corner allocations are generated automatically for tables with J <= 5 outcomes.
`verbose`	Logical; if `TRUE`, prints progress messages including the current u-allocation and p-value at each step. Default is FALSE.

Details

For an I by J table, the test statistic is a weighted sum of cell counts where weights are products of treatment and outcome scores: $T = \sum w_i v_j N_{ij}$ across all cells.

The method computes:

- Mean and variance of the test statistic under the generic bias model
- Standardized z-scores assuming asymptotic normality
- the one-sided upper tailed probability

When `u_space` is not provided, the function automatically generates corner u allocations which often contain the worst-case scenarios for sensitivity analysis.

Value

A list containing:

T_obs The observed test statistic value.

RCT.mean Mean of the test statistic under RCT (gamma = 0) or (Gamma = 1).

max.mean Mean of the test statistic under the sensitivity model at maximizer.

RCT.var Variance of the test statistic under RCT.

max.var Variance of the test statistic under the sensitivity model at maximizer.

RCT.prob P-value under RCT (no unmeasured confounding).

max.prob Maximum p-value across all u-allocations (sensitivity bound).

maximizer The u-allocation vector that yields max.prob.

treatment.scores The treatment scores used in the analysis.

outcome.scores The outcome scores used in the analysis.

See Also

`exact.general.sen.IxJ` for exact methods, `sampling.general.sen.IxJ` for Monte Carlo methods

Examples

```
# 2 by 3 table example with ordinal scores
obs.table <- matrix(c(10, 20, 30, 15, 25, 10), nrow = 2, byrow = TRUE)
treatment.scores <- c(0, 1)    # Control vs Treatment
outcome.scores <- c(0, 1, 2)   # Ordinal outcomes

result <- norm.score.sen.IxJ(obs.table = obs.table,
                      gamma = 0.5,
                      delta = c(0, 1),
                      treatment.scores = treatment.scores,
                      outcome.scores = outcome.scores
                      )

# 3 by 3 table with customized scores
obs.table <- matrix(data=c(10,30,10,14,15,24,4,5,15), nrow = 3)
treatment.scores <- c(0, 0.5, 1)  # Three treatment levels
outcome.scores <- c(0, 1, 2)   # three outcome levels

result <- norm.score.sen.IxJ(obs.table = obs.table,
                      gamma = 0.5,
                      delta = c(0, 0, 1),
                      treatment.scores = treatment.scores,
                      outcome.scores = outcome.scores
                      )
```

`norm_single_u_allocation_p_value`

Compute the normal-approximation-based z-score and p-value for a given 2 by 2, 2 by 3, 2 by 4, 2 by 5, 3 by 2, 4 by 2, or 3 by 3 contingency table.

Description

This function performs a score test for ordinal association in contingency tables with fixed margins under a sensitivity analysis framework. It uses the test statistic $T = \sum A_{ij} N_{ij}$ where $A_{ij} = w_i v_j$ (treatment score X outcome score) and computes the mean and variance under the null hypothesis using the Poisson-binomial approximation for the multivariate Fisher's noncentral hypergeometric distribution.

Usage

```
norm_single_u_allocation_p_value(
  obs.table,
  gamma,
  delta,
  u_allocation,
  row = "treatment",
  treatment.scores,
  outcome.scores,
  shared_divisor = 1e+06
)
```

Arguments

`obs.table`	A matrix or table object representing the contingency table. Rows represent treatments, columns represent outcomes.
`gamma`	A nonnegative scalar
`delta`	A binary vector. Must have length equal to the number of treatments. Can contain at most two unique values.
`u_allocation`	A numeric vector of unmeasured confounder allocations for each outcome category. Must have length equal to the number of outcomes. Each entry must be non-negative and not exceed the corresponding outcome margin.
`row`	Character indicating whether rows represent "treatment" or "outcome". If "outcome", the table will be transposed. Default is "treatment".
`treatment.scores`	A numeric vector of scores for treatments. Must be monotone (either increasing or decreasing). Higher scores typically indicate more intense treatments.
`outcome.scores`	A numeric vector of scores for outcomes. Must be monotone (either increasing or decreasing). Higher scores typically indicate better outcomes.
`shared_divisor`	A numeric value used for numerical stability in the Rcpp computations. Default is 1e6.

Details

The function implements a score test for ordinal association where the test statistic is a weighted sum of the cell counts, with weights given by the product of treatment and outcome scores. The function uses specialized Rcpp functions to compute the mean and variance-covariance structure of the free cells (those with degrees of freedom), then recovers the full mean vector and covariance matrix using the marginal constraints. The test statistic and its moments are then computed using matrix operations.

Value

A list containing:

`T_obs`	The observed test statistic
`mu_T`	The expected value of the test statistic under the null hypothesis

`var_T`	The variance of the test statistic under the null hypothesis
`z_score`	The standardized test statistic (T_obs - mu_T) / sqrt(var_T)
`p_value`	The one sided upper tailed p-value based on the normal approximation
`treatment.scores`	The treatment scores used
`outcome.scores`	The outcome scores used

Examples

```
# 2x3 contingency table
obs.table <- matrix(c(10, 20, 30, 15, 25, 10), nrow = 2, byrow = TRUE)

# Sensitivity parameters and u_allocation
gamma = 0.5
delta <- c(0, 1)
u_allocation <- c(5, 10, 8)

# Ordinal scores
treatment.scores <- c(0, 1)
outcome.scores <- c(0, 1, 2)

# Perform test
result <- norm_single_u_allocation_p_value(
  obs.table = obs.table,
  gamma = gamma,
  delta = delta,
  u_allocation = u_allocation,
  treatment.scores = treatment.scores,
  outcome.scores = outcome.scores
)
```

`possible.table` *Generate All Possible Contingency Tables Exceeding or Not Exceeding a Threshold*

Description

Given an observed contingency table (with fixed margins) and a user-defined function that computes a test statistic, this function enumerates all valid contingency tables that meet a particular criterion (i.e., those with `T(table) >= threshold` or `T(table) <= threshold`).

Usage

```
possible.table(
  threshold,
  table,
  direction = c("greater than", "less than"),
  transform.fun
)
```

Arguments

`threshold`	A numeric value representing the cutoff for the test statistic.
`table`	A matrix or table object specifying the observed contingency table. The function will preserve the row/column sums (margins) of this table when generating possible tables.
`direction`	A character string, either `"greater than"` or `"less than"`, indicating whether to return tables whose test statistic is `>= threshold` or `<= threshold`. Defaults to `c("greater than", "less than")` but you should pass in one value explicitly.
`transform.fun`	A user-defined function of the form `f(tbl)`, which takes a contingency table (as a matrix or table) and returns a numeric test statistic.

Details

The function systematically iterates over all valid ways to fill a contingency table of dimension `I x J` such that row and column sums match the original `table`'s margins. Only those tables that satisfy `transform.fun(tbl) >= threshold` (for `direction = "greater than"`) or `transform.fun(tbl) <= threshold` (for `direction = "less than"`) are returned.

Currently, this function supports certain table dimensions (`I` up to 5 when `J=2`, etc.). If the dimension is not supported, it returns `NULL` with a warning.

Value

A list of all contingency tables (each table is returned as a `table` object) that meet the specified `threshold` criterion. If no tables match or the dimension is not supported, `NULL` is returned.

Examples

```
# Suppose we have a 3x3 table:
obs_table <- matrix(c(5, 3, 2, 6, 11, 7, 3, 0,3), ncol = 3, byrow = TRUE)
obs_table

# Define a simple test statistic function (e.g., sum of diagonal)
diag_sum <- function(tbl) sum(diag(tbl))

# Find all 3x3 tables with the same margins where the diagonal sum >= 19
result <- possible.table(threshold = 19, table = obs_table,
                       direction = "greater than", transform.fun = diag_sum)
result[[1]]        # Inspect the first matching table
```

`sampling.general.sen.IxJ` *Monte Carlo Sensitivity Analysis for General Permutation-Invariant Test Statistics in I by J Tables*

Description

This function implements a sampling-based approach to sensitivity analysis for general (permutation-invariant) test statistics in I by J contingency tables under the generic bias model. It uses Sequential Importance Sampling (SIS) from Eisinger and Chen (2017) to approximate p-values when exact enumeration is computationally infeasible.

Usage

```
sampling.general.sen.IxJ(
  obs.table,
  gamma,
  delta,
  row = "treatment",
  mc.iteration = 5000,
  transform.fun,
  verbose = FALSE,
  u_space = NULL
)
```

Arguments

`obs.table`	A matrix or table object representing the observed contingency table.
`gamma`	a nonnegative scalar
`delta`	A binary vector with no more than two unique values, corresponding to treatment levels. The length must match the number of treatments (rows if `row = "treatment"`, columns if `row = "outcome"`).
`row`	A string indicating whether rows represent "outcome" or "treatment". Must be either "outcome" or "treatment". Default is "treatment".
`mc.iteration`	Integer specifying the number of Monte Carlo iterations for each u-allocation. Higher values increase accuracy but require more computation time. Default is 5000.
`transform.fun`	A user-defined function that takes a contingency table and returns a numeric test statistic. This function should be permutation-invariant within treatment groups.
`verbose`	Logical flag indicating whether to print progress messages showing current u-allocation and estimated probabilities. Default is FALSE.
`u_space`	Optional matrix where each row represents a candidate u-allocation. If NULL, a default set of corner allocations is generated. Each row should have length equal to the number of outcomes.

Details

This function performs Monte Carlo sensitivity analysis for arbitrary test statistics that are permutation-invariant. Unlike the score test version, this function can handle any user-defined test statistic through the `transform.fun` parameter.

The algorithm:

1. Generates tables using Sequential Importance Sampling (SIS)

2. Evaluates the test statistic on each sampled table
3. Estimates p-values using importance weights
4. Searches over u-allocations to find the maximum p-value

When `u_space` is not provided, the function generates default corner allocations that explore extreme cases in the unmeasured confounder space.

Value

A list containing:

`rct.prob`	Estimated probability under RCT (all u-allocations zero)
`max.prob`	Maximum estimated probability across all u-allocations
`maximizer`	The u-allocation vector yielding max.prob
`obs.stat`	Observed test statistic value from transform.fun
`obs.table`	The input observed table
`gamma`	Extracted gamma value from the generic bias model
`delta`	The delta vector

References

Eisinger, R. D., & Chen, Y. (2017). Sampling for Conditional Inference on Contingency Tables. Journal of Computational and Graphical Statistics, 26(1), 79–87.

See Also

`exact.general.sen.IxJ` for exact computation when feasible, `sampling.score.sen.IxJ` for score tests with ordinal data,

Examples

```
# Example with custom test statistic emphasizing corner cells
obs.table <- matrix(c(10, 5, 8, 12), ncol = 2, byrow = TRUE)

# Define test statistic: sum of diagonal elements
diag_stat <- function(tb) {
  sum(diag(tb))
}

# Run sensitivity analysis
result <- sampling.general.sen.IxJ(
  obs.table = obs.table,
  gamma = 0.5,
  delta = c(0, 1),
  transform.fun = diag_stat,
  mc.iteration = 5000,
  verbose = TRUE
)
```

```
# Custom u-space example
custom_u <- matrix(c(0, 0,
                    5, 0,
                    0, 8,
                    5, 8), ncol = 2, byrow = TRUE)

result_custom <- sampling.general.sen.IxJ(
  obs.table = obs.table,
  gamma = 0.5,
  delta = c(0, 1),
  transform.fun = diag_stat,
  u_space = custom_u
)
```

`sampling.score.sen.IxJ` *Monte Carlo Score Test Sensitivity Analysis for I by J Tables*

Description

This function implements a sampling-based approach to sensitivity analysis for score tests in I by J contingency tables under the generic bias model. It uses Sequential Importance Sampling (SIS) from Eisinger and Chen (2017) to approximate p-values when exact computation is infeasible due to large reference set

Usage

```
sampling.score.sen.IxJ(
  obs.table,
  gamma,
  delta,
  row = "treatment",
  mc.iteration = 5000,
  treatment.scores,
  outcome.scores,
  verbose = FALSE
)
```

Arguments

`obs.table`	A matrix or table object representing the observed contingency table.
`gamma`	a nonnegative scalar
`delta`	A binary vector with no more than two unique values, corresponding to treatment levels. The length must match the number of treatments. Must have a monotone trend for ordinal tests.

`row`	A string indicating whether rows represent "outcome" or "treatment". Must be either "outcome" or "treatment". Default is "treatment".
`mc.iteration`	Integer specifying the number of Monte Carlo iterations for each u-allocation. Higher values increase accuracy but require more computation time. Default is 5000.
`treatment.scores`	A numeric vector specifying scores for each treatment level. Must have the same length as the number of treatments and exhibit a monotone trend.
`outcome.scores`	A numeric vector specifying scores for each outcome level. Must have the same length as the number of outcomes and exhibit a monotone trend.
`verbose`	Logical flag indicating whether to print progress messages during computation. Default is FALSE.

Details

The function uses importance sampling to estimate p-values for score tests when both treatments and outcomes are ordinal. The score test statistic is computed as the sum of products of cell counts with their corresponding treatment and outcome scores.

The u-space is automatically constructed to respect the ordinal nature of the data, focusing on allocations that assign bias to higher outcome levels first (assuming an increasing trend in outcome scores).

Unlike exact methods, this function provides Monte Carlo estimates that converge to true values as mc.iteration increases. Results may vary slightly between runs unless the random seed is fixed.

Value

A list containing:

`rct.prob`	Estimated probability under RCT (all u-allocations zero)
`max.prob`	Maximum estimated probability across all u-allocations
`maximizer`	The u-allocation vector yielding max.prob
`obs.stat`	Observed test statistic value
`obs.table`	The input observed table
`gamma`	Extracted gamma value from the generic bias model
`delta`	delta vector

References

Eisinger, R. D., & Chen, Y. (2017). Sampling for Conditional Inference on Contingency Tables. Journal of Computational and Graphical Statistics, 26(1), 79–87.

See Also

`exact.score.sen.IxJ` for exact computation when feasible, `sampling.general.sen.IxJ` for general test statistics.

Examples

```
# Binary outcome example
obs.table <- matrix(c(15, 10, 5, 20), ncol = 2, byrow = TRUE)
result <- sampling.score.sen.IxJ(obs.table = obs.table,
                                 gamma = 0.5,
                                 delta = c(0, 1),
                                 treatment.scores = c(0, 1),
                                 outcome.scores = c(0, 1),
                                 mc.iteration = 5000)
```

Index