Package: DPComb (via r-universe)

June 24, 2026

Title Discrete p-Value Combination Tests

Version 1.0

Depends R (>= 3.5.0)

Description Provides tools for performing p-value combination tests with discrete input p-values. These tests combine significance evidence derived from independent discrete statistics to test a global null hypothesis, which is defined by the specified null distribution(s) of these discrete statistics. The testing procedure involves two main steps: (1) Wasserstein Adjustment: Each component of the combination statistic is replaced by an adjusted Z statistic. This adjustment, based on the minimum Wasserstein distance, preserves the discrete nature of the original statistics while better aligning them with their counterparts under continuity. (2) Calculation of the Significance of the Combination Statistic: A continuous distribution that optimally matches the discrete distribution of the combination statistic is obtained, and the testing p-value for the global null hypothesis is computed. The first step is analogous to Lancaster's approach but is generalized based on Wasserstein optimization. The second step allows for asymptotic control of Type I error with higher statistical power. The package implements several p-value combination methods, including Fisher's, Pearson's, George's, Stouffer's, and Edgington's methods. The individual tests to be combined can be right-sided, left-sided, or two-sided, and can be based on binomial, Poisson, hypergeometric, noncentral hypergeometric, negative binomial, or geometric distributions, or a mixture of them. The underlying methodology and its foundations are described in the following references: Contador, Gonzalo and Wu, Zheyang (2025). A minimum Wasserstein distance approach to Fisher's combination of independent, discrete p-values. Scandinavian Journal of Statistics, 52(3), 1281-1300. <doi:10.1111/sjos.12787> Contador, Gonzalo and Wu, Zheyang (2026). Optimal Adjustment and Combination of Independent Discrete p-Values. Under revision at the Journal of

Computational and Graphical Statistics. <doi:10.48550/arXiv.2508.02647> Lancaster, HO (1949). The combination of probabilities arising from data in discrete distributions. Biometrika, 36(3/4), 370-382. <doi:10.1093/biomet/36.3-4.370>.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Imports MCMCpack

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Repository https://gonzalocontador.r-universe.dev

Date/Publication 2026-06-15 19:39:12 UTC

RemoteUrl https://github.com/gonzalocontador/dpcomb

RemoteRef HEAD

RemoteSha e4a4003d4d11b579561379fd878de2cd66c44278

Contents

accuracy_metrics 2
adjZ_moments 4
case_control 5
computeZ 6
computeZmoment 7
convert_x_to_p 8
distn_to_x_support_probs 10
DPComb_tests 11
test_case_control_fisher 15

Index **17**

`accuracy_metrics` *Type I Error Accuracy Metrics for Method Selection*

Description

Computes the two metrics recommended for comparing and selecting discrete p-value combination methods: the variance ratio $\mathrm{Var}(Z)/\mathrm{Var}(Y)$ and the normalized (scaled) Wasserstein distance $W_2(Z, \tilde{Y})/\mathrm{SD}(Y)$. Here Z is the adjusted discrete statistic, Y is the continuous reference null, and $\tilde{Y}$ is the moment-matched continuous surrogate used in the testing procedure. A higher variance ratio (closer to 1) and a smaller normalized distance both indicate more accurate finite-sample Type I error control.

Usage

```
accuracy_metrics(p_support, method = "fisher_mean")
```

Arguments

p_support	A valid p-value support vector: nonnegative, nondecreasing, and ending at 1, characterizing the null distribution of the discrete p-value.
method	The combination method, one of "fisher_mean" (default), "fisher_median", "pearson", "george", "stouffer", or "edgington".

Details

The surrogate $\tilde{Y}$ matches the first two moments of Z: a gamma distribution for Fisher's and Pearson's methods, and a normal distribution for George's, Stouffer's, and Edgington's methods. The Wasserstein distance is computed under the optimal (quantile) coupling,

$$W_2^2(Z, \tilde{Y}) = \sum_i \int_{\tilde{G}^{-1}(P(Z<z_i))}^{\tilde{G}^{-1}(P(Z\le z_i))} (z_i - y)^2 \tilde{g}(y)\, dy,$$

where $\tilde{G}$ and $\tilde{g}$ are the CDF and density of $\tilde{Y}$. The integral over each atom is evaluated numerically. For the non-i.i.d. case, the average variance ratio $\sum_j \mathrm{Var}(Z_j)/(n\,\mathrm{Var}(Y))$ can be obtained by averaging `var_ratio` across the component supports.

Value

A list with the following elements:

var_Z	The variance of the adjusted statistic Z.
var_Y	The variance of the continuous reference null Y, a known constant per method: 4 for Fisher and Pearson, $\pi^2/3$ for George, 1 for Stouffer, and 1/12 for Edgington.
var_ratio	The variance ratio $\mathrm{Var}(Z)/\mathrm{Var}(Y)$, which lies in $[0, 1]$.
W2	The Wasserstein-2 distance $W_2(Z, \tilde{Y})$ between Z and the moment-matched surrogate $\tilde{Y}$.
norm_dist	The normalized distance $W_2(Z, \tilde{Y})/\mathrm{SD}(Y)$.

References

Contador, Gonzalo and Wu, Zheyang (2026). Optimal Adjustment and Combination of Independent Discrete p-Values. Under revision at the Journal of Computational and Graphical Statistics.

Examples

```
# Distribution P_L (large mass at small p-values), cf. Table 4 of Contador and Wu (2026).
p_support <- c(0.40, (41:100) / 100)
methods <- c("fisher_mean", "pearson", "stouffer", "edgington", "george")
round(sapply(methods, function(m) unlist(accuracy_metrics(p_support, m))), 3)
```

adjZ_moments *Calculate Adjusted Z Statistic and Its Moments*

Description

Calculates the adjusted Z statistic and its mean and variance corresponding to an observed p-value and the distribution of the p-value (which is fully characterized by its support vector).

Usage

```
adjZ_moments(p, p_support, method = "fisher_mean")
```

Arguments

p	An observed p-value.
p_support	A valid p-value support vector containing increasingly sorted possible p-values.
method	The p-value combination method, one of "fisher_mean" (default), "fisher_median", "pearson", "edgington", "stouffer", or "george".

Details

The input p should be in p_support. If not, the p_support element closest to p will be used to calculate Z, with a warning message. Zmean and Zvar are calculated based on p_support. The adjustment is made based on a specific p-value combination method.

Value

A list with the following elements:

Z	The adjusted Z statistic corresponding to the observed p-value.
Zmean	The mean of the adjusted Z statistic.
Zvar	The variance of the adjusted Z statistic.

Examples

```
# Example usage:
methods <- c("fisher_mean", "fisher_median", "pearson", "george", "stouffer", "edgington")
p_support <- seq(0.01, 1, length.out = 100)
sapply(methods, function(m) adjZ_moments(p_support[10], p_support, m))
sapply(methods, function(m) adjZ_moments(0.105, p_support, m)) # Warning if p is not in p_support.
```

`case_control` *Case-Control Study Data*

Description

A case-control study dataset containing genetic marker mutation indicators as covariates.

Usage

```
data(case_control)
```

Format

A data frame with 2000 rows and 16 variables:

disease_status Disease status (1 = case, 0 = control).

marker1 Binary mutation indicator for marker 1.

marker2 Binary mutation indicator for marker 2.

marker3 Binary mutation indicator for marker 3.

marker4 Binary mutation indicator for marker 4.

marker5 Binary mutation indicator for marker 5.

marker6 Binary mutation indicator for marker 6.

marker7 Binary mutation indicator for marker 7.

marker8 Binary mutation indicator for marker 8.

marker9 Binary mutation indicator for marker 9.

marker10 Binary mutation indicator for marker 10.

marker11 Binary mutation indicator for marker 11.

marker12 Binary mutation indicator for marker 12.

marker13 Binary mutation indicator for marker 13.

marker14 Binary mutation indicator for marker 14.

marker15 Binary mutation indicator for marker 15.

Details

The dataset contains:

- 1000 cases and 1000 controls
- 15 binary genetic markers. Binary mutation indicator: 1=mutation, 0=wild type.
- Number of mutations for each marker:
  - marker1: Total 19 (Cases 13)
  - marker2: Total 16 (Cases 11)
  - marker3: Total 16 (Cases 11)

- marker4: Total 10 (Cases 7)
- marker5: Total 13 (Cases 9)
- marker6: Total 12 (Cases 8)
- marker7: Total 10 (Cases 7)
- marker8: Total 12 (Cases 8)
- marker9: Total 11 (Cases 8)
- marker10: Total 16 (Cases 11)
- marker11: Total 19 (Cases 10)
- marker12: Total 9 (Cases 3)
- marker13: Total 14 (Cases 6)
- marker14: Total 8 (Cases 5)
- marker15: Total 7 (Cases 4)

Examples

```
data(case_control)
head(case_control, 3)

#1000 cases and 1000 controls
table(case_control$disease_status)

#Marker 1 has 13 mutations in cases and 6 mutations in controls.
table(case_control$disease_status, case_control$marker1)
```

`computeZ` *Compute Adjusted Z Statistic*

Description

This core function computes the combination method-related adjusted Z statistic for a given observed discrete p-value.

Usage

```
computeZ(f, f_prev, method)
```

Arguments

`f`	A given legitimate discrete p-value. It should be > 0 and <= 1.
`f_prev`	The next smaller p-value (i.e., the previous element in the p-value support vector). It is zero if `f` is the smallest possible p-value in its support.
`method`	The combination method that the adjusted Z statistic is related to. One of "fisher_mean", "fisher_median", "pearson", "george", "stouffer", or "edgington".

Details

The following are the formulas for the adjusted Z statistic when the p-value $P = F_i$. Notations: $\overline{F_i} \equiv 1 - F_i$; $K_i \equiv (2\pi)^{-1/2} \exp[-\Phi^{-1}(F_i)^2/2]$, where Φ is the cumulative distribution function of the standard normal distribution.

Method	Statistic	Value when $P = F_i$
Fisher	Z_F	$2 - 2(F_i - F_{i-1})^{-1}(F_i \log F_i - F_{i-1} \log F_{i-1})$
Pearson	Z_P	$2 - 2(F_i - F_{i-1})^{-1}(\overline{F_{i-1}} \log \overline{F_{i-1}} - \overline{F_i} \log \overline{F_i})$
George	Z_G	$(Z_P - Z_F)/2$
Stouffer	Z_S	$(F_i - F_{i-1})^{-1} [K_{i-1} - K_i]$
Edgington	Z_E	$(F_i + F_{i-1})/2$

Value

The adjusted Z statistic.

Examples

```
methods = c("fisher_mean", "fisher_median", "pearson", "george", "stouffer", "edgington")
sapply(methods, function(m) computeZ(0.1, 0.05, m))
sapply(methods, function(m) computeZ(0.1, 0, m)) # f_prev = 0
sapply(methods, function(m) computeZ(1, 0.9, m)) # f = 1
sapply(methods, function(m) computeZ(0.1, 0.1 - 1e-10, m)) # f - f_prev is small
```

`computeZmoment` *Compute Moments of Adjusted Z Statistic*

Description

This function calculates the moments (mean and variance) of the adjusted Z statistic based on a given p-value support vector and a specified combination method.

Usage

```
computeZmoment(p_support, method = "fisher_mean")
```

Arguments

`p_support`	A numeric vector of p-values. Its elements must be nonnegative, nondecreasing, and not larger than 1.
`method`	The combination method that the adjusted Z statistic is related to. One of "fisher_mean" (default), "fisher_median", "pearson", "george", "stouffer", or "edgington".

Details

This function relies on the computeZ function to calculate the adjusted Z statistic for each element in p_support. The mean and variance are then computed based on these Z statistic values and their corresponding probabilities.

Value

A list containing the mean (Zmean) and variance (Zvar) of the adjusted Z statistic.

Examples

```
methods <- c("fisher_mean", "fisher_median", "pearson", "george", "stouffer", "edgington")

# Toy example
p_support <- c(0.1, 0.2, 0.5, 1)
sapply(methods, function(m) computeZmoment(p_support, m))

# Example for p_support containing many 0's and 1's due to numerical limitations.
p_support <- pbinom(0:100000, size = 100000, prob = 0.7)
sapply(methods, function(m) computeZmoment(p_support, m))
```

---

convert_x_to_p *Convert X Distribution to the p-Value Distribution*

Description

This function converts a discrete X statistic and its distribution (characterized by its support vector and probability mass vector) to the p-value and its distribution (characterized by the p-value support, with increasingly sorted unique possible elements).

Usage

```
convert_x_to_p(x = NULL, x_support, x_prob, side = "two")
```

Arguments

x	An observed X value. Default is NULL (for the purpose of getting the p_support only).
x_support	Vector of support of X, i.e., all possible unique X values sorted increasingly.
x_prob	Vector of probability masses, matching with elements of x_support.
side	Side of p-value. One of "two" (default), "left", or "right", specifying the tail for the p-value computation.

Details

Let X be a random discrete variable. The p-values are defined as:

For left-sided:

$$p(x) = P(X \leq x)$$

For right-sided:

$$p(x) = P(X \geq x)$$

For two-sided:

$$p(x) = \sum_{y:P(X \leq y) \leq P(X \leq x)} P(X = y)$$

By definition, `p_support` does not contain 0 and always contains 1. However, the `p_support` may contain repeated 0's and 1's due to numerical limitations. The package has tried to account for this issue.

Value

A list with elements:

`p`	The observed p-value corresponding to the input `x`. It is NULL if `x` is NULL.
`p_support`	Unique sorted vector of p-value support.
`p_by_x`	Vector of possible p-values matching the X support. This is not used in the main function, but can be useful for debugging.

Examples

```
x_support <- 0:5
x_prob <- dbinom(x_support, size = 5, prob = 0.1)
convert_x_to_p(3, x_support, x_prob, side = "left")
convert_x_to_p(3, x_support, x_prob, side = "right")
convert_x_to_p(3, x_support, x_prob, side = "two")
convert_x_to_p(NULL, x_support, x_prob, side = "two") # x can be NULL: output p is NULL

# x must be in x_support if not NULL. Error message otherwise.
# convert_x_to_p(6, x_support, x_prob, side = "two") #This gives Error as expected.

# symmetric distribution leads to duplicated values in p_by_x and a shorter p_support.
x_prob <- dbinom(x_support, size = 5, prob = 0.5)
convert_x_to_p(3, x_support, x_prob, side = "two")

# For large distributions, the p_support is truncated for numerical stability.
# It is still valid for practical hypothesis testing purposes.
n = 100000
x_support = 0:n
x_prob = dbinom(x_support, size = n, prob = 0.7)
result = convert_x_to_p(3, x_support, x_prob, side = "left")
result$p
length(result$p_support) # truncated
```

`distn_to_x_support_probs` *Get Support and Probability Vectors from a Distribution Description*

Description

Generates the support vector (i.e., increasingly sorted unique possible values) and the corresponding probability mass vector for a given discrete distribution description, including the name of the distribution and its parameters. Allows for replication.

Usage

```
distn_to_x_support_probs(distn_params, repN = 1)
```

Arguments

`distn_params`	A list containing: 1) `distn`: the distribution name. One of "binom", "pois", "hyper", "noncenhypergeom", "nbinom", or "geom". 2) Parameters for the specified distribution. They are consistent with the corresponding R density functions. See details.
`repN`	The number of support/probability vectors to generate. Default is 1.

Details

Supported distributions and their parameters:

- Binomial: list(distn = "binom", size = , prob =).
- Poisson: list(distn = "pois", lambda =).
- Hypergeometric: list(distn = "hyper", m = , n = , k =).
- Noncentral Hypergeometric: list(distn = "noncenhypergeom", n1 = , n2 = , m1 = , psi =). Requires the `MCMCpack` package.
- Negative Binomial: list(distn = "nbinom", size = , prob =).
- Geometric: list(distn = "geom", prob =).

Value

A list of `repN` replicated lists, each containing the `x_support` and `x_prob` vectors.

Examples

```
# For binomial distribution
distn_params <- list(distn = "binom", size = 5, prob = 0.1)
distn_to_x_support_probs(distn_params)
distn_to_x_support_probs(distn_params, 2) # replicate 2 times

# For Poisson distribution
distn_params <- list(distn = "pois", lambda = 100)
```

```
distn_to_x_support_probs(distn_params)

# For hypergeometric distribution
distn_params <- list(distn = "hyper", m = 10, n = 5, k = 3)
distn_to_x_support_probs(distn_params)

# For noncentral hypergeometric distribution
distn_params <- list(distn = "noncenhypergeom", n1 = 10, n2 = 5, m1 = 3, psi = 1)
distn_to_x_support_probs(distn_params) # same as the hypergeometric distribution when psi = 1
distn_params <- list(distn = "noncenhypergeom", n1 = 10, n2 = 5, m1 = 3, psi = 0.5)
distn_to_x_support_probs(distn_params) # smaller psi makes higher prob for smaller values
distn_params <- list(distn = "noncenhypergeom", n1 = 10, n2 = 5, m1 = 3, psi = 2)
distn_to_x_support_probs(distn_params) # larger psi makes higher prob for larger values

# For negative binomial distribution
distn_params <- list(distn = "nbinom", size = 5, prob = 0.1)
distn_to_x_support_probs(distn_params)

# For geometric distribution
distn_params <- list(distn = "geom", prob = 0.1)
distn_to_x_support_probs(distn_params)
```

DPComb_tests *Discrete P-value Combination Tests*

Description

This function combines evidence of significance to test a global null hypothesis that the given discrete null distributions are true. The tests can be based on discrete p-values or the corresponding discrete X statistics, and their null distributions.

Usage

```
DPComb_tests(
  ps = NULL,
  p_supports = NULL,
  xs = NULL,
  side = "two",
  x_support_probs = NULL,
  x_distn_params = NULL,
  method = "fisher_mean"
)
```

Arguments

ps	Optional. A vector of observed discrete p-values to be combined.
p_supports	Optional. A vector or list of p-value supports, characterizing the null distribution of discrete p-values. A vector can be used when all p-values are from the same distribution under the null, i.e., the i.i.d. case. A list allows for non-identical

distributions, containing p-value support vectors matching the distributions of the elements in ps.

xs — Optional. A vector of observed discrete X statistics, from which the p-values in ps are obtained. It is not used if ps is given, but required if ps is NULL.

side — sidedness of p-values to be combined. One of "two" (default), "right" or "left".

x_support_probs — Optional. A list containing x_support and x_prob vectors to characterize the null distributions of X statistics under the null distribution. For the i.i.d. case, where each element in xs follows the same distribution, it is a list of two vectors x_support and x_prob. For non-identical distributions, x_support_probs is required to be a list of lists, each sublist containing the x_support and x_prob vectors to define the null distributions of X, corresponding to the elements of xs. The number of sublists should equal the length of xs or ps.

x_distn_params — Optional. A list containing the description of null distributions for the discrete X statistics. See the details. For the i.i.d. case, it is a simple list. For non-identical distributions, x_distn_params is required to be a list of lists, each sublist defining the null distributions of one X statistic, consistent with xs or ps. The number of sublists should equal the length of xs.

method — The combination testing method. One of "fisher_mean" (default), "fisher_median", "pearson", "edgington", "stouffer", or "george". See the details.

Details

This function calculates the following types of p-value combination statistics Sn and their testing p-values pval.

Method	Statistic
Fisher	$T_F = -2\sum_{j=1}^{n} \log P_j$
Pearson	$T_P = -2\sum_{j=1}^{n} \log(1-P_j)$
George	$T_G = \frac{T_P - T_F}{2} = \sum_{j=1}^{n} \log \frac{P_j}{1-P_j}$
Stouffer	$T_S = \sum_{j=1}^{n} \Phi^{-1}(P_j)$
Edgington	$T_E = \sum_{j=1}^{n} P_j$

Smaller p-values represent higher significance against the null hypothesis. By its formula, a larger Sn in Fisher's combination method, or a smaller Sn in other combination methods, indicates a higher significance level against the null hypothesis. These statistics are adjusted by an "adjusted Z statistic" obtained by a Wasserstein distance optimization process. For Fisher's combination, the two methods "fisher_mean" and "fisher_median" correspond to Lancaster's mean-value chi-squared method and median-value chi-squared method, respectively.

The supported distribution descriptions and parameters:

- Binomial: list(distn = "binom", size = , prob =).
- Poisson: list(distn = "pois", lambda =).
- Hypergeometric: list(distn = "hyper", m = , n = , k =).

- Noncentral Hypergeometric: list(distn = "noncenhypergeom", n1 = , n2 = , m1 = , psi =). Requires the MCMCpack package.
- Negative Binomial: list(distn = "nbinom", size = , prob =).
- Geometric: list(distn = "geom", prob =).

Value

A list with elements:

Sn	The combination statistic
pval	The p-value of the combination statistic

References

Lancaster, HO (1949). The combination of probabilities arising from data in discrete distributions. Biometrika, 36(3/4), 370-382.

Contador, Gonzalo and Wu, Zheyang (2025). A minimum Wasserstein distance approach to Fisher's combination of independent, discrete p-values. Scandinavian Journal of Statistics, 52(3), 1281-1300.

Contador, Gonzalo and Wu, Zheyang (2026). Optimal Adjustment and Combination of Independent Discrete p-Values. Under revision at the Journal of Computational and Graphical Statistics.

Examples

```
# Example 1: p-values from the same distribution
p_supports <- seq(0.01, 1, length.out = 100)
methods <- c("fisher_mean", "fisher_median", "pearson", "george", "stouffer", "edgington")

ps <- c(0.1, 0.2, 0.21, 0.35)
sapply(methods, function(m) DPComb_tests(ps=ps, p_supports=p_supports, method=m))

# Example 2: p-values from different distributions
p_supports1 <- seq(0.01, 1, length.out = 10)
p_supports2 <- seq(0.3, 1, length.out = 5)
p_supports3 <- seq(0.01, 1, length.out = 100)
p_supports <- list(p_supports1, p_supports2, p_supports3)
ps <- c(0.12, 0.475, 0.21)
sapply(methods, function(m) DPComb_tests(ps=ps, p_supports=p_supports, method=m))

# Example 3: input xs and x_support_probs from the same distribution
xs <- c(0, 1, 2)
x_support_probs <- list(x_support = 0:5, x_prob = dbinom(0:5, size = 5, prob = 0.1))
sapply(methods, function(m) DPComb_tests(xs=xs, x_support_probs=x_support_probs,
                                         side="two", method=m))

# Example 4: input xs and x_support_probs from different distributions
xs <- c(0, 1, 2)
x_supports <- list(0:5, 0:5, 0:5)
x_probs <- list(dbinom(0:5, size = 5, prob = 0.1), dbinom(0:5, size = 5, prob = 0.2),
                dbinom(0:5, size = 5, prob = 0.3))
```

```
x_support_probs <- lapply(1:length(x_supports),
                          function(i) list(x_support = x_supports[[i]],
                                           x_prob = x_probs[[i]]))
sapply(methods, function(m) DPComb_tests(xs=xs, x_support_probs=x_support_probs,
                                         side="two", method=m))
sapply(methods, function(m) DPComb_tests(xs=xs, x_support_probs=x_support_probs,
                                         side="right", method=m))

# Example 5: input xs and x_distn_params from the same distribution
xs <- c(0, 1, 2)
x_distn_params <- list(distn = "binom", size = 5, prob = 0.1)
sapply(methods, function(m) DPComb_tests(xs=xs, x_distn_params=x_distn_params,
                                         side="two", method=m))

# Example 6: input xs and x_distn_params from different distributions
xs <- c(0, 1, 2)
x_distn_params <- list(list(distn = "binom", size = 5, prob = 0.1),
                           list(distn = "binom", size = 5, prob = 0.2),
                           list(distn = "binom", size = 5, prob = 0.3))
sapply(methods, function(m) DPComb_tests(xs=xs, x_distn_params=x_distn_params,
                                         side="two", method=m))
sapply(methods, function(m) DPComb_tests(xs=xs, x_distn_params=x_distn_params,
                                         side="right", method=m))

# Example 7: input xs and x_distn_params from different types of distributions:
# binomial, poisson, and hypergeometric
xs <- c(0, 1, 2)
x_distn_params <- list(list(distn = "binom", size = 5, prob = 0.1),
                       list(distn = "pois", lambda = 5),
                       list(distn = "hyper", m = 10, n = 5, k = 3))
sapply(methods, function(m) DPComb_tests(xs=xs, x_distn_params=x_distn_params,
                                         side="two", method=m))
# Example 8: input ps and x_support_probs.
# Require ps and x_support_probs are consistent, i.e., ps are in the the p_supports
# generated from x_support_probs. Avoid using this if unsure.
ps <- c(0.00001, 0.00032, 0.00243)
x_support_probs <- list(list(x_support = 0:5, x_prob = dbinom(0:5, size = 5, prob = 0.1)),
            list(x_support = 0:5, x_prob = dbinom(0:5, size = 5, prob = 0.2)),
            list(x_support = 0:5, x_prob = dbinom(0:5, size = 5, prob = 0.3)))
sapply(methods, function(m) DPComb_tests(ps=ps, x_support_probs=x_support_probs, method=m))
sapply(methods, function(m) DPComb_tests(ps=ps, x_support_probs=x_support_probs,
                                         side="right", method=m))
# Example 9: input ps and x_distn_params
# Require ps and x_distn_params are consistent, i.e., ps are in the the p_supports
# generated from x_distn_params. Avoid using this if unsure.
ps <- c(0.00001, 0.00032, 0.00243)
x_distn_params <- list(list(distn = "binom", size = 5, prob = 0.1),
                       list(distn = "binom", size = 5, prob = 0.2),
                       list(distn = "binom", size = 5, prob = 0.3))
sapply(methods, function(m) DPComb_tests(ps=ps, x_distn_params=x_distn_params, method=m))
```

`test_case_control_fisher` *Combine Fisher's Exact Tests in Case-Control Analysis*

Description

This function performs a combination test on case-control data with binary covariates by first obtaining p-values from Fisher's exact test for each covariate and then combining them using the `DPComb_tests` function.

Usage

```
test_case_control_fisher(
  Data,
  response,
  covariates,
  method = "fisher_mean",
  alternative = "two.sided"
)
```

Arguments

`Data`	A dataframe containing case-control data. It must include a column with 1 for cases and 0 for controls, and one or more columns representing binary covariates (with values 1 or 0).
`response`	A character string specifying the name of the response variable indicating cases and controls.
`covariates`	A vector of character strings specifying the names of the covariate columns to analyze.
`method`	A character string specifying the combination method to be used. Default is `"fisher_mean"`. Supported methods include `"fisher_mean"`, `"fisher_median"`, `"pearson"`, `"george"`, `"stouffer"`, and `"edgington"`.
`alternative`	A character string indicating the tail for Fisher's exact test. One of "two.sided" (default), "greater", or "less", consistent with the `fisher.test` function.

Details

For each covariate, Fisher's exact test is performed to compute a p-value, based on the hypergeometric distribution under the null hypothesis. The parameters are derived from the total number of cases, controls, and the total count of 1's in each covariate. The `DPComb_tests` function is then applied to compute the test statistic and associated p-value for the combination of Fisher's exact tests.

Value

A list with the following elements:

Sn The testing statistic combining statistical significance from Fisher's exact tests.

pval The testing p-value for the combination test.

Examples

```
# Load case-control data from DPComb
data(case_control, package = "DPComb")
covariates <- c("marker1", "marker2", "marker3", "marker4", "marker5")
test_case_control_fisher(Data = case_control, response = "disease_status",
                         covariates = covariates,
                         method = "fisher_mean", alternative = "two.sided")
```

Index

∗ **datasets**
case_control, 5

accuracy_metrics, 2
adjZ_moments, 4

case_control, 5
computeZ, 6
computeZmoment, 7
convert_x_to_p, 8

distn_to_x_support_probs, 10
DPComb_tests, 11

test_case_control_fisher, 15