Package 'CAFT'

June 25, 2026

Type Package

Title Rank-Based Compositional Analysis using Log-Linear Models for Microbiome Data with Zero Cells

Version 0.1.0

Date 2026-06-09

Description Provides rank-based compositional differential abundance analysis for microbiome count data with zero cells using the compositional accelerated failure time model of Satten, Li and Zhao (2025) ``CAFT: A Compositional Log-Linear Model for Microbiome Data with Zero Cells'' <doi:10.1101/2025.11.26.690468>. Zero counts are treated as censored observations below sample-specific detection limits, avoiding the use of pseudo-counts. The package implements estimation and hypothesis testing procedures for assessing associations between microbial taxa and covariates while accounting for the compositional structure of sequencing count data. It supports taxon-level differential abundance analysis, estimation of regression effects under censoring induced by detection limits, and inference based on rank-based methods that remain applicable in the presence of excess zeros. Functions are provided for model fitting, significance testing, extraction of effect estimates, and summarization of results across taxa. The package also provides optional bootstrap calibration of taxon-level p-values for sensitivity analysis in small-taxon settings.

Imports doParallel, doRNG, expm, foreach, ggplot2, ggtext, ICSNP, MASS, stats

License LGPL (>= 2)

Depends R (>= 4.3.0)

URL https://github.com/mli171/CAFT

BugReports https://github.com/mli171/CAFT/issues

Suggests graphics, knitr, rmarkdown, testthat (>= 3.0.0)

Encoding UTF-8

LazyData true

LazyDataCompression bzip2

VignetteBuilder knitr

Config/testthat/edition 3

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Glen A. Satten [aut, ctb, cph],
Mo Li [aut, ctb, cre, cph],
Ni Zhao [aut, ctb, cph]

Maintainer Mo Li <mo.li@louisiana.edu>

Repository CRAN

Date/Publication 2026-06-25 11:20:14 UTC

Contents

`caft` *Rank-Based Compositional Analysis using Log-Linear Models for Microbiome Data with Zero Cells*

Description

This is a statistical framework for differential abundance analysis of microbiome data, termed the Compositional Accelerated Failure Time (CAFT) model. The CAFT model addresses zero read counts by treating them as censored observations below the detection limit, similar to censoring mechanisms employed in survival analysis. This approach is inherently resistant to multiplicative bias, eliminates the need for pseudocounts, and addresses compositional bias through the establishment of appropriate score test procedures.

In addition to the default asymptotic CAFT inference, the function optionally provides bootstrap calibration of taxon-level p-values. Bootstrap calibration is mainly intended as a sensitivity analysis or preferred calibration method when the number of taxa is small in the data set.

Usage

```
caft(
  otu.table,
  x.test = NULL,
  x.adj = NULL,
```

```
  x = NULL,
  Gamma = NULL,
  b = NULL,
  filter.thresh = 0.05,
  fdr.nominal = 0.2,
  adjust.method = "BH",
  regularize = TRUE,
  test.method = "rank",
  n.cores = 1L,
  boot.B = 0L,
  boot.parallel = FALSE,
  boot.n.cores = 1L,
  boot.seed = NULL,
  boot.return.dist = FALSE,
  verbose = FALSE,
  return.mr.resid = FALSE
)
```

Arguments

`otu.table`	The community OTU table (or taxa count table). Each row corresponds to a sample and each column corresponds to one OTU (taxa).
`x.test`	Covariate(s) of interest. This can be a vector, matrix, or data frame. Multi-level categorical variables should be coded as indicator variables before input. Use `x.test` together with `x.adj` when an automatically constructed hypothesis matrix `Gamma` is desired.
`x.adj`	Covariates that need to be adjusted. Requirements are the same as `x.test` above. Use with `x.test` when an automatically constructed hypothesis matrix `Gamma` is desired.
`x`	Optional design matrix containing all covariates. Default value is `NULL`. Only used if matrix `Gamma` is provided as input. It can be a vector, matrix, or dataframe. All K-level categorical variables should be converted to K-1 indicator variables before input.
`Gamma`	Optional hypothesis matrix specifying the linear combinations of regression coefficients to be tested. Not used if data are entered as `x.test` and `x.adj`.
`b`	Optional numeric vector specifying the right-hand side of the null hypothesis $H_0 : \Gamma\beta_j = b$. Its length must equal the number of rows of `Gamma`. If `b = NULL`, CAFT uses the median-based null value estimated across taxa.
`filter.thresh`	A real value between 0 and 1 for OTU table sample presence filtering. Taxa with presence proportion less than or equal to `filter.thresh` are skipped in the taxon-level fitting and testing steps. The default is `0.05`.
`fdr.nominal`	The nominal false discovery rate (FDR). The default is `0.20`.
`adjust.method`	A character string. Use multiple comparison/testing adjustment methods to control the family-wise error rate/false discovery rate. Default is `"BH"`. See `p.adjust` for the details.

`regularize` A logical value. If `TRUE`, adds a small penalty to the rank-based score to stabilize estimation and reduce small-sample bias. Use this when the unpenalized rank equations have flat directions, there is near-separation, heavy censoring/ties, or the optimizer fails to converge. Default is `TRUE`.

`test.method` A character string. The variance estimator used to estimate the variance of the score equation. The methods of `"Cox"`, `"rank"`, and `"martingale"` are available. Default is `"rank"`.

`n.cores` Integer. Number of CPU cores to use for parallel computation. Default is `1` (no parallelism). If `n.cores > 1`, the function runs tasks in parallel using `foreach/doParallel` with a PSOCK cluster. On typical desktops/laptops, a good choice is `max(1L, parallel::detectCores() - 1L)`.

`boot.B` Integer. Number of bootstrap replicates used for empirical calibration of taxon-level p-values. The default is `0L`, which disables bootstrap calibration. At least two bootstrap replicates are required for bootstrap calibration.

`boot.parallel` Logical. If `TRUE` and `boot.B >= 2`, the bootstrap loop is parallelized over bootstrap replicates. The default is `FALSE`.

`boot.n.cores` Integer. Number of CPU cores used for outer bootstrap parallelization when `boot.parallel = TRUE`. The default is `1L`.

`boot.seed` Optional integer seed for reproducible bootstrap resampling. The default is `NULL`.

`boot.return.dist` Logical. If `TRUE`, return the bootstrap-centered beta distribution and the names of taxa used in the bootstrap calibration. The default is `FALSE`.

`verbose` Logical. If `TRUE`, progress messages are shown during bootstrap computation. The default is `FALSE`.

`return.mr.resid` Logical; if `TRUE`, return subject-level residual contribution matrices from the observed CAFT fit. Default is `FALSE`. The returned residuals are the compensated Cox-type score contributions computed by `mySi.no.surv()` and evaluated at the restricted estimate used in the score test.

Details

CAFT fits a rank-based accelerated failure time model to the negative log-relative abundance of each taxon, treating zero counts as censored values below the sample-specific detection limit. The resulting taxon-specific regression coefficients are compared through compositional contrasts.

With the default `x.test`/`x.adj` interface, the wrapper constructs the full design matrix internally and tests the columns corresponding to `x.test`, while adjusting for the columns in `x.adj`. For more general hypotheses, users should provide a full design matrix `x` and a hypothesis matrix `Gamma`.

This wrapper implements the refactored CAFT workflow. It first calls `caft_estimate()` to compute unrestricted taxon-level estimates, then calls `caft_test()` to perform the restricted score test. If `boot.B >= 2`, it further calls `caft_bootstrap()` for bootstrap calibration. This separation allows the unrestricted estimates to be reused when testing different hypotheses or different choices of `b`.

When `boot.B >= 2`, CAFT performs fixed-design bootstrap calibration. The observed model is first fit, and the bootstrap is carried out on the taxa that pass the original filtering step. In each bootstrap replicate, samples are resampled with replacement from the OTU table while the covariates are

kept fixed. The model is then refit with `filter.thresh = 0` on the same set of taxa. Bootstrap p-values are computed from the centered bootstrap distribution of the tested coefficient contrasts. For a single testing variable, both a two-sided empirical beta-tilde p-value and a studentized chi-square/Wald bootstrap p-value are returned. For multiple testing variables, the bootstrap p-value is based on a multivariate statistic; in this case `p.boot` and `p.boot.chi` are identical.

If `boot.parallel = TRUE`, the outer bootstrap loop is parallelized over bootstrap replicates. To avoid nested parallelism, the inner CAFT fitting routine is run with `n.cores = 1L` inside each bootstrap worker.

Value

Return a list consisting of

beta.est A matrix containing the unrestricted coefficient estimates from the taxon-level log-linear CAFT model. The number of columns matches the number of covariates in the internally constructed or user-supplied design matrix.

betahat.median The median-based null reference value used when `b = NULL`. If `b` is supplied by the user, this is returned as `NA` because the median-based null value is not computed.

b.null The null value actually used in the restricted score test. This equals `betahat.median` when `b = NULL`, and equals the user-specified `b` otherwise.

b.user.specified Logical indicator of whether `b` was supplied by the user. If `FALSE`, `b.null` was computed as the median-based reference value. If `TRUE`, `b.null` equals the user-specified null value.

b.user The user-specified null value, returned only when `b` is supplied. Otherwise `NULL`.

Gamma The hypothesis matrix used in the restricted score test.

Gamma.source Character string indicating whether `Gamma` was user-supplied or constructed internally from `x.test`/`x.adj`.

rank.teststat Test statistics from the proposed score test.

rank.teststat.norm Numeric matrix with the normalized rank-based test statistics for each taxon (rows) and each tested contrast (columns, in the order of `Gamma`'s rows). Under the null, entries are approximately $N(0, 1)$; larger absolute values indicate stronger evidence against the null.

skip.otu The names of skipped OTU during taxa presence filtering.

skip.rare Indicator for taxa skipped by the rarity filter.

skip.fail.rank.fit.pen Indicator for taxa whose unrestricted rank-based fit failed.

skip.fail.rank.test.pen Indicator for taxa whose restricted estimation or rank-based score test failed.

fit.error.messages Error messages from failed unrestricted taxon-level fits, if any.

fit.error.messages.test Error messages from failed restricted taxon-level tests, if any.

p.otu P-values for individual OTU association tests.

df.test Degrees of freedom per taxon in the test.

beta.est.r Matrix of constrained (restricted) estimates from the restricted score test.

p.marginal.otu Taxa with raw p-values smaller than `fdr.nominal` based on `p.otu`.

q.otu Q-values from adjusting p-values by the method specified in `adjust.method`, for individual OTU association tests.

q.detected.otu Detected significantly differential abundant taxa (denoted by the column names of the OTU table) at the nominal FDR based on `q.otu`.

p.boot Bootstrap-calibrated empirical beta-tilde p-values. If `boot.B` < 2, this is not returned.

q.boot Multiplicity-adjusted values from `p.boot`.

p.boot.marginal.otu Taxa detected using `p.boot` < `fdr.nominal`. Empty if bootstrap calibration is not run.

q.boot.detected.otu Taxa detected using `q.boot` < `fdr.nominal`. Empty if bootstrap calibration is not run.

p.boot.chi Studentized chi-square/Wald bootstrap p-values. For multiple testing variables, these are the same as `p.boot`. If `boot.B` < 2, this is not returned.

q.boot.chi Multiplicity-adjusted values from `p.boot.chi`.

p.boot.chi.marginal.otu Taxa detected using `p.boot.chi` < `fdr.nominal`. Empty if bootstrap calibration is not run.

q.boot.chi.detected.otu Taxa detected using `q.boot.chi` < `fdr.nominal`. Empty if bootstrap calibration is not run.

boot.median Matrix of bootstrap null values used to center the bootstrap coefficient estimates. When `b` is user-specified, these values equal the specified null value across bootstrap replicates.

boot.n.success Number of bootstrap replicates that produced complete bootstrap null values.

boot.n.fail Number of bootstrap replicates skipped or failed.

boot.beta.tilde Returned only when `boot.return.dist` = `TRUE`. Array of bootstrap-centered tested coefficient contrasts.

boot.taxa Returned only when `boot.return.dist` = `TRUE`. Names of taxa used in the bootstrap calibration.

mr.resid Returned only when `return.mr.resid` = `TRUE`. A list of length equal to the number of taxa. Each element is an $n \times p$ matrix of subject-level compensated Cox-type residual contributions, evaluated at the restricted estimate used in the score test.

Examples

```
data(Colon)

count.tab <- Colon$otu
sample.tab <- Colon$meta

keep.sample <- !is.na(sample.tab$age)
count.tab <- count.tab[keep.sample, , drop = FALSE]
sample.tab <- sample.tab[keep.sample, , drop = FALSE]

## Use a small subset of taxa for a fast example.
prev <- colSums(count.tab > 0)
keep.top <- names(sort(prev, decreasing = TRUE))[
  seq_len(min(20L, length(prev)))
]
count.tab <- count.tab[, keep.top, drop = FALSE]
```

```
## Remove samples with zero library size after subsetting taxa.
keep.lib <- rowSums(count.tab) > 0
count.tab <- count.tab[keep.lib, , drop = FALSE]
sample.tab <- sample.tab[keep.lib, , drop = FALSE]

Disease1 <- Disease2 <- rep(0, nrow(sample.tab))
Disease1[sample.tab$disease == "CRC"] <- 1
Disease2[sample.tab$disease == "adenoma"] <- 1

Age <- as.numeric(sample.tab$age)
Gender <- as.numeric(factor(sample.tab$gender)) - 1

x.test <- cbind(CRC = Disease1, adenoma = Disease2)
x.adj <- cbind(Age = Age, Gender = Gender)

## Standard CAFT wrapper with a user-specified null value.
res <- caft(
  otu.table = count.tab,
  x.test = x.test,
  x.adj = x.adj,
  b = c(0, 0),
  boot.B = 0L
)

head(res$p.otu)
res$q.detected.otu
res$Gamma
res$Gamma.source

## Separated estimation and testing workflow.
x <- cbind(x.test, x.adj)

Gamma <- cbind(
  diag(ncol(x.test)),
  matrix(0, nrow = ncol(x.test), ncol = ncol(x.adj))
)

est <- caft_estimate(
  otu.table = count.tab,
  x = x
)

res.test <- caft_test(est, Gamma = Gamma, b = c(0, 0))
head(res.test$p.otu)
res.test$b.null

## Reuse the same unrestricted estimates with a different user-specified
## null value.
res.b <- caft_test(est, Gamma = Gamma, b = c(0.1, 0.1))
res.b$b.null


## Bootstrap calibration with a small number of replicates for illustration.
```

```
res.boot <- caft(
  otu.table = count.tab,
  x.test = x.test,
  x.adj = x.adj,
  b = c(0, 0),
  boot.B = 2L,
  boot.seed = 1L
)

res.boot$boot.n.success
head(res.boot$p.boot)
```

`caft_bootstrap` *Bootstrap Calibration for CAFT Tests*

Description

Provides bootstrap calibration of taxon-level p-values for an existing CAFT restricted score test object returned by `caft_test()`. Bootstrap calibration is mainly intended as a sensitivity analysis or preferred calibration method when the number of taxa is small in the data set.

Usage

```
caft_bootstrap(
  res,
  boot.B = 1000L,
  boot.parallel = FALSE,
  boot.n.cores = 1L,
  boot.seed = NULL,
  boot.return.dist = FALSE,
  verbose = FALSE
)
```

Arguments

`res`	An object of class `"caft_test"` returned by `caft_test()`.
`boot.B`	Integer. Number of bootstrap replicates used for empirical calibration of taxon-level p-values. The default is `1000L`. At least two bootstrap replicates are required for bootstrap calibration.
`boot.parallel`	Logical. If `TRUE`, the bootstrap loop is parallelized over bootstrap replicates. The default is `FALSE`.
`boot.n.cores`	Integer. Number of CPU cores used for outer bootstrap parallelization when `boot.parallel = TRUE`. The default is `1L`.
`boot.seed`	Optional integer seed for reproducible bootstrap resampling. The default is `NULL`.

`boot.return.dist`
: Logical. If `TRUE`, return the bootstrap-centered beta distribution and the names of taxa used in the bootstrap calibration. The default is `FALSE`.

`verbose`
: Logical. If `TRUE`, progress messages are shown during sequential bootstrap computation. The default is `FALSE`.

Details

This function is intended to be used after `caft_estimate()` and `caft_test()`. The observed restricted score test is not recomputed. Instead, `caft_bootstrap()` uses the hypothesis matrix `Gamma`, the null value `b.null`, the nominal FDR threshold, and the adjustment method stored in the `caft_test` object.

CAFT bootstrap calibration uses a fixed-design bootstrap procedure. Samples are resampled with replacement from the observed OTU table while the covariates are kept fixed. The bootstrap is carried out on the taxa that pass the original filtering step. In each bootstrap replicate, the unrestricted CAFT model is refit using `caft_estimate()` with `filter.thresh = 0` on the same set of taxa. The inner CAFT fitting routine is run with `n.cores = 1L` inside each bootstrap replicate to avoid nested parallelism.

Bootstrap p-values are computed from the centered bootstrap distribution of the tested coefficient contrasts. If the original test used the default median-based null value, the bootstrap null value is recomputed within each bootstrap replicate from the bootstrap taxon-level contrasts. If the original test used a user-specified `b`, the same fixed null value is used in every bootstrap replicate.

For a single testing variable, both a two-sided empirical beta-tilde bootstrap p-value and a studentized chi-square/Wald bootstrap p-value are returned. For multiple testing variables, the bootstrap p-value is based on a multivariate statistic; in this case `p.boot` and `p.boot.chi` are identical.

If `boot.parallel = TRUE`, the outer bootstrap loop is parallelized over bootstrap replicates using `foreach`/`doParallel`. Reproducible parallel bootstrap sampling is handled through `doRNG`.

Value

The input `caft_test` object with additional bootstrap calibration results:

p.boot Bootstrap-calibrated empirical beta-tilde p-values. If `boot.B < 2`, this is a named vector of `NA` values.

q.boot Multiplicity-adjusted values from `p.boot`, computed using the adjustment method stored in `res$adjust.method`.

p.boot.marginal.otu Taxa detected using `p.boot < fdr.nominal`. Empty if bootstrap calibration is not run.

q.boot.detected.otu Taxa detected using `q.boot < fdr.nominal`. Empty if bootstrap calibration is not run.

p.boot.chi Studentized chi-square/Wald bootstrap p-values. For multiple testing variables, these are the same as `p.boot`. If `boot.B < 2`, this is a named vector of `NA` values.

q.boot.chi Multiplicity-adjusted values from `p.boot.chi`.

p.boot.chi.marginal.otu Taxa detected using `p.boot.chi < fdr.nominal`. Empty if bootstrap calibration is not run.

q.boot.chi.detected.otu Taxa detected using `q.boot.chi < fdr.nominal`. Empty if bootstrap calibration is not run.

boot.median Matrix of bootstrap null values used to center the bootstrap coefficient estimates. When `b` is user-specified, these values equal the specified null value across bootstrap replicates.

boot.n.success Number of bootstrap replicates that produced complete bootstrap null values.

boot.n.fail Number of bootstrap replicates skipped or failed.

boot.beta.tilde Returned only when `boot.return.dist = TRUE`. Array of bootstrap-centered tested coefficient contrasts.

boot.taxa Returned only when `boot.return.dist = TRUE`. Names of taxa used in the bootstrap calibration.

Examples

```
data(Colon)

count.tab <- Colon$otu
sample.tab <- Colon$meta

keep.sample <- !is.na(sample.tab$age)
count.tab <- count.tab[keep.sample, , drop = FALSE]
sample.tab <- sample.tab[keep.sample, , drop = FALSE]

## Use a small subset of taxa for a fast example.
prev <- colSums(count.tab > 0)
keep.top <- names(sort(prev, decreasing = TRUE))[seq_len(min(5L, length(prev)))]
count.tab <- count.tab[, keep.top, drop = FALSE]

## Remove samples with zero library size after subsetting taxa.
keep.lib <- rowSums(count.tab) > 0
count.tab <- count.tab[keep.lib, , drop = FALSE]
sample.tab <- sample.tab[keep.lib, , drop = FALSE]

Disease1 <- rep(0, nrow(sample.tab))
Disease1[sample.tab$disease == "CRC"] <- 1

Age <- as.numeric(sample.tab$age)
Gender <- as.numeric(factor(sample.tab$gender)) - 1

x.test <- cbind(CRC = Disease1)
x.adj <- cbind(Age = Age, Gender = Gender)

x <- cbind(x.test, x.adj)

Gamma <- cbind(
  diag(ncol(x.test)),
  matrix(0, nrow = ncol(x.test), ncol = ncol(x.adj))
)

est <- caft_estimate(
```

```
  otu.table = count.tab,
  x = x,
  regularize = TRUE,
  n.cores = 1L
)

res <- caft_test(est, Gamma = Gamma, b = 0)


## Use a very small number of replicates only for illustration.
res.boot <- caft_bootstrap(
  res,
  boot.B = 2L,
  boot.seed = 1L
)

res.boot$boot.n.success
head(res.boot$p.boot)
```

`caft_estimate` *Unrestricted CAFT Estimation*

Description

Fits the unrestricted rank-based compositional accelerated failure time model for each taxon in microbiome count data with zero cells. Zero counts are treated as censored observations below sample-specific detection limits. This function performs only taxon-level unrestricted estimation and does not conduct hypothesis testing, multiple-testing adjustment, or bootstrap calibration.

Usage

```
caft_estimate(
  otu.table,
  x,
  filter.thresh = 0.05,
  regularize = TRUE,
  n.cores = 1L
)
```

Arguments

`otu.table`	The community OTU table (or taxa count table). Each row corresponds to a sample and each column corresponds to one OTU (taxa).
`x`	Design matrix containing all covariates. All K-level categorical variables should be converted to K-1 indicator variables before input.
`filter.thresh`	a real value between 0 and 1 for OTU table sample presence filtering. Any OTUs present in fewer than `filter.thresh` proportion of samples are filtered out. We set the default to be 0.05.

`regularize` A logical value. If `TRUE`, adds a small penalty to the rank-based score to improve numerical stability. This can be useful when the unpenalized rank equations have flat directions, heavy censoring, many ties, or when the optimizer has convergence difficulty. Default is `TRUE`.

`n.cores` Integer. Number of CPU cores to use for parallel computation. Default is `1` (no parallelism). If `n.cores > 1`, the function runs tasks in parallel using `foreach`/`doParallel` with a PSOCK cluster. On typical desktops/laptops, a good choice is `max(1L, parallel::detectCores() - 1L)`.

Details

CAFT fits a rank-based accelerated failure time model to the negative log-relative abundance of each taxon. Zero counts are treated as censored observations below the sample-specific detection limit, avoiding the use of pseudocounts.

This function performs only the unrestricted taxon-level estimation step. It returns the unrestricted coefficient estimates and the transformed survival-type data needed by downstream CAFT testing functions. It does not compute the median-based null value, restricted estimates, score test statistics, p-values, q-values, or bootstrap-calibrated p-values.

Value

An object of class `"caft_est"` containing unrestricted taxon-level coefficient estimates, censoring variables, the centered design matrix, and filtering/fitting status indicators.

call The matched function call.

otu.table The OTU count table used in the unrestricted CAFT estimation after input checking.

lib.size The sequencing library size for each sample.

x.raw The original, uncentered covariate matrix used in the model.

x The centered covariate matrix used in the unrestricted rank-based AFT estimation.

x.name Names of the covariates in the design matrix.

filter.thresh The prevalence filtering threshold used in the analysis.

regularize Logical value indicating whether regularized rank-based AFT estimation was used.

n.data Number of samples used in the unrestricted estimation.

n.taxa Number of taxa in the OTU table.

n.param Number of covariates in the design matrix.

taxa.name Names of the taxa in the OTU table.

taxa.used Names of taxa that passed the prevalence filter and had successful unrestricted estimation.

t.star.all Transformed censored abundance outcomes used in the rank-based AFT estimation.

delta.all Censoring indicators for `t.star.all`, where one indicates an observed nonzero count and zero indicates a censored zero count.

beta.est A matrix that includes the unrestricted estimated coefficients obtained by fitting each OTU count to the CAFT log-linear model. The number of columns matches the number of covariates from `x`.

skip.otu Indicator for OTUs skipped during taxa presence filtering.

skip.rare Indicator for OTUs skipped during taxa presence filtering.

skip.fail.rank.fit.pen Indicator for OTUs whose unrestricted regularized rank-based AFT estimation failed.

fit.error.messages Error messages from failed unrestricted taxon-level fits, if any.

Examples

```
data(Colon)

count.tab <- Colon$otu
sample.tab <- Colon$meta

keep.sample <- !is.na(sample.tab$age)
count.tab <- count.tab[keep.sample, , drop = FALSE]
sample.tab <- sample.tab[keep.sample, , drop = FALSE]

keep.otu <- colSums(count.tab > 0) > 1
count.tab <- count.tab[, keep.otu, drop = FALSE]

## Use a small subset of taxa for a fast example.
prev <- colSums(count.tab > 0)
keep.top <- names(sort(prev, decreasing = TRUE))[seq_len(min(10L, length(prev)))]
count.tab <- count.tab[, keep.top, drop = FALSE]

## Remove samples with zero library size after subsetting taxa.
keep.lib <- rowSums(count.tab) > 0
count.tab <- count.tab[keep.lib, , drop = FALSE]
sample.tab <- sample.tab[keep.lib, , drop = FALSE]

Disease1 <- Disease2 <- rep(0, nrow(sample.tab))
Disease1[sample.tab$disease == "CRC"] <- 1
Disease2[sample.tab$disease == "adenoma"] <- 1

Age <- as.numeric(sample.tab$age)
Gender <- as.numeric(factor(sample.tab$gender)) - 1

x.test <- cbind(CRC = Disease1, adenoma = Disease2)
x.adj <- cbind(Age = Age, Gender = Gender)
x <- cbind(x.test, x.adj)

est <- caft_estimate(
  otu.table = count.tab,
  x = x,
  regularize = TRUE,
  n.cores = 1L
)

print(est)
```

`caft_test` *CAFT Restricted Score Test*

Description

Performs the CAFT restricted score test using an existing unrestricted estimation object returned by `caft_estimate()`. This function separates testing from unrestricted estimation: the unrestricted taxon-level coefficient estimates are reused from `caft_estimate()`, and only the restricted estimation and score testing steps are carried out for the specified hypothesis.

Usage

```
caft_test(
  est,
  Gamma = NULL,
  x.test = NULL,
  x.adj = NULL,
  b = NULL,
  fdr.nominal = 0.2,
  adjust.method = "BH",
  regularize = est$regularize,
  test.method = "rank",
  n.cores = 1L,
  return.mr.resid = FALSE
)
```

Arguments

`est`	An object of class `"caft_est"` returned by `caft_estimate()`.
`Gamma`	Optional hypothesis matrix specifying the linear combinations of regression coefficients to be tested. If `Gamma = NULL`, `x.test` must be supplied and `caft_test()` constructs the corresponding hypothesis matrix internally.
`x.test`	Optional covariate matrix corresponding to the tested covariates used in the unrestricted fit. Used only when `Gamma = NULL` to construct the hypothesis matrix.
`x.adj`	Optional covariate matrix corresponding to adjustment covariates used in the unrestricted fit. Used only when `Gamma = NULL`.
`b`	Optional numeric vector specifying the right-hand side of the null hypothesis $H_{0j} : \Gamma\beta_j = b$ for each taxon j. Its length must equal the number of rows of `Gamma`. If `b = NULL`, CAFT uses a median-based reference value estimated from the unrestricted taxon-level contrasts $\Gamma\hat{\beta}_j$.
`fdr.nominal`	The nominal threshold used to report detected taxa. The default is `0.20`.
`adjust.method`	A character string specifying the p-value adjustment method used to compute `q.otu`. The default is `"BH"`. See `p.adjust`.
`regularize`	A logical value. If `TRUE`, adds a small penalty to the rank-based score to improve numerical stability in the restricted estimation step. The default uses the value stored in `est$regularize`.

`test.method` A character string specifying the variance estimator used in the rank-based score test. The available options are `"rank"`, `"Cox"`, and `"martingale"`. The default is `"rank"`.

`n.cores` Integer. Number of CPU cores to use for taxon-level parallel computation in the restricted testing step. The default is `1`.

`return.mr.resid` Logical. If `TRUE`, return subject-level residual contribution matrices evaluated at the restricted estimates. The default is `FALSE`.

Details

Let β_j denote the regression coefficient vector for taxon j. CAFT tests taxon-level departures from a compositional reference value through

$$H_{0j} : \Gamma\beta_j = b,$$

where `Gamma` specifies the tested linear combinations of regression coefficients and `b` is the corresponding null reference value.

This function is intended for use after `caft_estimate()`. The unrestricted estimates $\hat{\beta}_j$ stored in `est$beta.est` are not recomputed. This allows users to test multiple hypotheses, or multiple choices of `b`, using the same unrestricted CAFT fit.

The hypothesis can be specified either by supplying `Gamma` directly or by supplying `x.test` and optionally `x.adj`, in which case `caft_test()` constructs the corresponding hypothesis matrix internally.

This keeps unrestricted estimation separate from hypothesis specification: `caft_estimate()` computes and stores unrestricted estimates, while `caft_test()` computes the hypothesis-specific restricted estimates and score tests for the supplied `Gamma` and `b`.

If `b = NULL`, the null reference value is estimated from the unrestricted taxon-level contrasts $\Gamma\hat{\beta}_j$. For a one-dimensional contrast, the ordinary median across taxa is used. For multiple contrasts, a robust multivariate location estimate from `ICSNP::HR.Mest()` is used. If `b` is supplied by the user, the supplied value is used directly as the null reference value.

The output records whether the null value was user-specified through `b.user.specified`. This is used by `caft_bootstrap()` to determine whether the same fixed null value should be used in every bootstrap replicate, or whether the median-based null value should be recomputed within each bootstrap replicate.

For each taxon, CAFT obtains the restricted estimate under $H_{0j} : \Gamma\beta_j = b$ and then computes the rank-based score test statistic. Raw taxon-level p-values are returned as `p.otu`, and multiplicity-adjusted values are returned as `q.otu`. Bootstrap calibration is not performed by this function; it is handled separately by `caft_bootstrap()`.

Value

An object of class `"caft_test"`, which also inherits from `"caft_est"`. The object contains the original unrestricted estimation object together with the restricted testing results.

beta.est Unrestricted taxon-level coefficient estimates inherited from `caft_estimate()`.

Gamma The hypothesis matrix used in the restricted score test.

Gamma.source Character string indicating whether Gamma was user-supplied or constructed internally from x.test/x.adj.

Lambda The nuisance-parameter contrast matrix used in the restricted estimation step.

Gamma.ginv Generalized inverse of Gamma.

Lambda.ginv Generalized inverse of Lambda, when applicable.

n.test Number of tested contrasts, equal to the number of rows of Gamma.

betahat.median The median-based null reference value estimated from the unrestricted taxon-level contrasts $\Gamma\hat{\beta}_j$ when b = NULL. If b is supplied by the user, this is returned as NA because the median-based null value is not computed.

b.null The null value actually used in the restricted score test. This equals betahat.median when b = NULL, and equals the user-specified b otherwise.

b.user.specified Logical indicator of whether b was supplied by the user. If FALSE, b.null was computed as the median-based reference value. If TRUE, b.null equals the user-specified null value.

b.user The user-specified null value, returned only when b is supplied. Otherwise NULL.

rank.teststat Rank-based score test statistics for individual taxa.

rank.teststat.norm Numeric matrix with normalized score statistics for each taxon and each tested contrast. Rows correspond to taxa and columns correspond to the rows of Gamma.

p.otu Raw p-values for individual taxon-level association tests.

df.test Degrees of freedom used in the taxon-level score tests.

beta.est.r Restricted coefficient estimates obtained under $H_{0j} : \Gamma\beta_j = b$.

p.marginal.otu Taxa with raw p-values smaller than fdr.nominal based on p.otu.

q.otu Multiplicity-adjusted p-values computed from p.otu using the method specified by adjust.method.

q.detected.otu Taxa with adjusted p-values smaller than fdr.nominal.

skip.fail.rank.test.pen Indicator for taxa whose restricted estimation or rank-based score test failed.

fit.error.messages.test Error messages from failed restricted taxon-level tests, if any.

fdr.nominal The nominal threshold used to report detected taxa.

adjust.method The p-value adjustment method used to compute q.otu.

test.method The variance estimator used in the score test.

mr.resid Returned only when return.mr.resid = TRUE. A list of subject-level residual contribution matrices evaluated at the restricted estimates.

Examples

```
data(Colon)

count.tab <- Colon$otu
sample.tab <- Colon$meta

keep.sample <- !is.na(sample.tab$age)
count.tab <- count.tab[keep.sample, , drop = FALSE]
sample.tab <- sample.tab[keep.sample, , drop = FALSE]
```

```
## Use a small subset of taxa for a fast example.
prev <- colSums(count.tab > 0)
keep.top <- names(sort(prev, decreasing = TRUE))[seq_len(min(10L, length(prev)))]
count.tab <- count.tab[, keep.top, drop = FALSE]

## Remove samples with zero library size after subsetting taxa.
keep.lib <- rowSums(count.tab) > 0
count.tab <- count.tab[keep.lib, , drop = FALSE]
sample.tab <- sample.tab[keep.lib, , drop = FALSE]

Disease1 <- Disease2 <- rep(0, nrow(sample.tab))
Disease1[sample.tab$disease == "CRC"] <- 1
Disease2[sample.tab$disease == "adenoma"] <- 1

Age <- as.numeric(sample.tab$age)
Gender <- as.numeric(factor(sample.tab$gender)) - 1

x.test <- cbind(CRC = Disease1, adenoma = Disease2)
x.adj <- cbind(Age = Age, Gender = Gender)

x <- cbind(x.test, x.adj)
Gamma <- cbind(
  diag(ncol(x.test)),
  matrix(0, nrow = ncol(x.test), ncol = ncol(x.adj))
)

est <- caft_estimate(
  otu.table = count.tab,
  x = x,
  regularize = TRUE,
  n.cores = 1L
)

res <- caft_test(est, Gamma = Gamma, b = c(0, 0))
print(res)

## Equivalent interface: construct Gamma internally from x.test/x.adj.
res.default <- caft_test(est, x.test = x.test, x.adj = x.adj, b = c(0, 0))
res.default$Gamma

## Reuse the same unrestricted estimates with a different user-specified
## null value.
res.b <- caft_test(est, Gamma = Gamma, b = c(0.1, 0.1))

## Example with a full design matrix and an explicit Gamma.
x.all <- cbind(CRC = Disease1, adenoma = Disease2, Age = Age, Gender = Gender)
est2 <- caft_estimate(otu.table = count.tab, x = x.all)
Gamma2 <- rbind(c(1, 0, 0, 0), c(0, 1, 0, 0))
res2 <- caft_test(est2, Gamma = Gamma2, b = c(0, 0))
```

Colon *Example colon microbiome dataset*

Description

This gut microbiome data set is originally included in the R package "curatedMetagenomicData" and the package installation is required through "Bioconductor". It specifically focuses on the adult colorectal cancer studies using the stool samples. More details can be found through the following references.

Usage

```
data(Colon)
```

Format

An object of class `list` of length 3.

References

Pasolli, E., Schiffer, L., Manghi, P., et al. (2017). Accessible, curated metagenomic data through ExperimentHub. *Nature Methods*, 14(11), 1023–1024.

Examples

```
data(Colon)
```

`otuboxplot` *Generates a ggplot2-based boxplot showing the relative abundance of a specified OTU (Operational Taxonomic Unit) across different sample groups.*

Description

Generates a ggplot2-based boxplot showing the relative abundance of a specified OTU (Operational Taxonomic Unit) across different sample groups.

Usage

```
otuboxplot(plot.otu, count.data, groups, plot.title = NULL, type = "original")
```

Arguments

`plot.otu`	A character string giving the OTU name to plot. Must match one of the column names in `count.data`.
`count.data`	A numeric matrix or data frame of OTU counts (samples in rows, OTUs in columns).
`groups`	A factor vector indicating group membership for each sample. Must be the same length as the number of rows in `count.data`.
`plot.title`	An optional title for the plot. If `NULL`, the OTU name in `plot.otu` will be used.

type — A character string indicating which values to plot. Use "original" to plot empirical relative abundances. Use "transformed" to plot negative log10-transformed relative abundances. A pseudo-count of 1 is added before log transformation to avoid taking log10(0).

Details

The function converts raw counts to relative abundances, selects the OTU specified in plot.otu, and creates a grouped boxplot. If groups is not a factor, the function will return an error. Group means are shown as red dots.

Value

A ggplot object showing the OTU's relative abundance across sample groups.

Examples

```
# Example 1: Basic usage
set.seed(123)
count.data <- matrix(rpois(300, lambda = 10), nrow = 30, ncol = 10)
colnames(count.data) <- paste0("OTU", 1:10)
groups <- factor(rep(c("Control", "Treatment"), each = 15))
otuboxplot("OTU3", count.data, groups)

# Example 2: Combining two binary indicators into one group variable
set.seed(456)
n <- 40
count.data <- matrix(rpois(n * 8, lambda = 12), nrow = n, ncol = 8)
colnames(count.data) <- paste0("OTU", 1:8)
group1 <- rbinom(n, 1, 0.5)
group2 <- rbinom(n, 1, 0.5)
group <- factor(paste0("Y", group1, "_C", group2))
otuboxplot("OTU5", count.data, group, type = "transformed")
```

---

print.caft_est *Print Method for Unrestricted CAFT Estimation*

Description

Print Method for Unrestricted CAFT Estimation

Usage

```
## S3 method for class 'caft_est'
print(x, ...)
```

Arguments

x — An object of class "caft_est".

... — Additional arguments, currently unused.

Value

Invisibly returns `x`.

`print.caft_test` *Print Method for CAFT Restricted Score Test*

Description

Print Method for CAFT Restricted Score Test

Usage

```
## S3 method for class 'caft_test'
print(x, ...)
```

Arguments

`x`	An object of class `"caft_test"`.
`...`	Additional arguments, currently unused.

Value

Invisibly returns `x`.

`URT` *URT upper respiratory tract microbiome data (filtered)*

Description

A small example dataset derived from Charlson et al. (2010) as distributed by the **LOCOM** package and filtered to left oropharyngeal samples. Provided for illustrating CAFT workflows.

Usage

```
data("URT")
```

Format

A list with three components:

otu Integer matrix of counts, samples in rows and taxa in columns.

meta Data frame of sample metadata (e.g., SmokingStatus, Sex).

tax Data frame of taxonomy annotations for the taxa.

Details

Zeros in `otu` correspond to under-detection and are handled as left-censored measurements by CAFT.

Source

Charlson ES et al. (2010) PLoS One 5(12):e15216; dataset accessed via the **LOCOM** R package.

Examples

```
data("URT", package = "CAFT")
str(URT, max.level = 1)
# counts:
dim(URT$otu)
# metadata variables:
names(URT$meta)
```

Index