Package: radEmu (via r-universe)

October 12, 2026

Title Using Relative Abundance Data to Estimate of Multiplicative Differences in Mean Absolute Abundance

Version 2.4.0.0

Description A differential abundance method for the analysis of microbiome data. 'radEmu' estimates fold-differences in the abundance of taxa across samples relative to ``typical'' fold-differences. Notably, it does not require pseudocounts, nor choosing a denominator taxon. For more details, see Clausen et al. (2026) <doi:10.1093/biomet/asag009>.

URL https://github.com/statdivlab/radEmu, https://statdivlab.github.io/radEmu/

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

Depends MASS, Matrix, methods, magrittr, rlang, R (>= 2.10)

Suggests testthat (>= 3.0.0), numDeriv, phyloseq, TreeSummarizedExperiment, SummarizedExperiment, SingleCellExperiment, knitr, dplyr, ggplot2, stringr, parallel, rmarkdown

Config/testthat/edition 3

VignetteBuilder knitr

Config/roxygen2/version 8.0.0

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

Date/Publication 2026-10-12 07:46:50 UTC

RemoteUrl https://github.com/statdivlab/rademu

RemoteRef HEAD

RemoteSha eb2d6c596358ddafcef539a84a6ab034bb1d8168

Contents

control_fn . . . 2
dpseudohuber_median_dx . . . 4
emuFit . . . 5
emuFit_check . . . 10
emuFit_micro . . . 12
emuFit_micro_penalized . . . 13
fit_null . . . 15
fit_null_discrete . . . 16
fit_null_scc . . . 18
fit_null_symmetric . . . 19
fit_null_symmetric_subset . . . 21
make_design_matrix . . . 23
make_diagnostics_plots . . . 23
make_reference_constraints . . . 24
plot.emuFit . . . 25
power_model . . . 26
print.emuFit . . . 27
pseudohuber_median . . . 27
score_test . . . 28
simulate_data . . . 31
wirbel_otu . . . 32
wirbel_otu_small . . . 32
wirbel_sample . . . 33
wirbel_sample_small . . . 34
wirbel_taxonomy . . . 35

Index **36**

`control_fn` *Create list of control options (to pass to* `emuFit()`*)*

Description

Create list of control options (to pass to `emuFit()`)

Usage

```
control_fn(
  control = list(),
  max_step = 1,
  ignore_stop = FALSE,
  use_fullmodel_info = FALSE,
  use_fullmodel_cov = FALSE,
  use_both_cov = FALSE,
  inner_maxit = 25,
  inner_tol = 1,
```

```
  c1 = 1e-04,
  trackB = FALSE,
  return_nullB = FALSE,
  return_score_components = FALSE,
  return_both_score_pvals = FALSE,
  B_null_tol = 0.001,
  rho_init = 1,
  tau = 2,
  kappa = 0.8,
  constraint_tol = 1e-05,
  ntries = 4
)
```

Arguments

control	Current control list (optional), will augment it with missing arguments
max_step	Maximum stepsize; update directions computed during estimation (under the alternative). Will be rescaled if a step in any parameter exceeds this value. Defaults to 1.
ignore_stop	whether to ignore stopping criteria and run maxit iterations (helpful for diagnostic plots to determine convergence).
use_fullmodel_info	Used in estimation under the null hypothesis. Whether to use information matrix from estimation under the alternative hypothesis to construct the robust score statistic (instead of information matrix from estimation under the null hypothesis). Defaults to FALSE.
use_fullmodel_cov	Used in estimation under the null hypothesis. Whether to use covariance matrix from estimation under the alternative hypothesis to construct the robust score statistic (instead of covariance matrix from estimation under the null hypothesis). Defaults to FALSE.
use_both_cov	Used in estimation under the null hypothesis. Whether to do score test twice, once with covariance matrix under the alternative hypothesis and once with covariance matrix under the null hypothesis. Defaults to FALSE.
inner_maxit	Used in estimation under the null hypothesis. Maximum number of iterations within each inner loop of estimation under null hypothesis algorithm. Default is 25.
inner_tol	Used in estimation under the null hypothesis. Convergence tolerance within each inner loop of estimation under null hypothesis algorithm. Default is 1.
c1	Used in estimation under the null hypothesis. Parameter for Armijo line search. Default is 1e-4.
trackB	Used in estimation under the null hypothesis. When TRUE will track the value of B in each iteration of optimization algorithm. Defaults to FALSE.
return_nullB	Used in estimation under the null hypothesis. When TRUE will return the final value of B under each null hypothesis tested. Defaults to FALSE.

`return_score_components`
: Used in estimation under the null hypothesis. When `TRUE` will return the components of the robust score test statistic for each null hypothesis tested. Defaults to `FALSE`.

`return_both_score_pvals`
: Used in estimation under the null hypothesis, with `use_both_cov`. Defaults to `FALSE`.

`B_null_tol`
: Used in estimation under the null hypothesis, for the augmented Lagrangian algorithm. numeric: convergence tolerance for null model fits for score testing (if max of absolute difference in B across outer iterations is below this threshold, we declare convergence). Default is `0.001`.

`rho_init`
: Used in estimation under the null hypothesis, for the augmented Lagrangian algorithm. Value at which to initiate rho parameter in augmented Lagrangian algorithm. Default is `1`.

`tau`
: Used in estimation under the null hypothesis, for the augmented Lagrangian algorithm. Value to scale `rho` by in each iteration of augmented Lagrangian algorithm that does not move estimate toward zero sufficiently. Default is `2`.

`kappa`
: Used in estimation under the null hypothesis, for the augmented Lagrangian algorithm. Value between `0` and `1` that determines the cutoff on the ratio of current distance from feasibility over distance in last iteration triggering scaling of `rho`. If this ratio is above `kappa`, `rho` is scaled by `tau` to encourage estimate to move toward feasibility.

`constraint_tol`
: Used in estimation under the null hypothesis, for the augmented Lagrangian algorithm. Constraint tolerance for fits under null hypotheses (tested element of `B` must be equal to constraint function to within this tolerance for a fit to be accepted as a solution to constrained optimization problem). Default is `1e-5`.

`ntries`
: Used in estimation under the null hypothesis, for the augmented Lagrangian algorithm. The number of times to try optimization. Successive tries will change `tau` and `inner_maxit` and retry.

Value

A list containing control options, to have more control over optimization algorithms used by `radEmu`. This can be passed into `emuFit()`.

`dpseudohuber_median_dx` *Calculate the derivative to the pseudo-Huber smoothed median*

Description

Calculate the derivative to the pseudo-Huber smoothed median

Usage

```
dpseudohuber_median_dx(x, d = 0.1, na.rm = FALSE)

dpsuedohuber_median_dx(x, d = 0.1, na.rm = FALSE)
```

Arguments

x	A vector to calculate the derivative of the pseudo-Huber smoothed median for.
d	Smoothing parameter, by default set to 0.1. As d approaches 0 the pseudo-Huber median function approaches the median and as d approaches infinity this function approaches the mean.
na.rm	Passed to pseudohuber_median, default is FALSE, if FALSE then when x includes at least one NA value then NA is returned, if TRUE then when x includes at least one NA value then that value is removed and the pseudo-Huber median is computed without it.

Value

The derivative of the calculated pseudo-Huber smoothed median over x with smoothing parameter d.

Examples

```
dpseudohuber_median_dx(x = rnorm(10), d = 0.1)
```

---

emuFit *Fit radEmu model*

Description

Fit radEmu model

Usage

```
emuFit(
  Y,
  X = NULL,
  formula = NULL,
  data = NULL,
  assay_name = NULL,
  cluster = NULL,
  constraint_fn = pseudohuber_median,
  constraint_grad_fn = dpseudohuber_median_dx,
  constraint_param = 0.1,
  verbose = FALSE,
  match_row_names = TRUE,
```

```
  unobserved_taxon_error = TRUE,
  penalize = TRUE,
  B = NULL,
  fitted_model = NULL,
  refit = TRUE,
  tolerance = 1e-04,
  maxit = 1000,
  alpha = 0.05,
  return_wald_p = FALSE,
  compute_cis = TRUE,
  max_abs_B = 250,
  run_score_tests = TRUE,
  test_kj = NULL,
  null_fit_alg = NULL,
  B_null_list = NULL,
  maxit_null = 1000,
  tol_lik = 1e-05,
  tol_test_stat = 0.01,
  tol_discrete = 0.01,
  null_window = 5,
  null_diagnostic_plots = FALSE,
  remove_zero_comparison_pvals = 0.01,
  estimates_only = FALSE,
  control = NULL,
  ...
)
```

Arguments

Y	an n x J matrix or dataframe of nonnegative observations, or a phyloseq or TreeSummarizedExperiment object containing an otu table and sample data.
X	an n x p design matrix (either provide X or data and formula)
formula	a one-sided formula specifying the form of the mean model to be fit (used with data)
data	an n x p data frame containing variables given in formula
assay_name	a string containing the desired assay name within a TreeSummarizedExperiment object. This is only required if Y is a TreeSummarizedExperiment object, otherwise this argument can be ignored.
cluster	a vector giving cluster membership for each row of Y to be used in computing GEE test statistics. Default is NULL, in which case rows of Y are treated as independent.
constraint_fn	(Optional) User-provided constraint function, if default behavior of comparing log fold-difference parameters to smoothed median over all categories is not desired. If a number is provided a single category constraint will be used with the provided category as a reference category. This argument can either be a single constraint function to be used for all rows of B, or a list of length p of constraints to be used for each row of B.

constraint_grad_fn
: (Optional) User-provided derivative of constraint function, if default behavior of comparing log fold-difference parameters to smoothed median over all categories is not desired. If `constraint_fn` is a list of constraint functions, then this argument must also be a list. If `constraint_fn` is a single number, or a list that includes a single number, then the corresponding `constraint_grad_fn` can be set to `NULL`, and will be appropriately set within the function.

constraint_param
: (Optional) If the smoothed median is used as a constraint (this is the default), parameter controlling relative weighting of elements closer and further from center. (Limit as `constraint_param` approaches infinity is the mean; as this parameter approaches zero, the minimizer of the pseudo-Huber loss approaches the median.) If constraint function is not smoothed median (implemented in `radEmu::pseudohuber_median()`) then this argument will be ignored.

verbose
: provide updates as model is being fitted? Defaults to `FALSE`. If user sets `verbose = TRUE`, then key messages about algorithm progress will be displayed. If user sets `verbose = "development"`, then key messages and technical messages about convergence will be displayed. Most users who want status updates should set `verbose = TRUE`.

match_row_names
: logical: If `TRUE`, make sure rows on covariate data and response data correspond to the same sample by comparing row names and subsetting/reordering if necessary. Default is `TRUE`.

unobserved_taxon_error
: logical: should an error be thrown if Y includes taxa that have 0 counts for all samples? Default is TRUE.

penalize
: logical: should Firth penalty be used? Default is `TRUE`. Used in estimation.

B
: starting value of coefficient matrix (p x J) for estimation. If not provided, B will be initiated as a zero matrix. Used in estimation.

fitted_model
: a fitted model produced by a separate call to emuFit; to be provided if score tests are to be run without refitting the full unrestricted model. Default is `NULL`.

refit
: logical: if `B` or `fitted_model` is provided, should full model be fit (`TRUE`) or should fitting step be skipped (`FALSE`), e.g., if score tests are to be run on an already fitted model. Default is `TRUE`.

tolerance
: tolerance for stopping criterion in estimation; once no element of B is updated by more than this value in a single step, we exit optimization. Defaults to `1e-3`. Used in estimation.

maxit
: maximum number of outer iterations to perform before exiting optimization. Default is `1000`. Used in estimation.

alpha
: nominal type 1 error level to be used to construct confidence intervals. Default is `0.05` (corresponding to 95% confidence intervals)

return_wald_p
: logical: return p-values from Wald tests? Default is `FALSE`.

compute_cis
: logical: compute and return Wald CIs? Default is `TRUE`.

max_abs_B
: maximum allowed value for elements of B (in absolute value) in full model fitting. In most cases this is not needed as Firth penalty will prevent infinite

estimates under separation. However, such a threshold may be helpful in very poorly conditioned problems (e.g., with many nearly collinear regressors). Default is 250.

run_score_tests
: logical: perform robust score testing? Default is TRUE.

test_kj
: a data frame whose rows give coordinates (in category j and covariate k) of elements of B to construct hypothesis tests for. k could also be the name of a covariate included in X or data. If you don't know which coordinates k correspond to the covariate(s) that you would like to test, run the function radEmu::make_design_matrix() in order to view the design matrix, and identify which column of the design matrix corresponds to each covariate in your model. This argument is required when running score tests.

null_fit_alg
: Which null fitting algorithm to use for score tests: "constraint_sandwich" or "augmented_lagrangian", or "discrete" when design matrix only includes categorical covariates. Default and recommended approach is "constraint_sandwich", or "discrete" for a design matrix with only categorical covariates and J < 150. Augmented lagrangian is used when J < 20.

B_null_list
: list of starting values of coefficient matrix (p x J) for null estimation for score testing. This should either be a list with the same length as test_kj. If you only want to provide starting values for some tests, include the other elements of the list as NULL.

maxit_null
: maximum number of outer iterations to perform before exiting optimization. Default is 1000. Used in estimation under null hypothesis for score tests.

tol_lik
: tolerance for relative changes in likelihood for stopping criteria. Default is 1e-5. Used in estimation under null hypothesis for score tests with "constraint_sandwich" algorithm.

tol_test_stat
: tolerance for relative changes in test statistic for stopping criteria. Default is 0.01. Used in estimation under null hypothesis for score tests with "constraint_sandwich" algorithm.

tol_discrete
: tolerance for the root mean norm of the score vector, for stopping criteria. Default is 0.01. Used in estimation under null hypothesis for score tests with "discrete" algorithm (for discrete designs).

null_window
: window to use for stopping criteria (this many iterations where stopping criteria is met). Default is 5. Used in estimation under null hypothesis for score tests with "constraint_sandwich" algorithm.

null_diagnostic_plots
: logical: should diagnostic plots be made for estimation under the null hypothesis? Default is FALSE.

remove_zero_comparison_pvals
: Should score p-values be replaced with NA for zero-comparison parameters? These parameters occur for categorical covariates with three or more levels, and represent parameters that compare a covariate level to the reference level for a category in which the comparison level and reference level both have 0 counts in all samples. These parameters can have misleadingly small p-values and are not thought to have scientifically interesting signals. We recommend removing

them before analyzing data further. If TRUE, all zero-comparison parameter p-values will be set to NA. If FALSE no zero-comparison parameter p-values will be set to NA. If a value between 0 and 1, all zero-comparison p-values below the value will be set to NA. Default is `0.01`.

`estimates_only` logical: return parameter estimates only? If `TRUE`, confidence intervals and score tests are not computed (overriding `compute_cis` and `run_score_tests`), and `test_kj` is not required. Default is `FALSE`.

`control` A list of control parameters, to have more control over estimation and hypothesis testing. See `control_fn` for details.

`...` Additional arguments. Arguments matching the names of `control_fn()` options are forwarded to that function and override defaults. Unknown arguments are ignored with a warning.

Value

`emuFit` returns a list containing elements `coef`, `B`, `penalized`, `Y_augmented`, `z_hat`, `I`, `Dy`, and `score_test_hyperparams` if score tests are run.

The `coef` table contains log fold-difference parameter estimates by covariate and outcome category (e.g., taxon for microbiome data). A log fold-difference estimate of `1` for a treatment (versus control) and taxon X can be interpreted to say that we expect taxon X is `exp(1) = 2.72` times more abundant in someone who has received the treatment compared to someone who has received the control (holding all other covariates equal), when compared to typical fold-differences in abundances of taxa in this analysis.

`coef` also includes optionally-computed confidence intervals and robust Wald p-values. Robust score statistics and score test p-values are also included in `coef`. As explained in the associated manuscript, we recommend use of the robust score test values instead of the robust Wald test p-values, due to better error rate control (i.e. fewer false positives).

If there are any zero-comparison parameters in the model, a column "zero_comparison" is also included, which is `TRUE` for any parameters that compare the level of a categorical covariate to a reference level for a category with only zero counts for both the comparison level and the reference level. This check is currently implemented for an arbitrary design matrix generated using the `formula` and `data` arguments, and for a design matrix with no more than one categorical covariate if the design matrix `X` is input directly.

`B` contains parameter estimates in matrix format (rows indexing covariates and columns indexing outcome category / taxon). `penalized` is `TRUE` if Firth penalty is used in estimation (default) and `FALSE` otherwise. `z_hat` returns the nuisance parameters (sample-specific sequencing effects). `I` and `Dy` contain an information matrix and empirical score covariance matrix computed under the full model. `score_test_hyperparams` contains parameters and hyperparameters related to estimation under the null, including whether or not the algorithm converged, which can be helpful for debugging.

Examples

```
# data frame example (for phyloseq and TreeSummarizedExperiment examples, see the vignettes)
data(wirbel_sample_small)
data(wirbel_otu_small)
emuRes <- emuFit(formula = ~ Group, data = wirbel_sample_small, Y = wirbel_otu_small,
```

```
                test_kj = data.frame(k = 2, j = 1), tolerance = 0.01)
# here we set large tolerances for the example to run quickly,
# but we recommend smaller tolerances in practice
```

`emuFit_check` *Runs checks for appropriate arguments before running* `emuFit()`

Description

Runs checks for appropriate arguments before running `emuFit()`

Usage

```
emuFit_check(
  Y,
  X = NULL,
  formula = NULL,
  data = NULL,
  assay_name = NULL,
  cluster = NULL,
  B_null_list = NULL,
  test_kj = NULL,
  match_row_names = TRUE,
  verbose = FALSE,
  remove_zero_comparison_pvals = 0.01,
  unobserved_taxon_error = TRUE,
  constraint_fn,
  constraint_grad_fn,
  constraint_param,
  run_score_tests = TRUE,
  null_fit_alg = "constraint_sandwich"
)
```

Arguments

`Y`	an n x J matrix or dataframe of nonnegative observations, or a phyloseq object containing an otu table and sample data.
`X`	an n x p matrix or dataframe of covariates (optional)
`formula`	a one-sided formula specifying the form of the mean model to be fit
`data`	an n x p data frame containing variables given in `formula`
`assay_name`	a string containing the desired assay name within a `TreeSummarizedExperiment` object. This is only required if Y is a `TreeSummarizedExperiment` object, otherwise this argument does nothing and can be ignored.
`cluster`	a vector giving cluster membership for each row of Y to be used in computing GEE test statistics. Default is NULL, in which case rows of Y are treated as independent.

`B_null_list` list of starting values of coefficient matrix (p x J) for null estimation. This should either be a list with the same length as `test_kj`. If you only want to provide starting values for some tests, include the other elements of the list as `NULL`.

`test_kj` a data frame whose rows give coordinates (in category j and covariate k) of elements of B to construct hypothesis tests for. `k` could also be the name of a covariate included in `X` or `data`. If you don't know which coordinates k correspond to the covariate(s) that you would like to test, run the function `radEmu::make_design_matrix()` in order to view the design matrix, and identify which column of the design matrix corresponds to each covariate in your model. This argument is required when running score tests.

`match_row_names`
logical: Make sure rows on covariate data and response data correspond to the same sample by comparing row names and subsetting/reordering if necessary.

`verbose` provide updates as model is being fitted? Defaults to FALSE. If user sets verbose = TRUE, then key messages about algorithm progress will be displayed. If user sets verbose = "development", then key messages and technical messages about convergence will be displayed. Most users who want status updates should set verbose = TRUE.

`remove_zero_comparison_pvals`
Should score p-values be replaced with NA for zero-comparison parameters? These parameters occur for categorical covariates with three or more levels, and represent parameters that compare a covariate level to the reference level for a category in which the comparison level and reference level both have 0 counts in all samples. These parameters can have misleadingly small p-values and are not thought to have scientifically interesting signals. We recommend removing them before analyzing data further. If TRUE, all zero-comparison parameter p-values will be set to NA. If FALSE no zero-comparison parameter p-values will be set to NA. If a value between 0 and 1, all zero-comparison p-values below the value will be set to NA. Default is `0.01`.

`unobserved_taxon_error`
logical: should an error be thrown if Y includes taxa that have 0 counts for all samples? Default is TRUE.

`constraint_fn` function g defining a constraint on rows of B; g(B_k) = 0 for rows k = 1, ..., p of B. Default function is a smoothed median (minimizer of pseudohuber loss). If a number is provided a single category constraint will be used with the provided category as a reference category. This argument can either be a single constraint function to be used for all rows of B, or a list of length p of constraints to be used for each row of B.

`constraint_grad_fn`
derivative of constraint_fn with respect to its arguments (i.e., elements of a row of B). If `constraint_fn` is a list of constraint functions, then this argument must also be a list.

`constraint_param`
If pseudohuber centering is used (this is the default), parameter controlling relative weighting of elements closer and further from center. (Limit as `constraint_param` approaches infinity is the mean; as this parameter approaches zero, the minimizer of the pseudo-Huber loss approaches the median.)

run_score_tests
: logical: perform robust score testing?

null_fit_alg
: Which null fitting algorithm to use for score tests: "constraint_sandwich" or "augmented_lagrangian", or "discrete" when design matrix only includes categorical covariates. Default and recommended approach is "constraint_sandwich" or "discrete" when approach, unless J < 20.

Value

returns objects Y, X, cluster, and B_null_list, which may be modified by tests, and throw any useful errors, warnings, or messages.

---

emuFit_micro *Fit radEmu model. Called by emuFit*

Description

Fit radEmu model. Called by emuFit

Usage

```
emuFit_micro(
  X,
  Y,
  B = NULL,
  constraint_fn = NULL,
  maxit = 250,
  tolerance = 1e-05,
  verbose = TRUE,
  warm_start = TRUE,
  c1 = 1e-04,
  max_stepsize = 0.5,
  max_abs_B = 50,
  use_working_constraint = TRUE,
  j_ref = NULL,
  optimize_rows = TRUE,
  use_discrete = TRUE
)
```

Arguments

X
: a p x J design matrix

Y
: an n x p matrix of nonnegative observations

B
: starting value of coefficient matrix (p x J)

constraint_fn
: function g defining constraint on rows of B; g(B_k) = 0 for rows k = 1, ..., p of B.

maxit	maximum number of coordinate descent cycles to perform before exiting optimization
tolerance	tolerance on improvement in log likelihood at which to exit optimization
verbose	logical: print information about optimization progress? Default is TRUE.
warm_start	logical: begin from "warm start" obtained from linear regression on transformed counts? Default is TRUE.
c1	numeric: value of constant in Armijo condition checked in backtracking line search
max_stepsize	numeric: maximum sup-norm value of proposed step. Default is 0.5.
max_abs_B	numeric: maximum value elements of B are allowed to take in absolute value. Helps prevent optimization failure in larger problems. Defaults to 50.
use_working_constraint	logical: set a column of B equal to zero within optimization. Default is TRUE.
j_ref	If use_working_constraint is TRUE, column index of column of B to set to zero. Default is NULL, in which case this column is chosen to maximize the number of nonzero entries of Y_j_ref.
optimize_rows	If use_working_constraint is TRUE, update overall location of rows of B relative to column constrained to equal zero under working constraint before iterating through updates to columns of B individually. Default is TRUE.
use_discrete	If discrete design matrix, use fast discrete implementation.

Value

A p x J matrix containing regression coefficients (under constraint g(B_k) = 0)

`emuFit_micro_penalized` *Fit radEmu model with Firth penalty*

Description

Fit radEmu model with Firth penalty

Usage

```
emuFit_micro_penalized(
  X,
  Y,
  B = NULL,
  X_cup = NULL,
  constraint_fn = NULL,
  maxit = 500,
  ml_maxit = 5,
  tolerance = 0.001,
```

```
  max_step = 5,
  verbose = TRUE,
  max_abs_B = 250,
  j_ref = NULL,
  use_discrete = TRUE
)
```

Arguments

X	a p x J design matrix
Y	an n x p matrix of nonnegative observations
B	starting value of coefficient matrix (p x J)
X_cup	design matrix for Y in long format. Defaults to NULL, in which case matrix is computed from X.
constraint_fn	function g defining constraint on rows of B; g(B_k) = 0 for rows k = 1, ..., p of B.
maxit	maximum number of coordinate descent cycles to perform before exiting optimization
ml_maxit	numeric: maximum number of coordinate descent cycles to perform inside of maximum likelihood fits. Defaults to 5.
tolerance	tolerance on improvement in log likelihood at which to exit optimization
max_step	numeric: maximum sup-norm for proposed update steps
verbose	logical: report information about progress of optimization? Default is TRUE.
max_abs_B	numeric: maximum allowed value for elements of B (in absolute value). In most cases this is not needed as Firth penalty will prevent infinite estimates under separation. However, such a threshold may be helpful in very poorly conditioned problems (e.g., with many nearly collinear regressors). Default is 50.
j_ref	which column of B to set to zero as a convenience identifiability during optimization. Default is NULL, in which case this column is chosen based on characteristics of Y (i.e., j_ref chosen to maximize number of entries of Y_j_ref greater than zero).
use_discrete	logical: if the design matrix is discrete (it has exactly p distinct rows), compute the penalized estimate and data augmentations in closed form rather than iteratively. Default is TRUE.

Value

A p x J matrix containing regression coefficients (under constraint g(B_k) = 0)

fit_null *fits model with B_kj constrained to equal g(B_k) for constraint fn g*

Description

fits model with B_kj constrained to equal g(B_k) for constraint fn g

Usage

```
fit_null(
  B,
  Y,
  X,
  X_cup = NULL,
  k_constr,
  j_constr,
  j_ref,
  constraint_fn,
  constraint_grad_fn,
  rho_init = 1,
  tau = 1.2,
  kappa = 0.8,
  B_tol = 0.01,
  inner_tol = 0.01,
  constraint_tol = 1e-04,
  max_step = 5,
  c1 = 1e-04,
  maxit = 1000,
  inner_maxit = 25,
  verbose = FALSE,
  trackB = FALSE,
  ignore_stop = FALSE,
  null_diagnostic_plots = FALSE
)
```

Arguments

B	description
Y	Y (with augmentations)
X	design matrix
X_cup	design matrix for Y in long format. Defaults to NULL, in which case matrix is computed from X.
k_constr	row index of B to constrain
j_constr	col index of B to constrain
j_ref	column index of convenience constraint

constraint_fn	constraint function
constraint_grad_fn	gradient of constraint fn
rho_init	where to start quadratic penalty parameter
tau	how much to increment rho by each iteration
kappa	cutoff above which to increment rho. If distance to feasibility doesn't shrink by at least this factor in an iteration, increment rho by tau.
B_tol	tolerance for convergence in $max_{k,j}\|B^t_{kj} - B^{(t-1)}_{kj}\|$
inner_tol	tolerance for inner loop
constraint_tol	tolerance for $\|B_k j - g(B_k)\|$
max_step	maximum step size
c1	constant for armijo rule
maxit	maximum iterations
inner_maxit	max iterations per inner loop
verbose	shout at you?
trackB	track value of beta across iterations and return?
ignore_stop	whether to ignore stopping criteria and run maxit iterations (could be helpful for diagnostic plots).
null_diagnostic_plots	logical: should diagnostic plots be made for estimation under the null hypothesis? Default is FALSE.

Value

A list containing elements B, k_constr, j_constr, niter gap, u, rho, and Bs. B is a matrix containing parameter estimates under the null (obtained by maximum likelihood on augmented observations Y), k_constr, and j_constr give row and column indexes of the parameter fixed to be equal to the constraint function $g()$ under the null. niter is a scalar giving total number of outer iterations used to fit the null model, gap gives the final value of $g(B_{kconstr}) - B_{kconstr,jconstr}$, u and rho are final values of augmented Lagrangian parameters, and Bs is a data frame containing values of B by iteration if trackB was set equal to TRUE (otherwise it contains a NULL value).

---

fit_null_discrete *fits model with B_kj constrained to equal g(B_k) for constraint fn g, for a symmetric constraint with a discrete design*

Description

fits model with B_kj constrained to equal g(B_k) for constraint fn g, for a symmetric constraint with a discrete design

Usage

```
fit_null_discrete(
  Y,
  X,
  k_constr,
  j_constr,
  j_ref,
  trackB = FALSE,
  maxit = 5000,
  tol = 0.1,
  verbose = FALSE,
  ls_max = 20,
  ls_rho = 0.5,
  max_step = 1,
  constraint = "pseudohuber"
)
```

Arguments

Y	Y (with augmentations)
X	design matrix
k_constr	row index of B to constrain
j_constr	col index of B to constrain
j_ref	column index of convenience constraint
trackB	track value of beta across iterations and return? Default is FALSE.
maxit	maximum iterations. Default is 5000.
tol	tolerance for stopping criteria. Algorithm stops when the root mean of the norm of the score vector is less than the tolerance. Default is 0.01.
verbose	should the algorithm print updates for you? Default is FALSE.
ls_max	maximum number of iterations in the line search. Default is 20.
ls_rho	scaling factor in the line search. Default is 0.5.
max_step	step capping after the line sesarch. Default is 1.
constraint	What type of symmetric constraint do we have? Options are "mean" and "pseudohuber.

Value

A list containing elements B, k_constr, j_constr, niter and Bs. B is a matrix containing parameter estimates under the null (obtained by maximum likelihood on augmented observations Y), k_constr, and j_constr give row and column indexes of the parameter fixed to be equal to the constraint function $g()$ under the null. niter is a scalar giving total number of outer iterations used to fit the null model, and Bs is a data frame containing values of B by iteration if trackB was set equal to TRUE (otherwise it contains a NULL value).

`fit_null_scc` *fits model with B_kj constrained to equal g(B_k) for constraint fn g, for a single category constraint*

Description

fits model with B_kj constrained to equal g(B_k) for constraint fn g, for a single category constraint

Usage

```
fit_null_scc(
  B,
  Y,
  X,
  X_cup = NULL,
  k_constr,
  j_constr,
  j_ref,
  constraint_fn,
  constraint_grad_fn,
  rho_init = 1,
  tau = 1.2,
  kappa = 0.8,
  B_tol = 0.01,
  inner_tol = 0.01,
  constraint_tol = 1e-04,
  max_step = 5,
  c1 = 1e-04,
  maxit = 1000,
  inner_maxit = 25,
  verbose = FALSE,
  trackB = FALSE
)
```

Arguments

`B`	description
`Y`	Y (with augmentations)
`X`	design matrix
`X_cup`	design matrix for Y in long format. Defaults to NULL, in which case matrix is computed from X.
`k_constr`	row index of B to constrain
`j_constr`	col index of B to constrain
`j_ref`	column index of convenience constraint
`constraint_fn`	constraint function

constraint_grad_fn	gradient of constraint fn
rho_init	where to start quadratic penalty parameter
tau	how much to increment rho by each iteration
kappa	cutoff above which to increment rho. If distance to feasibility doesn't shrink by at least this factor in an iteration, increment rho by tau.
B_tol	tolerance for convergence in $max_{k,j}\|B_{kj}^{t} - B_{kj}^{(t-1)}\|$
inner_tol	tolerance for inner loop
constraint_tol	tolerance for $\|B_k j - g(B_k)\|$
max_step	maximum step size
c1	constant for armijo rule
maxit	maximum iterations
inner_maxit	max iterations per inner loop
verbose	shout at you?
trackB	track value of beta across iterations and return?

Value

A list containing elements B, k_constr, j_constr, niter gap, u, rho, and Bs. B is a matrix containing parameter estimates under the null (obtained by maximum likelihood on augmented observations Y), k_constr, and j_constr give row and column indexes of the parameter fixed to be equal to the constraint function $g()$ under the null. niter is a scalar giving total number of outer iterations used to fit the null model, gap gives the final value of $g(B_{kconstr}) - B_{kconstr,jconstr}$, u and rho are final values of augmented Lagrangian parameters, and Bs is a data frame containing values of B by iteration if trackB was set equal to TRUE (otherwise it contains a NULL value). - update based on new algorithm

---

fit_null_symmetric *fits model with B_kj constrained to equal g(B_k) for constraint fn g, for a symmetric constraint*

Description

fits model with B_kj constrained to equal g(B_k) for constraint fn g, for a symmetric constraint

Usage

```
fit_null_symmetric(
  B,
  Y,
  X,
  X_cup = NULL,
  k_constr,
```

```
  j_constr,
  j_ref,
  constraint_fn,
  constraint_grad_fn,
  inner_tol = 0.01,
  c1 = 0.01,
  maxit = 1000,
  inner_maxit = 25,
  verbose = FALSE,
  trackB = FALSE,
  use_optim = FALSE,
  ignore_stop = FALSE,
  tol_lik = 1e-05,
  tol_test_stat = 0.01,
  null_window = 5,
  max_step = 1
)
```

Arguments

B	description
Y	Y (with augmentations)
X	design matrix
X_cup	design matrix for Y in long format. Defaults to NULL, in which case matrix is computed from X.
k_constr	row index of B to constrain
j_constr	col index of B to constrain
j_ref	column index of convenience constraint
constraint_fn	constraint function
constraint_grad_fn	gradient of constraint fn
inner_tol	tolerance for inner loop
c1	constant for armijo rule
maxit	maximum iterations
inner_maxit	max iterations per inner loop
verbose	shout at you?
trackB	track value of beta across iterations and return?
use_optim	whether to use `optim` instead of fisher scoring. Default is FALSE.
ignore_stop	whether to ignore stopping criteria and run `maxit` iterations (could be helpful for diagnostic plots).
tol_lik	tolerance for relative changes in likelihood for stopping criteria. Default is `1e-5`.
tol_test_stat	tolerance for relative changes in test statistic for stopping criteria. Default is `0.01`.
null_window	window to use for stopping criteria (this many iterations where stopping criteria is met). Default is `5`.
max_step	Default is `1`

Value

A list containing elements B, k_constr, j_constr, niter and Bs. B is a matrix containing parameter estimates under the null (obtained by maximum likelihood on augmented observations Y), k_constr, and j_constr give row and column indexes of the parameter fixed to be equal to the constraint function $g()$ under the null. niter is a scalar giving total number of outer iterations used to fit the null model, and Bs is a data frame containing values of B by iteration if trackB was set equal to TRUE (otherwise it contains a NULL value). - update based on new algorithm

---

`fit_null_symmetric_subset` *fits model with B_kj constrained to equal g(B_k) for constraint fn g, for a symmetric constraint*

Description

fits model with B_kj constrained to equal g(B_k) for constraint fn g, for a symmetric constraint

Usage

```
fit_null_symmetric_subset(
  B,
  Y,
  X,
  X_cup = NULL,
  k_constr,
  j_constr,
  j_ref,
  constraint_fn,
  constraint_grad_fn,
  inner_tol = 0.01,
  c1 = 0.01,
  maxit = 1000,
  inner_maxit = 25,
  verbose = FALSE,
  trackB = FALSE,
  use_optim = FALSE,
  ignore_stop = FALSE,
  tol_lik = 1e-05,
  tol_test_stat = 0.01,
  null_window = 5,
  max_step = 1,
  reference_set = NULL
)
```

Arguments

B	description
Y	Y (with augmentations)
X	design matrix
X_cup	design matrix for Y in long format. Defaults to NULL, in which case matrix is computed from X.
k_constr	row index of B to constrain
j_constr	col index of B to constrain
j_ref	column index of convenience constraint
constraint_fn	constraint function
constraint_grad_fn	gradient of constraint fn
inner_tol	tolerance for inner loop
c1	constant for armijo rule
maxit	maximum iterations
inner_maxit	max iterations per inner loop
verbose	shout at you?
trackB	track value of beta across iterations and return?
use_optim	whether to use optim instead of fisher scoring. Default is FALSE.
ignore_stop	whether to ignore stopping criteria and run maxit iterations (could be helpful for diagnostic plots).
tol_lik	tolerance for relative changes in likelihood for stopping criteria. Default is 1e-5.
tol_test_stat	tolerance for relative changes in test statistic for stopping criteria. Default is 0.01.
null_window	window to use for stopping criteria (this many iterations where stopping criteria is met). Default is 5.
max_step	Default is 1.
reference_set	Optional reference set to act as the subset. Default is NULL in which case the function will look into constraint_fn for the reference set.

Value

A list containing elements B, k_constr, j_constr, niter and Bs. B is a matrix containing parameter estimates under the null (obtained by maximum likelihood on augmented observations Y), k_constr, and j_constr give row and column indexes of the parameter fixed to be equal to the constraint function $g()$ under the null. niter is a scalar giving total number of outer iterations used to fit the null model, and Bs is a data frame containing values of B by iteration if trackB was set equal to TRUE (otherwise it contains a NULL value). - update based on new algorithm

make_design_matrix *Generates the design matrix for a given* formula *and* data *object*

Description

Generates the design matrix for a given formula and data object

Usage

```
make_design_matrix(Y = NULL, formula, data = NULL, assay_name = NULL)
```

Arguments

Y	Optionally, a phyloseq or TreeSummarizedExperiment object containing an otu table and sample data.
formula	a one-sided formula specifying the form of the mean model to be fit
data	an n x p data frame containing variables given in formula (required unless Y is included as a phyloseq or TreeSummarizedExperiment object)
assay_name	Optionally, a string containing the desired assay name within a TreeSummarizedExperiment object. This is only required if data is a TreeSummarizedExperiment object, otherwise this argument does nothing and can be ignored.

Value

returns design matrix X.

make_diagnostics_plots *makes plots to investigate convergence of estimation under null hypothesis*

Description

makes plots to investigate convergence of estimation under null hypothesis

Usage

```
make_diagnostics_plots(diagnostic_df)
```

Arguments

diagnostic_df	Dataframe with relevant information

Value

Plots with diagnostic information. If the fisher scoring algorithm has been fit, this will be two plots showing how the log likelihood and test statistic change over iterations. If the augmented lagrangian algorithm has been fit, this will also include how the constraint gap and maximum changing element of B change over iterations.

`make_reference_constraints` *Make lists of constraint functions and their gradients when using a reference taxon*

Description

Make lists of constraint functions and their gradients when using a reference taxon

Usage

```
make_reference_constraints(p, j)
```

Arguments

`p`	The number of columns in the design matrix `X`. If you don't know the number of columns, you can find it with `ncol(radEmu::make_design_matrix(formula = your_formula, data = your_data))`. `your_formula` should be the expression you give to `emuFit`'s `formula` argument.
`j`	A single value or a vector of length `p - 1` where `p` is the number of columns in the design matrix `X`. If a single value, `j` will be used as the reference category for all covariates. If a vector of values, `j[k]` will be used as the reference category for the covariate in design matrix column `k + 1`.

Value

A list with elements `constraints_list` and `constraints_grad_list`. The `constraints_list` is a list of constraint functions for each column `p` of the design matrix. By default, the constraint for the intercept is the pseudo Huber median. The constraints for covariates are determined by reference categories given by the argument `j`. The `constraints_grad_list` is a list of gradients of each constraint function.

Examples

```
# two columns in design matrix, reference taxon is taxon 5
list1 <- make_reference_constraints(p = 2, j = 5)

# four columns in design matrix, reference taxon for all covariates is taxon 2
list2 <- make_reference_constraints(p = 4, j = 2)

# four columns in design matrix, reference taxon for covariates 1 and 2 is taxon 3 and
```

```
# reference taxon for covariate 3 is taxon 4
list3 <- make_reference_constraints(p = 4, j = c(3, 3, 4))
```

---

plot.emuFit *Plotting function*

Description

Plotting function

Usage

```
## S3 method for class 'emuFit'
plot(
  x,
  plot_key = NULL,
  title = NULL,
  taxon_names = NULL,
  display_taxon_names = TRUE,
  data_only = FALSE,
  ...
)
```

Arguments

x	Output from emuFit()
plot_key	(Optional) Default NULL. List of named vectors containing names in the "covariate" column of the coef output of the radEmu model object. If you wish for multiple covariate values to be plotted on the same plot, then those variables should be included in the same named vector. By default, each column of the design matrix receives its own plot.
title	(Optional). Default NULL. Character string. The main title for the graphic.
taxon_names	(Optional). Default NULL. Data frame. If NULL, keep taxon names as listed in radEmu model. Otherwise, users can input a data frame with two columns: one labelled "category" with the same levels as in the radEmu output and another labelled "cat_small" with the preferred labels.
display_taxon_names	(Optional). Default TRUE. Boolean. If FALSE, remove sample names from the plot.
data_only	(Optional). Default FALSE. Boolean. If TRUE, only returns data frame.
...	There are no optional parameters at this time.

Value

Object of class ggplot. Plot of radEmu model fit with 95% confidence intervals.

Examples

```
data(wirbel_sample_small)
data(wirbel_otu_small)

ch_fit <- emuFit(formula = ~ Group + Study,
                 data = wirbel_sample_small,
                 Y = wirbel_otu_small,
                 run_score_tests = FALSE,
                 tolerance = 0.01)

plot_key <- list(p1 = c("Control" = "GroupCTR"),
                 p2 = c("FR-Control" = "StudyFR-CRC",
                        "US-Control" = "StudyUS-CRC"))

out <- plot(x = ch_fit,
            plot_key = plot_key,
            display_taxon_names = FALSE)

out$plots$p1
out$plots$p2
```

`power_model` *A fitted GLM model for the power of radEmu to reject a truly false null hypothesis*

Description

100 simulations were drawn with n = 10, 30, 50 and 100 total samples; J = 250 and 500 taxa; a single categorical covariate (p = 2); effect sizes beta11 (the parameter of interest) from 0, 0.5, 1, ... 2.5. beta0's range from -3 to 3, and the other beta1's range from -1 to 1 with no correlation between the beta0s (which, roughly speaking, control the relative abundance of the taxa) and the beta1s (which control the difference in abundance between the two covariate groups). Counts were drawn from a zero-inflated negative binomial model with size parameter 5, zero-inflation probability of 0.5 and average z's around log(50). A model for the probability of rejecting the null hypothesis of beta11 = 0 was fit. Model fitting was guided by plotting the log odds of rejection, where effect modification between n and beta11 was observed. This model may be useful for power calculations in future, though as with any simulation, its generalizability is limited to similar data generating processes. Simulation code that can be generalized is available at https://github.com/statdivlab/radEmu_supplementary under fig-power/power_simulations.R

Usage

```
power_model
```

Format

A GLM object.

power_model A GLM object modelling the odds of rejecting the null hypothesis at a given sample size, number of taxa, and effect size

References

Wirbel, J et al. (2023). *Meta-analysis of fecal metagenomes reveals global microbial signatures that are specific for colorectal cancer*. Nature Medicine, 25, 679–689. <doi: 10.1038/s41591-019-0406-6>.

`print.emuFit` *Print function*

Description

Print function

Usage

```
## S3 method for class 'emuFit'
print(x, n = 20, ...)
```

Arguments

`x`	Object of class `emuFit`
`n`	The number of coefficient estimates to be printed (ordered by largest absolute value to smallest)
`...`	No optional arguments are accepted at this time.

Value

`NULL`. Displays printed model summary.

`pseudohuber_median` *Calculate the pseudo-Huber smoothed median*

Description

Calculate the pseudo-Huber smoothed median using a quadratic approximation to the pseudo-Huber criterion detailed in supplement to Clausen et al. (2024).

Usage

```
pseudohuber_median(x, d = 0.1, tolerance = 1e-08, na.rm = FALSE)

psuedohuber_median(x, d = 0.1, tolerance = 1e-08, na.rm = FALSE)
```

Arguments

x	A vector to calculate the pseudo-Huber smoothed median for.
d	Smoothing parameter, by default set to 0.1. As d approaches 0 this function approaches the median and as d approaches infinity this function approaches the mean.
tolerance	Tolerance used to determine convergence in the algorithm used to calculate this function value.
na.rm	Default is FALSE, if FALSE then when x includes at least one NA value then NA is returned, if TRUE then when x includes at least one NA value then that value is removed and the pseudo-Huber median is computed without it.

Value

The calculated pseudo-Huber smoothed median over x with smoothing parameter d.

Examples

```
pseudohuber_median(x = rnorm(10), d = 0.1)
```

---

score_test *Run robust score test*

Description

Run robust score test

Usage

```
score_test(
  B,
  Y,
  X,
  X_cup,
  k_constr,
  j_constr,
  constraint_fn,
  constraint_grad_fn,
  rho_init = 1,
  tau = 2,
  kappa = 0.8,
  B_tol = 0.001,
  inner_tol = 0.01,
  constraint_tol = 0.001,
  j_ref,
  c1 = 1e-04,
```

```
  maxit = 1000,
  inner_maxit = 25,
  ntries = 4,
  verbose = FALSE,
  trackB = FALSE,
  I_inv = NULL,
  Dy = NULL,
  return_both_score_pvals = FALSE,
  cluster = NULL,
  null_diagnostic_plots = FALSE,
  ignore_stop = FALSE,
  tol_lik = 1e-05,
  tol_test_stat = 0.01,
  tol_discrete = 0.01,
  null_window = 5
)
```

Arguments

B	value of coefficient matrix (p x J) returned by full model fit or value of coefficient matrix to start null estimation at given as input to emuFit
Y	an n x J matrix or dataframe of *augmented* nonnegative observations (i.e., observations Y plus augmentations from last iteration of maximum penalized likelihood estimation for full model)
X	an n x p matrix of covariates
X_cup	the design matrix for long format Y in long format B (nJ x pJ)
k_constr	row index of element of B to be tested for equality to row identifiability constraint
j_constr	column index of element of B to be tested for equality to row identifiability constraint
constraint_fn	function g defining a constraint on rows of B; g(B_k) = 0 for rows k = 1, ..., p of B. Default function is a smoothed median (minimizer of pseudohuber loss).
constraint_grad_fn	function returning gradient of constraint function (as a function of a row of B)
rho_init	numeric: value at which to initiate rho parameter in augmented Lagrangian algorithm. Default is 1.
tau	numeric: value to scale rho by in each iteration of augmented Lagrangian algorithm that does not move estimate toward zero sufficiently. Default is 2.
kappa	numeric: value between 0 and 1 that determines the cutoff on the ratio of current distance from feasibility over distance in last iteration triggering scaling of rho. If this ratio is above kappa, rho is scaled by tau to encourage estimate to move toward feasibility.
B_tol	numeric: convergence tolerance for null model fits for score testing (if max of absolute difference in B across outer iterations is below this threshold, we declare convergence). Default is 0.001.

`inner_tol` numeric: convergence tolerance for inner loop of null fitting algorithm (if max of absolute difference in B across inner iterations is below this threshold, we declare convergence). Default is 0.01.

`constraint_tol` numeric: constraint tolerance for fits under null hypotheses (tested element of B must be equal to constraint function to within this tolerance for a fit to be accepted as a solution to constrained optimization problem). Default is 1e-5.

`j_ref` column index of convenience constraint

`c1` numeric: parameter for Armijo line search. Default is 1e-4.

`maxit` maximum number of outer iterations of augmented lagrangian algorithm to perform before exiting optimization. Default is 1000.

`inner_maxit` maximum number of coordinate descent passes through columns of B to make within each outer iteration of augmented lagrangian algorithm before exiting inner loop

`ntries` numeric: total number of times to attempt optimization under null if optimization fails (optimization parameters will be tweaked in subsequent fits to attempt to avoid failure). Default is 4.

`verbose` provide updates as model is being fitted? Defaults to TRUE.

`trackB` store and return values of B at each iteration of optimization algorithm? Useful for debugging. Default is FALSE.

`I_inv` Optional: matrix containing inverted information matrix computed under full model. Default is NULL, in which case information is recomputed under null, which we recommend.

`Dy` Optional: matrix containing empirical score covariance computed under full model. Default is NULL, in which case this quantity is recomputed under null, which we recommend.

`return_both_score_pvals`
logical: should score p-values be returned using both information matrix computed from full model fit and from null model fits? Default is FALSE. This parameter is used for simulations - in any applied analysis, type of p-value to be used should be chosen before conducting tests.

`cluster` a numeric vector giving cluster membership for each row of Y to be used in computing GEE test statistics. Default is NULL, in which case rows of Y are treated as independent.

`null_diagnostic_plots`
logical: should diagnostic plots be made for estimation under the null hypothesis? Default is `FALSE`.

`ignore_stop` whether to ignore stopping criteria and run `maxit` iterations (could be helpful for diagnostic plots).

`tol_lik` tolerance for relative changes in likelihood for stopping criteria. Default is `1e-5`.

`tol_test_stat` tolerance for relative changes in test statistic for stopping criteria. Default is `0.01`.

`tol_discrete` tolerance for root mean norm of score vector for stopping criteria. Default is `0.01`.

`null_window` window to use for stopping criteria (this many iterations where stopping criteria is met). Default is 5.

Value

A list containing elements score_stat, pval, log_pval,'niter, convergence, gap, u, rho, tau, inner_maxit, null_B, and Bs. score_statgives the value of the robust score statistic for $H_0: B_{k_constr,j_ = TRUE, Bs‘ is a data frame containing values of B by outcome category, covariate, and iteration.

---

simulate_data *Data simulation function*

Description

Function to simulate data for simulations in Clausen & Willis (2024) and for the cluster vignette

Usage

```
simulate_data(
  n,
  J,
  b0 = NULL,
  b1 = NULL,
  distn,
  zinb_size = NULL,
  zinb_zero_prop = NULL,
  mean_z,
  X = NULL,
  B = NULL,
  cluster = NULL
)
```

Arguments

n	Number of samples
J	Number of categories
b0	Intercept parameter vector
b1	Covariate paramter vector
distn	Distribution to simulate from, either "Poisson" or "ZINB"
zinb_size	Size parameter for negative binomial draw for ZINB data
zinb_zero_prop	Proportion of zeros for ZINB data
mean_z	Parameter controlling the mean of the sample-specific effects.
X	Optional design matrix, this must have two columns and n rows.
B	Optional B matrix, if p is not equal to 2
cluster	Optional cluster vector, this must have n elements.

Value

Y. A n times J dimension matrix of simulated response counts.

wirbel_otu — *mOTU table from Wirbel et al. meta-analysis of fecal metagenomes.*

Description

A matrix with counts of mOTUs (metagenome-derived OTUs) for each sample.

Usage

```
wirbel_otu
```

Format

A matrix with mOTU counts.

otu_table OTU table with 566 samples and 845 taxa

References

Wirbel, J et al. (2019). *Meta-analysis of fecal metagenomes reveals global microbial signatures that are specific for colorectal cancer*. Nature Medicine, 25, 679–689. <doi: 10.1038/s41591-019-0406-6>.

wirbel_otu_small — *subset of mOTU table from Wirbel et al. meta-analysis of fecal metagenomes.*

Description

A matrix with counts of mOTUs (metagenome-derived OTUs) for each sample.

Usage

```
wirbel_otu_small
```

Format

A matrix with mOTU counts.

otu_table OTU table with 320 samples and 47 taxa

References

Wirbel, J et al. (2019). *Meta-analysis of fecal metagenomes reveals global microbial signatures that are specific for colorectal cancer*. Nature Medicine, 25, 679–689. <doi: 10.1038/s41591-019-0406-6>.

`wirbel_sample` *Covariates from Wirbel et al. meta-analysis of fecal metagenomes.*

Description

A data frame with covariates.

Usage

```
wirbel_sample
```

Format

A data frame object with sample data, 566 observations of 14 covariates.

Sample_ID id of specific sample

External_ID id of specific sample from original study

Age age of subject from which sample was taken

Gender gender of subject from which sample was taken

BMI BMI of subject from which sample was taken

Country country of study

Study study that sample is from

Group CRC for colorectal cancer or CTR for control

Library_Size library size of sample

Age_spline.1 value of first coordinate of age spline

Age_spline.2 value of second coordinate of age spline

BMI_spline.1 value of first coordinate of BMI spline

BMI_spline.2 value of second coordinate of age spline

Sampling whether sampling happened before or after colonoscopy

References

Wirbel, J et al. (2019). *Meta-analysis of fecal metagenomes reveals global microbial signatures that are specific for colorectal cancer*. Nature Medicine, 25, 679–689. <doi: 10.1038/s41591-019-0406-6>.

`wirbel_sample_small` *Covariates from Wirbel et al. meta-analysis of fecal metagenomes, for a subset of samples.*

Description

A data frame with covariates.

Usage

```
wirbel_sample_small
```

Format

A data frame object with sample data, 566 observations of 14 covariates.

Sample_ID id of specific sample

External_ID id of specific sample from original study

Age age of subject from which sample was taken

Gender gender of subject from which sample was taken

BMI BMI of subject from which sample was taken

Country country of study

Study study that sample is from

Group CRC for colorectal cancer or CTR for control

Library_Size library size of sample

Age_spline.1 value of first coordinate of age spline

Age_spline.2 value of second coordinate of age spline

BMI_spline.1 value of first coordinate of BMI spline

BMI_spline.2 value of second coordinate of age spline

Sampling whether sampling happened before or after colonoscopy

References

Wirbel, J et al. (2019). *Meta-analysis of fecal metagenomes reveals global microbial signatures that are specific for colorectal cancer*. Nature Medicine, 25, 679–689. <doi: 10.1038/s41591-019-0406-6>.

`wirbel_taxonomy` *Taxonomy information for mOTUs from Wirbel et al. meta analysis.*

Description

A matrix with taxonomy information.

Usage

```
wirbel_taxonomy
```

Format

A matrix with taxonomy information.

tax_table taxonomy table

References

Wirbel, J et al. (2019). *Meta-analysis of fecal metagenomes reveals global microbial signatures that are specific for colorectal cancer*. Nature Medicine, 25, 679–689. <doi: 10.1038/s41591-019-0406-6>.

Index

∗ **datasets**
power_model, 26
wirbel_otu, 32
wirbel_otu_small, 32
wirbel_sample, 33
wirbel_sample_small, 34
wirbel_taxonomy, 35

control_fn, 2

dpseudohuber_median_dx, 4
dpsuedohuber_median_dx
*(*dpseudohuber_median_dx*)*, 4

emuFit, 5
emuFit_check, 10
emuFit_micro, 12
emuFit_micro_penalized, 13

fit_null, 15
fit_null_discrete, 16
fit_null_scc, 18
fit_null_symmetric, 19
fit_null_symmetric_subset, 21

make_design_matrix, 23
make_diagnostics_plots, 23
make_reference_constraints, 24

plot.emuFit, 25
power_model, 26
print.emuFit, 27
pseudohuber_median, 27
psuedohuber_median
*(*pseudohuber_median*)*, 27

score_test, 28
simulate_data, 31

wirbel_otu, 32
wirbel_otu_small, 32
wirbel_sample, 33
wirbel_sample_small, 34
wirbel_taxonomy, 35