Package ‘breakaway’

March 9, 2022

Title Species Richness Estimation and Modeling

Version 4.7.9

Date 2022-02-09

Description Understanding the drivers of microbial diversity is an important frontier of microbial ecology, and investigating the diversity of samples from microbial ecosystems is a common step in any microbiome analysis. 'breakaway' is the premier package for statistical analysis of microbial diversity. 'breakaway' implements the latest and greatest estimates of species richness, as well as the most commonly used estimates. Methods uniquely available in this package include objective Bayes estimators described in Barger and Bunge (2010) <doi:10.1214/10-BA527>, frequency-ratio-based estimators described in Willis and Bunge (2015) <doi:10.1111/biom.12332>, and as described in Willis, Whitman, and Bunge (2016) <doi:10.1111/rssc.12206>, a linear modeling approach for detecting changes in diversity.

License GPL-2

BugReports `https://github.com/adw96/breakaway/issues`

LazyData true

RoxygenNote 7.1.2

Depends R (>= 3.5.0)

Imports ggplot2, graphics, lme4, magrittr, MASS, phyloseq, stats, tibble, utils

Suggests corncob, covr, devtools, dplyr, DT, knitr, openxlsx, plyr, RCurl, reshape2, R.rsp, remotes, rmarkdown, testthat, tidyverse

VignetteBuilder knitr

URL `https://adw96.github.io/breakaway/`

NeedsCompilation no

Author Amy Willis [aut, cre] (<`https://orcid.org/0000-0002-2802-4317`>),
Bryan D Martin [aut] (<`https://orcid.org/0000-0002-8002-5296`>),
Pauline Trinh [aut],
Sarah Teichman [aut],
David Clausen [aut],
Kathryn Barger [aut],
John Bunge [aut]

Maintainer Amy Willis <`adwillis@uw.edu`>

Repository CRAN

Date/Publication 2022-03-09 08:00:09 UTC

R topics documented:

`alpha_estimate` *alpha_estimate*

Description

Build objects of class alpha_estimate from their components. `alpha_estimate()` is a constructor method

Usage

```
alpha_estimate(
  estimate = NULL,
  error = NULL,
  estimand = NULL,
  name = NULL,
  interval = NULL,
  interval_type = NULL,
  type = NULL,
  model = NULL,
  warnings = NULL,
  frequentist = NULL,
  parametric = NULL,
  plot = NULL,
  reasonable = NULL,
  other = NULL,
  ...
)
```

Arguments

`estimate`	The estimate
`error`	The standard error in the estimate
`estimand`	What is the estimate trying to estimate? (richness, Shannon...)
`name`	The name of the method
`interval`	An interval estimate
`interval_type`	Type of interval estimate
`type`	TODO(Amy): Deprecate?
`model`	What model is fit
`warnings`	Any warnings?
`frequentist`	Logical. Frequentist or Bayesian?
`parametric`	Logical. Parametric or not?
`plot`	A ggplot associated with the estimate
`reasonable`	Is the estimate likely to be reasonable?
`other`	Any other relevant objects
`...`	Any other objects

Value

An object of class alpha_estimate

`alpha_estimates` *alpha_estimates*

Description

Build objects of class alpha_estimates from their components. `alpha_estimates()` is a constructor method

Usage

```
alpha_estimates(...)
```

Arguments

`...`	Objects of class alpha_estimate, or a list of objects of class alpha_estimate

Value

An object of class alpha_estimates

`apples` *(Data) Frequency count table of soil microbes in an apples orchard.*

Description

(Data) Frequency count table of soil microbes in an apples orchard.

Usage

```
apples
```

Format

A data frame with 88 rows and 2 variables:

index an index variable

frequency number of taxa that were observed with this frequency ...

Source

https://onlinelibrary.wiley.com/doi/abs/10.1111/biom.12332

References

Willis, A. and Bunge, J. (2015). Estimating diversity via frequency ratios. *Biometrics*, **71**(4), 1042–1049. https://onlinelibrary.wiley.com/doi/abs/10.1111/biom.12332

Walsh, F. et al. (2014). (2014). Restricted streptomycin use in apple orchards did not adversely alter the soil bacteria communities. *Frontiers in Microbiology* **4**, 383.

`atacama` *A canonical QIIME2 dataset*

Description

Data used for internal testing retrieved from https://raw.githubusercontent.com/paulinetrinh/data/master/otu_table_atacama.tx

Usage

```
atacama
```

Format

character vector

`betta` *Modelling total diversity with betta*

Description

This function tests for heterogeneity of total diversity (observed plus unobserved) across multiple sites. It can account or test for fixed effects that may explain diversity. It returns the significance of the covariates in explaining diversity and a hypothesis test for heterogeneity.

Usage

```
betta(
  chats = NULL,
  ses,
  X = NULL,
  initial_est = NULL,
  formula = NULL,
  data = NULL
)
```

Arguments

`chats`	A vector of estimates of total diversity at different sampling locations. ‘`breakaway`’ estimates are suggested in the high-diversity case but not enforced.
`ses`	The standard errors in `chats`, the diversity estimates. This can either be a vector of standard errors (with the arguments `chats` and `X`), or the name of the variable in the dataframe `data` that contains the standard errors (with the arguments `formula` and `data`).
`X`	A numeric matrix of covariates. If not supplied, an intercept-only model will be fit. This is optional with the `chats` argument.
`initial_est`	(Optional) A vector of length 1 + `ncol`(X) giving the starting values for the likelihood maximisation search. The first element is the starting estimate for sigma^2_u, and the remaining elements are the starting elements for beta. Defaults to NULL, in which case the starting values outlined in the paper are used.
`formula`	A formula object of the form $y\ x|group$. Required with the `data` argument.
`data`	A dataframe containing the response, response standard errors, covariates, and grouping variable. Required with the `formula` argument.

Value

`table`	A coefficient table for the model parameters. The columns give the parameter estimates, standard errors, and p-values, respectively. This model is only as effective as your diversity estimation procedure; for this reason please confirm that your estimates are appropriate and that your model is not misspecified. betta_pic may be useful for this purpose.

`cov`	Estimated covariance matrix of the parameter estimates.
`ssq_u`	The estimate of the heterogeneity variance.
`homogeneity`	The test statistic and p-value for the test of homogeneity.
`global`	The test statistic and p-value for the test of model explanatory power.
`blups`	The conditional expected values of the diversity estimates (conditional on the random effects). The authors propose that if the practitioner believes that information from one diversity estimator may inform the others, then using the '`condfits`' as estimators of total diversity rather than '`Chats`' may reduce variance of diversity estimates by "sharing strength" across the samples.
`blupses`	The estimated standard deviation (standard errors) in the blups.
`loglikelihood`	The log likelihood of the fitted model.
`aic`	The Akaike information criterion for the fitted model.
`aicc`	The finite sample correction of the Akaike information criterion for the fitted model.
`r_squared_wls`	The weighted R^2 statistic, appropriate for heteroskedastic linear models.
`function.args`	A list containing values initially passed to betta_random.

Note

Ecologists who are interested in the way species richness varies with covariate information often run a regression-type analysis on the observed diversity using their covariate information as predictors. However, in many settings (especially microbial), rare and unobserved taxa play a hugely important role in explaining the subtleties of the ecosystem, however, a regression analysis on the observed diversity level fails to account for these unobserved taxa. By predicting the total level of diversity (for example, via `breakaway`) and estimating the standard error in the estimate, one can take account of these unobserved, but important, taxa. In order to account for the estimated nature of the response, a mixed model approach is taken, whereby the varying levels of confidence in the estimates contributes to a diagonal but heteroscedastic covariance matrix. Given covariates constitute the fixed effects in the mixed model, and significance of the random effect term `sigsq_u` reflects heterogeneity in the sample, that is, variability that cannot be explained by only the covariates. The authors believe this to be the first attempt at modelling total diversity in a way that accounts for its estimated nature.

Author(s)

Amy Willis

References

Willis, A., Bunge, J., and Whitman, T. (2015). Inference for changes in biodiversity. *arXiv preprint.*

Willis, A. and Bunge, J. (2015). Estimating diversity via frequency ratios. *Biometrics.*

See Also

`breakaway`; `breakaway_nof1`; `apples`

Examples

```
df <- data.frame(chats = c(2000, 3000, 4000, 3000), ses = c(100, 200, 150, 180),
                 Cont_var = c(100, 150, 100, 50))

# formula notation
betta(formula = chats ~ Cont_var, ses = ses, data = df)

# direct input
betta(c(2000, 3000, 4000, 3000), c(100, 200, 150, 180), cbind(1, c(100, 150, 100,
    50)))

## handles missing data
betta(c(2000, 3000, 4000, 3000), c(100, 200, 150, NA))

## A test for heterogeneity of apples diversity estimates vs butterfly estimates
betta(c(1552, 1500, 884), c(305, 675, 205), cbind(1, c(0, 0, 1)))
```

`betta_lincom` *Confidence intervals and testing for linear combinations of fixed effects estimated via betta() or betta_random()*

Description

This function provides point estimates, standard errors, and equal-tailed confidence intervals for linear combinations of fixed effects estimated via betta() or betta_random(). A p-value for a Wald test of the null that the linear combination of effects is equal to zero (against a general alternative) is also returned.

Usage

```
betta_lincom(fitted_betta, linear_com, signif_cutoff = 0.05)
```

Arguments

`fitted_betta`	A fitted betta object – i.e., the output of either betta() or betta_random() – containing fixed effect estimates of interest.
`linear_com`	The linear combination of fixed effects for which a point estimate, confidence interval, and hypothesis test are to be produced.
`signif_cutoff`	The type-I significance threshold for confidence intervals. Defaults to 0.05.

Value

`table`	A table containing a point estimate, standard error, lower and upper confidence bounds, and a p-value for the linear combination of fixed effects specified in input. The p-value is generated via a two-sided Wald test of the null that the linear combination of fixed effects is equal to zero.

Author(s)

David Clausen

References

Willis, A., Bunge, J., and Whitman, T. (2015). Inference for changes in biodiversity. *arXiv preprint.*

See Also

`betta`;

Examples

```
# generate example data
df <- data.frame(chats = c(2000, 3000, 4000, 3000), ses = c(100, 200, 150, 180),
                 Cont_var = c(100, 150, 100, 50))

# fit betta()
example_fit <- betta(formula = chats ~ Cont_var, ses = ses, data = df)

# generate point estimate and 95% CI for mean richness at Cont_var = 125

betta_lincom(fitted_betta = example_fit,
linear_com = c(1, 125)) # this tells betta_lincom to estimate value of beta_0 + 125*beta_1,
# where beta_0 is the intercept, and beta_1 is the (true value of the) coefficient on Cont_var
```

`betta_pic` *function for plotting total diversity*

Description

A simple plotting interface for comparing total diversity across samples or a covariate gradient.

Usage

```
betta_pic(
  y,
  se,
  x = 1:length(y),
  ylimu = NULL,
  myy = NULL,
  mymain = NULL,
  mycol = NULL,
  labs = NULL,
  mypch = NULL,
  myxlim = NULL
)
```

Arguments

`y`	A vector of estimates of total diversity. Other parameter estimates are accessible; this method may be used for plotting any parameter estimates..
`se`	The standard errors in ‘y’, the diversity (or other parameter’s) estimates.
`x`	A vector of covariates to form the x-coordinates of the intervals. If no argument is given, defaults to the order.
`ylimu`	The upper endpoint of the y-axis.

`myy`	Deprecated, for backwards compatibility
`mymain`	Deprecated, for backwards compatibility
`mycol`	Deprecated, for backwards compatibility
`labs`	Deprecated, for backwards compatibility
`mypch`	Deprecated, for backwards compatibility
`myxlim`	Deprecated, for backwards compatibility

Value

A ggplot object

Author(s)

Amy Willis

References

Willis, A., Bunge, J., and Whitman, T. (2015). Inference for changes in biodiversity. arXiv preprint.

See Also

`betta`

Examples

```
betta_pic(c(1552, 1500, 884), c(305, 675, 205), mymain = "Example title")
```

`betta_random` *modelling total diversity with random effects*

Description

This function extends betta() to permit random effects modelling.

Usage

```
betta_random(
  chats = NULL,
  ses,
  X = NULL,
  groups = NULL,
  formula = NULL,
  data = NULL,
  n_initializations = 10
)
```

Arguments

`chats`	A vector of estimates of total diversity at different sampling locations. Required with the `groups` argument, and optionally with the `X` argument.
`ses`	The standard errors in `chats`, the diversity estimates. This can either be a vector of standard errors (with the arguments `chats` and `X`), or the name of the variable in the dataframe `data` that contains the standard errors (with the arguments `formula` and `data`).
`X`	A numeric matrix of covariates corresponding to fixed effects. If not supplied, an intercept-only model will be fit. Optional with the `chats` and `groups` arguments.
`groups`	A categorical variable representing the random-effects groups that each of the estimates belong to. Required with the `chats` argument and optionally with the `X` argument.
`formula`	A formula object of the form $y\ x|group$. Required with the `data` argument.
`data`	A dataframe containing the response, response standard errors, covariates, and grouping variable. Required with the `formula` argument.
`n_initializations`	A positive integer giving number of random initializations to be used in optimization (default 10)

Value

`table`	A coefficient table for the model parameters. The columns give the parameter estimates, standard errors, and p-values, respectively. This model is only as effective as your diversity estimation procedure; for this reason please confirm that your estimates are appropriate and that your model is not misspecified. betta_pic may be useful for this purpose.
`cov`	Estimated covariance matrix of the parameter estimates.
`ssq_u`	The estimate of the heterogeneity variance.
`ssq_g`	Estimates of within-group variance. The estimate will be zero for groups with only one observation.
`homogeneity`	The test statistic and p-value for the test of homogeneity.
`global`	The test statistic and p-value for the test of model explanatory power.
`blups`	The conditional expected values of the diversity estimates (conditional on the random effects). Estimates of variability for the random effects case are unavailable at this time; please contact the maintainer if needed.
`function.args`	A list containing values initially passed to betta_random.

Author(s)

Amy Willis

References

Willis, A., Bunge, J., and Whitman, T. (2015). Inference for changes in biodiversity. *arXiv preprint.*

See Also

betta;

Examples

```
df <- data.frame(chats = c(2000, 3000, 4000, 3000),
ses = c(100, 200, 150, 180),
Cont_var = c(100, 150, 100, 50),
groups = c("a", "a", "a", "b"))

# formula notation
betta_random(formula = chats ~ Cont_var| groups,
ses = ses,
 data = df)

# direct input
betta_random(c(2000, 3000, 4000, 3000), c(100, 200, 150, 180),
X = cbind(Int = 1, Cont_var = c(100, 150, 100, 50)),
groups = c("a", "a", "a", "b"))

## handles missing data
betta_random(c(2000, 3000, 4000, 3000), c(100, 200, 150, NA),
groups = c("a", NA,
    "b", "b"))
```

`breakaway` *Species richness estimation with breakaway*

Description

breakaway is a wrapper for modern species richness estimation for modern datasets

Usage

```
breakaway(
  input_data,
  cutoff = NA,
  output = NULL,
  plot = NULL,
  answers = NULL,
  print = NULL,
  ...
)
```

Arguments

`input_data`	An input type that can be processed by `convert()`
`cutoff`	The maximum frequency count to use for model fitting
`output`	Deprecated; only for backwards compatibility
`plot`	Deprecated; only for backwards compatibility
`answers`	Deprecated; only for backwards compatibility
`print`	Deprecated; only for backwards compatibility
`...`	Additional arguments will be ignored; this is for backward compatibility

Value

An object of class `alpha_estimate`

Note

‘`breakaway`’ presents an estimator of species richness that is well-suited to the high-diversity/microbial setting. However, many microbial datasets display more diversity than the Kemp-type models can permit. In this case, the log-transformed `WLRM` diversity estimator of Rocchetti et. al. (2011) is returned. The authors’ experience suggests that some datasets that require the log-transformed `WLRM` contain “false” diversity, that is, diversity attributable to sequencing errors (via an inflated singleton count). The authors encourage judicious use of diversity estimators when the dataset may contain these errors, and recommend the use of `breakaway_nof1` as an exploratory tool in this case.

Author(s)

Amy Willis

References

Willis, A. and Bunge, J. (2015). Estimating diversity via frequency ratios. *Biometrics*, **71**(4), 1042–1049.

See Also

`breakaway_nof1`; `kemp`; `apples`

Examples

```
breakaway(apples)
breakaway(apples, plot = FALSE, output = FALSE, answers = TRUE)
```

`breakaway_nof1` *species richness estimation without singletons*

Description

This function permits estimation of total diversity based on a sample frequency count table. Unlike `breakaway`, it does not require an input for the number of species observed once, making it an excellent exploratory tool for microbial ecologists who believe that their sample may contain spurious singletons. The underlying estimation procedure is similar to that of `breakaway` and is outlined in Willis & Bunge (2014). The diversity estimate, standard error, estimated model coefficients and plot of the fitted model are returned.

Usage

```
breakaway_nof1(
  input_data,
  output = NULL,
  plot = NULL,
  answers = NULL,
  print = NULL
)
```

Arguments

`input_data`	An input type that can be processed by `convert()`
`output`	Deprecated; only for backwards compatibility
`plot`	Deprecated; only for backwards compatibility
`answers`	Deprecated; only for backwards compatibility
`print`	Deprecated; only for backwards compatibility

Value

An object of class `alpha_estimate`

`code`	A category representing algorithm behaviour. ‘`code=1`’ indicates no nonlinear models converged and the transformed `WLRM` diversity estimate of Rocchetti et. al. (2011) is returned. ‘`code=2`’ indicates that the iteratively reweighted model converged and was returned. ‘`code=3`’ indicates that iterative reweighting did not converge but a model based on a simplified variance structure was returned (in this case, the variance of the frequency ratios is assumed to be proportional to the denominator frequency index). Please peruse your fitted model before using your diversity estimate.
`name`	The “name” of the selected model. The first integer represents the numerator polynomial degree and the second integer represents the denominator polynomial degree. See Willis & Bunge (2014) for details.
`para`	Estimated model parameters and standard errors.

`est`	The estimate of total (observed plus unobserved) diversity.
`seest`	The standard error in the diversity estimate.
`full`	The chosen nonlinear model for frequency ratios.

Note

It is common for microbial ecologists to believe that their dataset contains false diversity. This often arises because sequencing errors result in classifying already observed organisms as new organisms. ‘`breakaway_nof1`’ was developed as an exploratory tool in this case. Practitioners can run ‘`breakaway`’ on their dataset including the singletons, and ‘`breakaway_nof1`’ on their dataset excluding the singletons, and assess if the estimated levels of diversity are very different. Great disparity may provide evidence of an inflated singleton count, or at the very least, that ‘`breakaway`’ is especially sensitive to the number of rare species observed. Note that ‘`breakaway_nof1`’ may be less stable than ‘`breakaway`’ due to predicting based on a reduced dataset, and have greater standard errors.

Author(s)

Amy Willis

References

Willis, A. (2015). Species richness estimation with high diversity but spurious singletons. *arXiv.*

Willis, A. and Bunge, J. (2015). Estimating diversity via frequency ratios. *Biometrics.*

See Also

`breakaway`; `apples`

Examples

```
breakaway_nof1(apples[-1, ])
breakaway_nof1(apples[-1, ], plot = FALSE, output = FALSE, answers = TRUE)
```

`build_frequency_count_tables` *Build frequency count tables from an OTU table*

Description

Build frequency count tables from an OTU table

Usage

```
build_frequency_count_tables(the_table)
```

Arguments

`the_table`	An OTU table as a data frame or a matrix. Columns are the samples and rows give the taxa.

Value

A list of frequency count tables corresponding to the columns.

`chao1` *Chao1 species richness estimator*

Description

This function implements the Chao1 richness estimate, which is often mistakenly referred to as an index.

Usage

```
chao1(input_data, output = NULL, answers = NULL)
```

Arguments

`input_data`	An input type that can be processed by `convert()` or a `phyloseq` object
`output`	Deprecated; only for backwards compatibility
`answers`	Deprecated; only for backwards compatibility

Value

An object of class `alpha_estimate`, or `alpha_estimates` for `phyloseq` objects

Note

The authors of this package strongly discourage the use of this estimator. It is only valid when you wish to assume that every taxa has equal probability of being observed. You don't really think that's possible, do you?

Author(s)

Amy Willis

Examples

```
chao1(apples)
```

`chao1_bc` *Bias-corrected Chao1 species richness estimator*

Description

This function implements the bias-corrected Chao1 richness estimate.

Usage

```
chao1_bc(input_data, output = NULL, answers = NULL)
```

Arguments

`input_data`	An input type that can be processed by `convert()` or a `phyloseq` object
`output`	Deprecated; only for backwards compatibility
`answers`	Deprecated; only for backwards compatibility

Value

An object of class `alpha_estimate`, or `alpha_estimates` for `phyloseq` objects

Note

The authors of this package strongly discourage the use of this estimator. It is only valid when you wish to assume that every taxa has equal probability of being observed. You don't really think that's possible, do you?

Author(s)

Amy Willis

Examples

```
chao1_bc(apples)
```

`chao_bunge` *Chao-Bunge species richness estimator*

Description

This function implements the species richness estimation procedure outlined in Chao & Bunge (2002).

Usage

```
chao_bunge(input_data, cutoff = 10, output = NULL, answers = NULL)
```

Arguments

`input_data`	An input type that can be processed by `convert()` or a `phyloseq` object
`cutoff`	The maximum frequency to use in fitting.
`output`	Deprecated; only for backwards compatibility
`answers`	Deprecated; only for backwards compatibility

Value

An object of class `alpha_estimate`, or `alpha_estimates` for `phyloseq` objects

Author(s)

Amy Willis

Examples

```
chao_bunge(apples)
```

`chao_shen` *The Chao-Shen estimate of Shannon diversity*

Description

The Chao-Shen estimate of Shannon diversity

Usage

```
chao_shen(input_data)
```

Arguments

`input_data`	An input type that can be processed by `convert()`

Value

An object of class `alpha_estimate`

`check_format` *Run some basic checks on a possible frequency count table*

Description

Run some basic checks on a possible frequency count table

Usage

```
check_format(output_data)
```

Arguments

`output_data`	A matrix to test

Value

A checked frequency count table

`convert` *convert between different inputs for alpha-diversity estimates*

Description

Inputs slated for development include phyloseq and otu_table

Usage

```
convert(input_data)
```

Arguments

`input_data`	Supported types include filenames, data frames, matrices, vectors...

Value

Frequency count able

`cutoff_wrap` *Find a cut-off for estimates relying on contiguous counts*

Description

Find a cut-off for estimates relying on contiguous counts

Usage

```
cutoff_wrap(my_data, requested = NA)
```

Arguments

`my_data`	Frequency count table
`requested`	The user-requested cutoff

Value

Cutoff value

`dv` *DivNet model fitted to* `soil_phylo` *dataset*

Description

DivNet model fit via call #dv <- DivNet::divnet(soil_phylum, X = NULL) Further details in diversity hypothesis testing vignette

DivNet model fit via call #dv <- DivNet::divnet(soil_phylum, X = NULL)

Usage

```
dv
```

```
dv
```

Format

An object of class `diversityEstimates`

An object of class `diversityEstimates`

References

Whitman, T., Pepe-Ranney, C., Enders, A., Koechli, C., Campbell, A., Buckley, D. H., Lehmann, J. (2016). *Dynamics of microbial community composition and soil organic carbon mineralization in soil following addition of pyrogenic and fresh organic matter*. The ISME journal, 10(12):2918. <doi: 10.1038/ismej.2016.68>.

Whitman, T., Pepe-Ranney, C., Enders, A., Koechli, C., Campbell, A., Buckley, D. H., Lehmann, J. (2016). *Dynamics of microbial community composition and soil organic carbon mineralization in soil following addition of pyrogenic and fresh organic matter*. The ISME journal, 10(12):2918. <doi: 10.1038/ismej.2016.68>.

`dv_days_1_2` *DivNet model fitted to* `soil_phylo` *dataset including observations on day 1*

Description

DivNet model fit via call #dv_days_1_2 <- DivNet::divnet(soil_phylum_days_1_2, X = NULL) Further details in diversity hypothesis testing vignette

Usage

```
dv_days_1_2
```

Format

An object of class `diversityEstimates`

References

Whitman, T., Pepe-Ranney, C., Enders, A., Koechli, C., Campbell, A., Buckley, D. H., Lehmann, J. (2016). *Dynamics of microbial community composition and soil organic carbon mineralization in soil following addition of pyrogenic and fresh organic matter*. The ISME journal, 10(12):2918. <doi: 10.1038/ismej.2016.68>.

`F_test` *Conduct F test of null hypothesis LB = 0 using output from betta() or betta_random()*

Description

This function performs an F-test of a null hypothesis LB = 0 where B is a vector of p fixed effects returned by betta() or betta_random() and L is an m x p matrix with linearly independent rows.

Usage

```
F_test(fitted_betta, L, method = "bootstrap", nboot = 1000)
```

Arguments

`fitted_betta`	A fitted betta object – i.e., the output of either betta() or betta_random() – containing fixed effect estimates of interest.
`L`	An m x p matrix defining the null LB = 0. L must have full row rank.
`method`	A character variable indicating which method should be used to estimate the distribution of the test statistic under the null.
`nboot`	Number of bootstrap samples to use if method = "bootstrap". Ignored if method = "asymptotic".

Value

A list containing

`pval`	The p-value
`F_stat`	The calculated F statistic
`boot_F`	A vector of bootstrapped F statistics if bootstrap has been used. Otherwise NULL.

Author(s)

David Clausen

References

Willis, A., Bunge, J., and Whitman, T. (2015). Inference for changes in biodiversity. *arXiv preprint.*

See Also

`betta`;

`betta_random`;

`betta`;

`get_F_stat`

Examples

```
# generate example data
df <- data.frame(chats = c(2000, 3000, 4000, 3000,
2000, 3000, 4000, 3000), ses = c(100, 200, 150, 180,
100, 200, 150, 180),
                 Cont_var = c(100, 150, 100, 50,
                 100, 150, 100, 50),
                 Cont_var_2 = c(50,200,25,125,
                 50,200,25,125))

# fit betta()
example_fit <- betta(formula = chats ~ Cont_var + Cont_var_2, ses = ses, data = df)
```

```
# construct L for hypothesis that B_cont_var = B_cont_var_2 = 0
L <- rbind(c(0,1,0),
           c(0,0,1))

F_test_results <- F_test(example_fit,
L,
nboot = 10) #for speed; recommend much higher nboot in practice
```

`get_F_stat` *Calculate F statistic under null hypothesis LB = 0 using output from betta() or betta_random()*

Description

This function calculates an F statistic for a test of null hypothesis LB = 0 (against an unrestricted alternative) where B is a vector of p fixed effects returned by betta() or betta_random() and L is an m x p matrix with linearly independent rows.

Usage

```
get_F_stat(fitted_betta, L)
```

Arguments

`fitted_betta`	A fitted betta object – i.e., the output of either betta() or betta_random() – containing fixed effect estimates of interest.
`L`	An m x p matrix defining the null LB = 0. L must have full row rank.

Value

The calculated F-statistic

`good_turing` *The Good-Turing estimate of species richness*

Description

The Good-Turing estimate of species richness

Usage

```
good_turing(input_data)
```

Arguments

`input_data`	An input type that can be processed by `convert()` or a `phyloseq` object

Value

An object of class `alpha_estimate`, or `alpha_estimates` for `phyloseq` objects

`hawaii` *(Data) Frequency count table of soil microbes in Hawaii.*

Description

(Data) Frequency count table of soil microbes in Hawaii.

Usage

```
hawaii
```

Format

A data frame with 198 rows and 2 variables:

index an index variable

frequency number of taxa that were observed with this frequency ...

Source

`https://onlinelibrary.wiley.com/doi/abs/10.1111/biom.12332`

References

Willis, A. and Bunge, J. (2015). Estimating diversity via frequency ratios. *Biometrics*, **71**(4), 1042–1049. `https://onlinelibrary.wiley.com/doi/abs/10.1111/biom.12332`

`kemp` *Species richness estimation with Kemp-type models*

Description

This function implements the species richness estimation procedure outlined in Willis & Bunge (2015). The diversity estimate, standard error, estimated model coefficients, model details and plot of the fitted model are returned.

Usage

```
kemp(
  input_data,
  cutoff = NA,
  output = NULL,
  plot = NULL,
  answers = NULL,
  print = NULL,
  ...
)
```

Arguments

`input_data`	An input type that can be processed by `convert()`
`cutoff`	The maximum frequency count to use for model fitting
`output`	Deprecated; only for backwards compatibility
`plot`	Deprecated; only for backwards compatibility
`answers`	Deprecated; only for backwards compatibility
`print`	Deprecated; only for backwards compatibility
`...`	Additional arguments will be ignored; this is for backward compatibility

Value

An object of class `alpha_estimate`

`code`	A category representing algorithm behaviour. ‘`code=1`’ indicates no nonlinear models converged and the transformed `WLRM` diversity estimate of Rocchetti et. al. (2011) is returned. ‘`code=2`’ indicates that the iteratively reweighted model converged and was returned. ‘`code=3`’ indicates that iterative reweighting did not converge but a model based on a simplified variance structure was returned (in this case, the variance of the frequency ratios is assumed to be proportional to the denominator frequency index). Please peruse your fitted model before using your diversity estimate.

`name`	The “name” of the selected model. The first integer represents the numerator polynomial degree and the second integer represents the denominator polynomial degree of the model for the frequency ratios. See Willis & Bunge (2015) for details.
`para`	Estimated model parameters and standard errors.
`est`	The estimate of total (observed plus unobserved) diversity.
`seest`	The standard error in the diversity estimate.
`full`	The chosen nonlinear model for frequency ratios.
`ci`	An asymmetric 95% confidence interval for diversity.

Author(s)

Amy Willis

References

Willis, A. and Bunge, J. (2015). Estimating diversity via frequency ratios. *Biometrics*, **71**(4), 1042–1049.

Rocchetti, I., Bunge, J. and Bohning, D. (2011). Population size estimation based upon ratios of recapture probabilities. *Annals of Applied Statistics*, **5**.

See Also

`breakaway`; `breakaway_nof1`; `apples`

Examples

```
kemp(apples)
kemp(apples, plot = FALSE, output = FALSE, answers = TRUE)
```

`make_design_matrix` *Make design matrix*

Description

Make design matrix

Usage

```
make_design_matrix(phyloseq_object, variables)
```

Arguments

`phyloseq_object`	A phyloseq object
`variables`	variable names

Value

A design matrix

`make_frequency_count_table` *Draw frequency count subtables from an OTU table*

Description

Draw frequency count subtables from an OTU table

Usage

```
make_frequency_count_table(labels)
```

Arguments

`labels`	A vector of counts of the taxa; i.e. a vector giving the number of times each taxon was observed.

Value

A frequency count subtable

`objective_bayes_geometric` *Estimate species richness with an objective Bayes method using a geometric model*

Description

Estimate species richness with an objective Bayes method using a geometric model

Usage

```
objective_bayes_geometric(
  data,
  output = TRUE,
  plot = TRUE,
  answers = FALSE,
  tau = 10,
  burn.in = 100,
  iterations = 2500,
  Metropolis.stdev.N = 75,
  Metropolis.start.theta = 1,
  Metropolis.stdev.theta = 0.3
)
```

Arguments

`data`	A frequency count table
`output`	Print summary of fitted model. Default is TRUE.
`plot`	Plot estimates? Default is TRUE.
`answers`	Return dataframe containing model summaries? Default is FALSE.
`tau`	Truncation cutoff for model fitting. (Counts above this cutoff are not used in model fitting.) Default is 10.
`burn.in`	Number of MCMC burn-in iterations. Default is 100.
`iterations`	Number of MCMC iterations after burn-in. Default is 2500.
`Metropolis.stdev.N`	Standard deviation of proposal distribution parameter N within MCMC. Default is 75.
`Metropolis.start.theta`	Starting value of parameter theta. Default is 1
`Metropolis.stdev.theta`	Standard deviation of proposal distribution parameter theta within MCMC. Default is 0.3.

Value

If answers set to TRUE, a list containing posterior median and 95% posterior credible intervals for richness, fitted values, and MCMC diagnostics. If answers set to FALSE (default), no value is returned.

`objective_bayes_mixedgeo` *Objective Bayes species richness estimate with the mixed-geometric model*

Description

Objective Bayes species richness estimate with the mixed-geometric model

Usage

```
objective_bayes_mixedgeo(
  data,
  output = TRUE,
  plot = TRUE,
  answers = FALSE,
  tau = 10,
  burn.in = 100,
  iterations = 2500,
  Metropolis.stdev.N = 100,
  Metropolis.start.T1 = 1,
```

```
  Metropolis.stdev.T1 = 2,
  Metropolis.start.T2 = 3,
  Metropolis.stdev.T2 = 2
)
```

Arguments

`data`	A frequency count table
`output`	Print summary of fitted model. Default is TRUE.
`plot`	Plot estimates? Default is TRUE.
`answers`	Return dataframe containing model summaries? Default is FALSE.
`tau`	Truncation cutoff for model fitting. (Counts above this cutoff are not used in model fitting.) Default is 10.
`burn.in`	Number of MCMC burn-in iterations. Default is 1000.
`iterations`	Number of MCMC iterations after burn-in. Default is 5000.
`Metropolis.stdev.N`	Standard deviation of proposal distribution parameter N within MCMC. Default is 100.
`Metropolis.start.T1`	Starting value of parameter T1. Default is 1.
`Metropolis.stdev.T1`	Standard deviation of proposal distribution parameter T1 within MCMC. Default is 2.
`Metropolis.start.T2`	Starting value of parameter T2. Default is 3.
`Metropolis.stdev.T2`	Standard deviation of proposal distribution parameter T2 within MCMC. Default is 2.

Value

If answers set to TRUE, a list containing posterior median and 95% posterior credible intervals for richness, fitted values, and MCMC diagnostics. If answers set to FALSE (default), no value is returned.

`objective_bayes_negbin` *Objective Bayes species richness estimate with the Negative Binomial model*

Description

Objective Bayes species richness estimate with the Negative Binomial model

Usage

```
objective_bayes_negbin(
  data,
  output = TRUE,
  plot = TRUE,
  answers = FALSE,
  tau = 10,
  burn.in = 1000,
  iterations = 5000,
  Metropolis.stdev.N = 100,
  Metropolis.start.T1 = -0.8,
  Metropolis.stdev.T1 = 0.01,
  Metropolis.start.T2 = 0.8,
  Metropolis.stdev.T2 = 0.01
)
```

Arguments

`data`	A frequency count table
`output`	Print summary of fitted model. Default is TRUE.
`plot`	Plot estimates? Default is TRUE.
`answers`	Return dataframe containing model summaries? Default is FALSE.
`tau`	Truncation cutoff for model fitting. (Counts above this cutoff are not used in model fitting.) Default is 10.
`burn.in`	Number of MCMC burn-in iterations. Default is 1000.
`iterations`	Number of MCMC iterations after burn-in. Default is 5000.
`Metropolis.stdev.N`	Standard deviation of proposal distribution parameter N within MCMC. Default is 100.
`Metropolis.start.T1`	Starting value of parameter T1. Default is -0.8.
`Metropolis.stdev.T1`	Standard deviation of proposal distribution parameter T1 within MCMC. Default is 0.01.
`Metropolis.start.T2`	Starting value of parameter T2. Default is 0.8.
`Metropolis.stdev.T2`	Standard deviation of proposal distribution parameter T2 within MCMC. Default is 0.01.

Value

If answers set to TRUE, a list containing posterior median and 95% posterior credible intervals for richness, fitted values, and MCMC diagnostics. If answers set to FALSE (default), no value is returned.

`objective_bayes_poisson`

Objective Bayes species richness estimate with the Poisson model

---

Description

Objective Bayes species richness estimate with the Poisson model

Usage

```
objective_bayes_poisson(
  data,
  output = TRUE,
  plot = TRUE,
  answers = FALSE,
  tau = 10,
  burn.in = 100,
  iterations = 2500,
  Metropolis.stdev.N = 75,
  Metropolis.start.lambda = 1,
  Metropolis.stdev.lambda = 0.3
)
```

Arguments

`data`	A frequency count table
`output`	Print summary of fitted model. Default is TRUE.
`plot`	Plot estimates? Default is TRUE.
`answers`	Return dataframe containing model summaries? Default is FALSE.
`tau`	Truncation cutoff for model fitting. (Counts above this cutoff are not used in model fitting.) Default is 10.
`burn.in`	Number of MCMC burn-in iterations. Default is 100.
`iterations`	Number of MCMC iterations after burn-in. Default is 2500.
`Metropolis.stdev.N`	Standard deviation of proposal distribution parameter N within MCMC. Default is 75.
`Metropolis.start.lambda`	Starting value of parameter lambda. Default is 1
`Metropolis.stdev.lambda`	Standard deviation of proposal distribution parameter T1 within MCMC. Default is 0.3.

Value

If answers set to TRUE, a dataframe containing posterior median and 95% posterior credible intervals for richness, fitted values, and MCMC diagnostics. If answers set to FALSE (default), no value is returned.

`physeq_wrap` *Wrapper for* `phyloseq` *objects*

Description

Wrapper for `phyloseq` objects

Usage

```
physeq_wrap(fn, physeq, ...)
```

Arguments

`fn`	alpha diversity estimator function with `breakaway` to be applied to `physeq`
`physeq`	A `phyloseq` object, or an object of class `otu_table`
`...`	Additional arguments for `fn`

Value

Object of class `alpha_estimates`

`plot.alpha_estimates` *Plot function for alpha_estimates class*

Description

Plot function for alpha_estimates class

Usage

```
## S3 method for class 'alpha_estimates'
plot(
  x,
  physeq = NULL,
  measure = NULL,
  color = NULL,
  shape = NULL,
  title = NULL,
  trim_plot = FALSE,
  ...
)
```

Arguments

`x`	Object of class `alpha_estimates`.
`physeq`	(Optional). Default `NULL`. Required object of class `phyloseq` if including a `sample_data` variable for `color` or `shape`.
`measure`	(Optional). If there are multiple richness measures included in `x`, this can be set to the the desired measure to be plotted. Defaults to the measure of the first alpha diversity estimate.
`color`	(Optional). Default `NULL`. The sample variable to map to different colors. Can be a single character string of the variable name in `sample_data` or a custom supplied vector with length equal to the number of samples.
`shape`	(Optional). Default `NULL`. The sample variable to map to different shapes. Can be a single character string of the variable name in `sample_data` or a custom supplied vector with length equal to the number of samples.
`title`	(Optional). Default NULL. Character string. The main title for the graphic.
`trim_plot`	(Optional). Default `FALSE`. Boolean indicator for whether you want the plot to include the full confidence intervals.
`...`	See details

Details

... does not currently have any implemented options. Optional arguments currently include "trim_plot", a Optional

Value

A ggplot object with samples on the x-axis and estimated alpha diversities on the y-axis

Examples

```
## Not run:
library(phyloseq)
data(GlobalPatterns)
alphas <- breakaway(GlobalPatterns)
plot(alphas)

## End(Not run)
```

`poisson_model` *PoissonModel*

Description

A model to estimate the number of missing taxa under a Poisson Model

Usage

```
poisson_model(input_data, cutoff = 10)
```

Arguments

`input_data`	A frequency count table
`cutoff`	The largest frequency to use for predicting f0. Defaults to 10.

Value

An object of class `alpha_estimate` containing a summary of the fitted model

---

`poisson_model_nof1` *PoissonModelNof1*

Description

A model to estimate the number of missing taxa under a zero- and one-truncated Poisson Model

Usage

```
poisson_model_nof1(input_data, cutoff = 10)
```

Arguments

`input_data`	A frequency count table
`cutoff`	The largest frequency to use for predicting f0 and f1. Defaults to 10.

Value

An object of class `alpha_estimate` containing a summary of the fitted model

---

`proportions_instead` *OTU table to relative abundances*

Description

OTU table to relative abundances

Usage

```
proportions_instead(the_table)
```

Arguments

`the_table`	An OTU table

Value

A table of relative abundances

`rnbinomtable` *Negative binomially distributed frequency count tables.*

Description

Simulate a frequency count table based on a negative binomial model. Zero-truncated, obviously.

Usage

```
rnbinomtable(C, size, probability)
```

Arguments

`C`	species richness
`size`	size parameter for the negative binomial distribution
`probability`	probability parameter for the negative binomial distribution

Value

A frequency count table

Author(s)

Amy Willis

`rztnbinomtable` *beta version: Zero-truncated negative binomially distributed frequency count tables.*

Description

Simulate a frequency count table based on a negative binomial model. Zero-truncated, obviously.

Usage

```
rztnbinomtable(C, size, probability)
```

Arguments

`C`	species richness
`size`	size parameter for the negative binomial distribution
`probability`	probability parameter for the negative binomial distribution

Value

A simulated frequency count table

Author(s)

Amy Willis

`sample_inverse_simpson` *Plug-in Inverse Simpson diversity*

Description

This function implements the plug-in Inverse Simpson diversity

Usage

```
sample_inverse_simpson(input_data)
```

Arguments

`input_data`	An input type that can be processed by `convert()` or a `phyloseq` object

Value

An object of class `alpha_estimate`, or `alpha_estimates` for `phyloseq` objects

Examples

```
sample_inverse_simpson(apples)
```

`sample_richness` *Sample richness estimator*

Description

This function implements the sample richness estimate, which is the number of non-zero taxa per sample.

Usage

```
sample_richness(input_data)
```

Arguments

`input_data`	An input type that can be processed by `convert()` or a `phyloseq` object

Value

An object of class `alpha_estimate`, or `alpha_estimates` for `phyloseq` objects

Examples

```
sample_richness(apples)
```

`sample_shannon` *Plug-in Shannon diversity*

Description

This function implements the plug-in Shannon diversity

Usage

```
sample_shannon(input_data)
```

Arguments

`input_data`	An input type that can be processed by `convert()` or a `phyloseq` object

Value

An object of class `alpha_estimate`, or `alpha_estimates` for `phyloseq` objects

Examples

```
sample_shannon(apples)
```

`sample_shannon_e` *Plug-in Shannon's E ("Equitability")*

Description

This function implements the plug-in Shannon's E

Usage

```
sample_shannon_e(input_data)
```

Arguments

`input_data`	An input type that can be processed by `convert()` or a `phyloseq` object

Value

An object of class `alpha_estimate`, or `alpha_estimates` for `phyloseq` objects

Examples

```
sample_shannon_e(apples)
```

`sample_simpson` *Plug-in Simpson diversity*

Description

This function implements the plug-in Simpson diversity

Usage

```
sample_simpson(input_data)
```

Arguments

`input_data`	An input type that can be processed by `convert()` or a `phyloseq` object

Value

An object of class `alpha_estimate`, or `alpha_estimates` for `phyloseq` objects

Examples

```
sample_simpson(apples)
```

`sample_size_estimate` *Estimate the sample size needed to do an unpaired one-way test using betta*

Description

Estimate the sample size needed to do an unpaired one-way test using betta

Usage

```
sample_size_estimate(
  control_group_est,
  se_est,
  diff = 5,
  alpha = 0.05,
  prop = 0.8,
  samples = 20,
  precision = 5
)
```

Arguments

`control_group_est`	An estimate of the alpha diversity parameter for the control group
`se_est`	An estimate of the (common) standard deviation
`diff`	An estimate of the difference between the control and non-control groups
`alpha`	Minimum significance level desired
`prop`	Desired power
`samples`	Number of bootstrap resamples used to estimate the sample size. Increase for a more accurate estimate.
`precision`	How much to increment the sample size as we try to increase the power

Value

An estimate of the necessary sample size and some details

`sample_size_figure` *Plot the power obtained with sample size*

Description

Plot the power obtained with sample size

Usage

```
sample_size_figure(control_group_est, se_est, diff = 5, samples = 20)
```

Arguments

`control_group_est`	An estimate of the alpha diversity parameter for the control group
`se_est`	An estimate of the (common) standard deviation
`diff`	An estimate of the difference between the control and non-control groups
`samples`	Number of bootstrap resamples used to estimate the sample size. Increase for a more accurate estimate.

Value

A plot of the power with the sample size

`simulate_betta` *Simulate from a fitted betta model*

Description

Simulate from a fitted betta model

Usage

```
simulate_betta(fitted_betta, nsim)
```

Arguments

`fitted_betta`	A fitted betta object
`nsim`	Number of times to simulate

Value

A list of length `nsim`, each element of which is a vector of simulated Y-values under the fitted betta model

---

`simulate_betta_random` *Simulate from a fitted betta_random model*

Description

Simulate from a fitted betta_random model

Usage

```
simulate_betta_random(fitted_betta, nsim)
```

Arguments

`fitted_betta`	A fitted betta_random object
`nsim`	Number of times to simulate

Value

A list of length `nsim`, each element of which is a vector of simulated Y-values under the fitted betta model

---

`submodel_test` *Output from test_submodel() for use in hypothesis testing vignette*

Description

Output from test_submodel() for use in hypothesis testing vignette

Usage

```
submodel_test
```

Format

A list containing the bootstrap p-value, the observed F-statistic, and a vector of bootstrapped F-statistics

---

`test_submodel` *Conduct F test of null hypothesis LB = 0 using output from betta() or betta_random()*

Description

This function performs an F-test of a null hypothesis LB = 0 where B is a vector of p fixed effects returned by betta() or betta_random() and L is an m x p matrix with linearly independent rows.

Usage

```
test_submodel(
  fitted_betta,
  submodel_formula,
  method = "bootstrap",
  nboot = 1000
)
```

Arguments

`fitted_betta`	A fitted betta object – i.e., the output of either betta() or betta_random() – containing fixed effect estimates of interest.
`submodel_formula`	A formula defining which submodel to treat as the null. It is not necessary to include random effects in this formula (they will be ignored if included – the submodel will be fit with the same random effect structure as the full model regardless of input.)
`method`	A character variable indicating which method should be used to estimate the distribution of the test statistic under the null.
`nboot`	Number of bootstrap samples to use if method = "bootstrap". Ignored if method = "asymptotic".

Value

A list containing

`pval`	The p-value
`F_stat`	The calculated F statistic
`boot_F`	A vector of bootstrapped F statistics if bootstrap has been used. Otherwise NULL.

Author(s)

David Clausen

References

Willis, A., Bunge, J., and Whitman, T. (2015). Inference for changes in biodiversity. *arXiv preprint.*

See Also

betta;

betta_random;

betta;

F_test

Examples

```
# generate example data
df <- data.frame(chats = c(2000, 3000, 4000, 3000,
2000, 3000, 4000, 3000), ses = c(100, 200, 150, 180,
100, 200, 150, 180),
                 Cont_var = c(100, 150, 100, 50,
                 100, 150, 100, 50),
                 Cont_var_2 = c(50,200,25,125,
                 50,200,25,125))

# fit betta()
example_fit <- betta(formula = chats ~ Cont_var + Cont_var_2, ses = ses,
data = df)


# construct L for hypothesis that B_cont_var = B_cont_var_2 = 0
L <- rbind(c(0,1,0),
          c(0,0,1))

F_test_results <- F_test(example_fit,
L,
nboot = 10) #nboot = 10 for speed here; recommend >= 1000 in practice
```

`toy_metadata` *(Data) Data frame of covariate information about toy_otu_table.*

Description

(Data) Data frame of covariate information about toy_otu_table.

Usage

```
toy_metadata
```

Format

A data frame with 143 rows and 4 variables:

Years Year of sampling

bloom2 Did the sample correspond to a bloom event?

Period What season was sampled?

Site Where was the sample taken from? ...

`toy_otu_table` *(Data) A toy OTU table.*

Description

Covariate info available in ‘toy_metadata‘. A data frame with 448 rows and 143 columns. Rows give the abundance of each taxon; columns give the samples

Usage

```
toy_otu_table
```

Format

An object of class `data.frame` with 448 rows and 143 columns.

`toy_taxonomy` *(Data) The taxonomy of the OTUs in ‘toy_otu_table‘.*

Description

(Data) The taxonomy of the OTUs in ‘toy_otu_table‘.

Usage

```
toy_taxonomy
```

Format

An object of class `factor` of length 448.

`true_gini` *Calculate the true Gini-Simpson index*

Description

Calculate the true Gini-Simpson index

Usage

```
true_gini(input)
```

Arguments

`input`	A vector of proportions.

Value

The Gini-Simpson index of the population given by input.

Note

This function is intended for population-level data. If you are dealing with a microbial sample, use DivNet instead.

`true_hill` *Calculate the true Hill numbers*

Description

Calculate the true Hill numbers

Usage

```
true_hill(input, q)
```

Arguments

`input`	A vector of proportions.
`q`	The Hill number of interest. q = 0 corresponds to species richness, q = 2 corresponds to inverse Simpson, etc.

Value

The Hill number of the population given by input.

`true_inverse_simpson` *Calculate the true Inverse Simpson index*

Description

Calculate the true Inverse Simpson index

Usage

```
true_inverse_simpson(input)
```

Arguments

`input`	A vector of proportions.

Value

The inverse-Simpson index of the population given by input.

Note

This function is intended for population-level data. If you are dealing with a microbial sample, use DivNet instead.

`true_shannon` *Calculate the true Shannon index based on proportions*

Description

Calculate the true Shannon index based on proportions

Usage

```
true_shannon(input)
```

Arguments

`input`	A vector of proportions.

Value

The Shannon index of the population given by input.

Note

This function is intended for population-level data. If you are dealing with a microbial sample, use DivNet instead.

`true_shannon_e` *Calculate the true Shannon's equitability index*

Description

Calculate the true Shannon's equitability index

Usage

```
true_shannon_e(input)
```

Arguments

`input`	A vector of proportions.

Value

The Shannon E's of the population given by input.

Note

This function is intended for population-level data. If you are dealing with a microbial sample, use DivNet instead.

`true_simpson` *Calculate the true Simpson index*

Description

Calculate the true Simpson index

Usage

```
true_simpson(input)
```

Arguments

`input`	A vector of proportions.

Value

The Simpson index of the population given by input.

Note

This function is intended for population-level data. If you are dealing with a microbial sample, use DivNet instead.

`wlrm_transformed` *The transformed weighted linear regression estimator for species richness estimation*

Description

This function implements the transformed version of the species richness estimation procedure outlined in Rocchetti, Bunge and Bohning (2011).

Usage

```
wlrm_transformed(
  input_data,
  cutoff = NA,
  print = NULL,
  plot = NULL,
  answers = NULL
)
```

Arguments

`input_data`	An input type that can be processed by `convert()` or a `phyloseq` object
`cutoff`	Maximum frequency count to use
`print`	Deprecated; only for backwards compatibility
`plot`	Deprecated; only for backwards compatibility
`answers`	Deprecated; only for backwards compatibility

Value

An object of class `alpha_estimate`, or `alpha_estimates` for `phyloseq` objects

Note

While robust to many different structures, model is almost always misspecified. The result is usually implausible diversity estimates with artificially low standard errors. Extreme caution is advised.

Author(s)

Amy Willis

References

Rocchetti, I., Bunge, J. and Bohning, D. (2011). Population size estimation based upon ratios of recapture probabilities. *Annals of Applied Statistics*, **5**.

See Also

`breakaway`; `apples`; `wlrm_untransformed`

Examples

```
wlrm_transformed(apples)
wlrm_transformed(apples, plot = FALSE, print = FALSE, answers = TRUE)
```

`wlrm_untransformed` *The untransformed weighted linear regression estimator for species richness estimation*

Description

This function implements the untransformed version of the species richness estimation procedure outlined in Rocchetti, Bunge and Bohning (2011).

Usage

```
wlrm_untransformed(
  input_data,
  cutoff = NA,
  print = NULL,
  plot = NULL,
  answers = NULL
)
```

Arguments

`input_data`	An input type that can be processed by `convert()` or a `phyloseq` object
`cutoff`	Maximum frequency count to use
`print`	Deprecated; only for backwards compatibility
`plot`	Deprecated; only for backwards compatibility
`answers`	Deprecated; only for backwards compatibility

Value

An object of class `alpha_estimate`, or `alpha_estimates` for `phyloseq` objects

Note

This estimator is based on the negative binomial model and for that reason generally produces poor fits to microbial data. The result is usually artificially low standard errors. Caution is advised.

Author(s)

Amy Willis

References

Rocchetti, I., Bunge, J. and Bohning, D. (2011). Population size estimation based upon ratios of recapture probabilities. *Annals of Applied Statistics*, **5**.

See Also

`breakaway`; `apples`; `wlrm_transformed`

Examples

```
wlrm_untransformed(apples)
```

Index