Package: tidydiversity (via r-universe)

November 16, 2026

Title Taxonomic, Phylogenetic and Functional Diversity with Tidy Hill Numbers

Version 0.3.0

Description Tidy, pipe-friendly tools for estimating taxonomic, phylogenetic and functional diversity from long-format community data. All three dimensions are expressed as Hill numbers of the same order q (effective numbers of species), following Chao, Chiu and Jost (2014) <doi:10.1146/annurev-ecolsys-120213-091540> and Chao et al. (2019) <doi:10.1002/ecm.1343>. Computes observed and asymptotic diversity, sample coverage, and size- and coverage-based rarefaction and extrapolation following Chao et al. (2014) <doi:10.1890/13-0133.1>, and partitions beta diversity into turnover and nestedness components following Baselga (2010) <doi:10.1111/j.1466-8238.2009.00490.x>. Every function takes a data frame and returns a tibble, uses data-masking for column arguments, and supports grouped data and the '.by' argument.

License MIT + file LICENSE

URL https://tidydiversity.in4metrix.dev/, https://github.com/jmozez/tidydiversity

BugReports https://github.com/jmozez/tidydiversity/issues

Depends R (>= 4.1.0)

Imports cli (>= 3.6.0), dplyr (>= 1.1.0), generics, grDevices, pillar, Rcpp, rlang (>= 1.1.0), stats, utils, tibble (>= 3.2.0), tidyselect (>= 1.2.0), tools, vctrs (>= 0.6.0)

LinkingTo Rcpp

Suggests png, ade4, FD, indicspecies, permute, ape, phytools, betapart, broom, flextable, gawdis, gt, hillR, ggplot2 (>= 3.5.0), ggrepel, iNEXT, iNEXT.3D, iNEXT.beta3D, iNEXT.4steps, knitr, lme4, patchwork (>= 1.3.0), recipes, rsample, hardhat, parsnip, workflows, TPD, tune, yardstick, mgcv, rmarkdown, testthat (>= 3.0.0), MASS, SpadeR, rangemodelR, vegan

VignetteBuilder knitr

Config/Needs/website tidyr

Config/testthat/edition 3

Encoding UTF-8

Language en-NZ

LazyData true

Roxygen list(markdown = TRUE)

Config/roxygen2/version 8.1.1

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

Date/Publication 2026-11-16 14:38:26 UTC

RemoteUrl https://github.com/jmozez/tidydiversity

RemoteRef HEAD

RemoteSha 908db332880898a0458aba47c18b27a31178d959

Contents

as_community . . . 4
autoplot-tidydiversity . . . 5
autoplot.td_ordination . . . 10
beta_decay . . . 14
beta_decay_boot . . . 15
beta_decay_model . . . 16
beta_dist . . . 18
beta_hill . . . 18
beta_hill_pair . . . 21
beta_indicator . . . 23
beta_mantel . . . 24
beta_pair . . . 26
beta_permanova . . . 27
beta_shared . . . 29
beta_turnover_counts . . . 31
beta_welch . . . 33
data_aravo . . . 34
div_boot . . . 35
div_completeness . . . 36
div_coverage . . . 38
div_effort . . . 39
div_estimate . . . 40
div_evenness . . . 43
div_fisher_alpha . . . 44
div_hill . . . 45
div_homogenisation . . . 47
div_invasion . . . 49
div_rarefy . . . 50

div_richness . . . 52
div_standardise . . . 54
div_table . . . 57
model_case_weights . . . 58
model_count . . . 59
model_group_folds . . . 60
ord_advise . . . 61
ord_fit . . . 62
ord_interpret . . . 65
ord_prc . . . 66
ord_select . . . 68
ord_style . . . 69
ord_test . . . 71
ord_varpart . . . 72
phylo_combine_dist . . . 74
phylo_dist . . . 74
phylo_signal . . . 75
plot-partial . . . 76
plot_panel . . . 77
plot_phylo . . . 79
plot_save . . . 80
plot_trait_space . . . 81
plot_venn . . . 82
png_ants . . . 83
png_frogs . . . 84
png_tree_ants . . . 85
portal_rodents . . . 86
range_edges . . . 87
range_mde . . . 88
range_partition . . . 90
range_shift_resurvey . . . 91
range_shift_validate . . . 93
range_through . . . 95
stems . . . 96
step_hill . . . 98
theme_tidydiversity . . . 99
tidy-tidydiversity . . . 100
tidy.td_ordination . . . 102
tidyvisor . . . 103
trait_cwm . . . 104
trait_cwm_test . . . 105
trait_dist . . . 106
trait_fd_indices . . . 107
trait_fourth_corner . . . 108
trait_itv . . . 110
trait_rao . . . 111
trait_ses . . . 112
trait_tpd . . . 113

wanang_butterflies 115
wanang_caterpillars 116

Index **117**

`as_community` *Tidy community data*

Description

`as_community()` tidies long-format records into a community table with one row per site and species, and standard column names `site`, `species` and `abundance`. Repeated records are summed and zero counts are dropped.

`community_wide()` pivots a community into a site by species matrix stored as a tibble, and `community_long()` turns such a wide table back into long format.

Usage

```
as_community(
  data,
  site = site,
  species = species,
  abundance = NULL,
  .by = NULL
)

community_wide(
  data,
  site = site,
  species = species,
  abundance = NULL,
  fill = 0,
  .by = NULL
)

community_long(data, cols)
```

Arguments

`data`	A data frame in long format, with one row per record.
`site, species`	<tidy-select> The column holding site identifiers and the column holding species names.
`abundance`	<tidy-select> The column holding counts. If `NULL` (the default), a column called `abundance` is used when present, otherwise every row counts as one individual.
`.by`	<tidy-select> Optional grouping columns, kept in the output. Grouped data frames are also respected.

`fill`	Value used for species absent from a site.
`cols`	<tidy-select> In `community_long()`, the species columns of a wide table.

Value

`as_community()` returns a `td_community` tibble with the grouping columns followed by `site`, `species` and abundance.

`community_wide()` returns a tibble with one column per species. `community_long()` returns a `td_community`.

Examples

```
ants <- as_community(png_ants, .by = season)
ants

wide <- community_wide(ants, .by = season)
wide

community_long(wide, -c(season, site))
```

---

autoplot-tidydiversity

Plot tidydiversity results

Description

`autoplot()` methods draw a ggplot for every result type:

Usage

```
## S3 method for class 'td_rarefy'
autoplot(
  object,
  x = c("size", "coverage"),
  palette = "okabe_ito",
  theme = theme_tidydiversity(),
  ci = TRUE,
  facet = TRUE,
  scales = "free_y",
  ncol = NULL,
  xlab = NULL,
  ylab = y_hill,
  ...
)

## S3 method for class 'td_standardise'
autoplot(
```

```
  object,
  palette = "okabe_ito",
  theme = theme_tidydiversity(),
  ci = TRUE,
  facet = TRUE,
  scales = "free_x",
  ncol = NULL,
  xlab = NULL,
  ylab = NULL,
  ...
)

## S3 method for class 'td_beta_decay'
autoplot(
  object,
  palette = "okabe_ito",
  theme = theme_tidydiversity(),
  ci = TRUE,
  xlab = "Distance between sites",
  ylab = "Dissimilarity",
  ...
)

## S3 method for class 'td_hill'
autoplot(
  object,
  gradient = NULL,
  palette = "okabe_ito",
  theme = theme_tidydiversity(),
  facet = TRUE,
  scales = "free_y",
  ncol = NULL,
  xlab = NULL,
  ylab = y_hill,
  ...
)

## S3 method for class 'td_estimate'
autoplot(
  object,
  palette = "okabe_ito",
  theme = theme_tidydiversity(),
  ci = TRUE,
  facet = TRUE,
  scales = "free_x",
  ncol = NULL,
  xlab = y_hill,
  ylab = NULL,
```

```
  ...
)

## S3 method for class 'td_coverage'
autoplot(
  object,
  palette = "okabe_ito",
  theme = theme_tidydiversity(),
  xlab = "Sample size",
  ylab = "Sample coverage",
  labels = TRUE,
  ...
)

## S3 method for class 'td_beta_pair'
autoplot(
  object,
  component = c("total", "turnover", "nestedness"),
  theme = theme_tidydiversity(grid = "none"),
  ...
)

## S3 method for class 'td_beta_hill_pair'
autoplot(
  object,
  component = c("dis_sorensen", "dis_jaccard", "dis_turnover_sorensen",
    "dis_turnover_jaccard"),
  theme = theme_tidydiversity(grid = "none"),
  ...
)

## S3 method for class 'td_beta_multi'
autoplot(
  object,
  palette = "okabe_ito",
  theme = theme_tidydiversity(),
  xlab = NULL,
  ylab = "Multiple-site dissimilarity",
  labels = TRUE,
  digits = 2,
  ...
)

## S3 method for class 'td_beta_hill'
autoplot(
  object,
  what = c("diversity", "dissimilarity"),
  palette = "okabe_ito",
```

```
  theme = theme_tidydiversity(),
  ci = TRUE,
  xlab = "Order q",
  ylab = NULL,
  ...
)

## S3 method for class 'td_dispersion'
autoplot(object, palette = "okabe_ito", theme = theme_tidydiversity(), ...)

## S3 method for class 'td_completeness'
autoplot(
  object,
  palette = "okabe_ito",
  theme = theme_tidydiversity(),
  xlab = "Order q",
  ylab = "Sample completeness",
  ...
)

## S3 method for class 'td_evenness'
autoplot(
  object,
  palette = "okabe_ito",
  theme = theme_tidydiversity(),
  xlab = "Order q",
  ylab = "Evenness",
  ...
)

## S3 method for class 'td_mde'
autoplot(
  object,
  palette = "okabe_ito",
  theme = theme_tidydiversity(),
  xlab = NULL,
  ylab = "Species richness",
  ...
)
```

Arguments

`object`	A tibble returned by one of the functions above.
`x`	For curves, plot against sample size (`"size"`) or sample coverage (`"coverage"`).
`palette`	Colours for sites or groups: `"okabe_ito"` (default), `"grey"`, a character vector of colours, or `NULL` to keep the ggplot2 default so you can add your own scale.
`theme`	A ggplot2 theme added to the plot, by default `theme_tidydiversity()`. Use `NULL` to keep the ggplot2 default.

ci	Draw confidence intervals when the result has them.
facet	For results with several orders q, facet by q (TRUE, the default) or draw all orders on one panel with q mapped to line type or shape (FALSE). Curves computed with .by groups are also faceted by group.
scales	Facet scales, passed to ggplot2::facet_wrap().
ncol	Number of facet columns.
xlab, ylab	Axis titles. The defaults describe the quantity and, where relevant, its unit.
...	Not used.
gradient	For div_hill() results, an optional numeric column (such as elevation, kept by .by) to plot diversity along.
labels	For div_coverage() results, label each point with its site; for beta_multi() results, print the turnover, nestedness and total values on the bars. Set to FALSE to leave them out.
component	For beta_pair() results, the component to map to colour: "total", "turnover" or "nestedness".
digits	For beta_multi() results, decimal places of the value labels.
what	For beta_hill() results, "diversity" (gamma, alpha and beta) or "dissimilarity" (the four normalised dissimilarities).

Details

- div_hill(): a Hill diversity profile across q, or diversity along a gradient column when gradient is given.
- div_estimate(): observed and asymptotic diversity with intervals.
- div_coverage(): sample coverage against sample size, points labelled by site.
- div_rarefy(): rarefaction and extrapolation curves by sample size or by sample coverage.
- div_standardise(): standardised diversity with intervals.
- beta_pair(): a pairwise dissimilarity heatmap.
- beta_multi(): multi-site turnover and nestedness as stacked bars, labelled with each component and the total.
- beta_decay(): distance decay of dissimilarity components, with quasibinomial (logit-link) trends that stay within 0 to 1.
- beta_hill(): gamma, alpha and beta diversity (or, with what = "dissimilarity", the four normalised dissimilarities) against the order q for each region, with bootstrap intervals when computed.

Every method returns an ordinary ggplot object, so labels, scales, themes and facets can be changed afterwards with +, for example + labs(x = "Number of workers"), and saved with ggplot2::ggsave(). The defaults use theme_tidydiversity() and the colour-blind-safe Okabe-Ito palette. Solid lines show rarefaction (interpolation), dashed lines extrapolation, points the observed sample, and shaded ribbons or bars the bootstrap confidence intervals when they were computed.

They need the ggplot2 package.

Value

A ggplot object.

Examples

```
library(ggplot2)
curves <- div_rarefy(png_ants, q = c(0, 1), .by = season, knots = 20)
autoplot(curves)
autoplot(curves, x = "coverage")

# Customise like any ggplot
autoplot(curves, palette = "grey", ncol = 1) +
  labs(x = "Number of workers") +
  theme_tidydiversity(base_size = 9)


div_hill(png_ants, q = seq(0, 3, by = 0.25), .by = season) |>
  autoplot()
div_hill(png_ants, .by = c(season, elevation)) |>
  autoplot(gradient = elevation, xlab = "Elevation (m)")

set.seed(1)
autoplot(div_estimate(png_ants, .by = season, nboot = 20))
autoplot(div_coverage(png_ants, .by = season))
autoplot(div_standardise(png_ants, .by = season, nboot = 20))

autoplot(beta_pair(png_ants, .by = season))
autoplot(beta_multi(png_ants, .by = season))

beta_decay(png_ants, gradient = elevation, .by = season) |>
  autoplot(xlab = "Difference in elevation (m)")
```

---

autoplot.td_ordination

Plot an ordination

Description

Draws a publication-quality ordination diagram in the style of CANOCO (ter Braak and Smilauer 2012): a biplot of sites and species for unconstrained methods and a triplot with environmental arrows for constrained ones, with equal axis scaling, zero lines and the variation explained by each axis in the axis titles. Species are drawn as arrows for linear methods (PCA, RDA, PCoA, db-RDA), whose scores are directions of increase, and as points for unimodal methods (CA, CCA, DCA) and NMDS, whose scores are optima.

Usage

```
## S3 method for class 'td_ordination'
autoplot(
  object,
  axes = c(1, 2),
  colour = NULL,
  shape = NULL,
  group = NULL,
  sites = TRUE,
  site_labels = FALSE,
  species = 10,
  species_arrows = FALSE,
  arrows = TRUE,
  ellipse = c("none", "sd", "ci", "se"),
  level = 0.95,
  hull = FALSE,
  spider = FALSE,
  palette = "okabe_ito",
  theme = theme_tidydiversity(grid = "none"),
  surface = NULL,
  size = NULL,
  path = NULL,
  style = ord_style(),
  type = c("biplot", "gradient", "response", "scree", "shepard", "tvalue", "vandobben",
    "stability"),
  variable = NULL,
  axis = 1,
  model = c("glm", "gam"),
  layout = c("facet", "overlay"),
  n_axes = 10,
  nboot = 199,
  inset = c("none", "scree", "shepard"),
  ...
)
```

Arguments

object	A td_ordination from ord_fit().
axes	The two axes to plot.
colour, shape	Site variables (unquoted column names) to map to point colour and shape, such as colour = elevation. Sites are points, so they are drawn as categories (as in CANOCO), never on a continuous colour bar: a numeric variable with up to 8 distinct values gets one class per value, and otherwise is cut into up to 6 classes at pretty breaks; each class has its own colour and symbol. Discrete variables also define the groups for ellipses, hulls and spiders.
group	A site variable defining groups for ellipses, hulls and spiders, if different from colour.

`sites` Draw the sites.

`site_labels` Label the sites.

`species` Number of species to draw and label, those best represented in the plotted plane (longest scores). Use `Inf` for all and `0` for none.

`species_arrows` Draw species as arrows (a PCA/RDA biplot, as `vegan::biplot.rda()`), only for linear methods. By default species are taxa and are drawn as points (crosses) with italic labels, as in `vegan::plot.rda()`; arrows are kept for continuous explanatory variables.

`arrows` Draw environmental arrows and factor centroids.

`ellipse` `"none"` (the default), `"sd"`, `"ci"` or `"se"`.

`level` Coverage of `"ci"` and `"se"` ellipses.

`hull` Draw convex hulls around groups.

`spider` Join each site to its group centroid.

`palette` Colours for groups: `"okabe_ito"` or `"grey"`; continuous variables use the viridis scale.

`theme` A ggplot2 theme, by default `theme_tidydiversity()`.

`surface` A numeric site variable to draw as a fitted surface: contour lines from a thin-plate GAM of the variable on the two axes (as in `vegan::ordisurf()`), or a linear trend surface (a plane, as `vegan::envfit()` assumes) with fewer than 15 sites, where a smooth surface would overfit. A continuous variable is a gradient, so it is drawn with lines (contours, or the arrows of the fitted `formula`), not by colouring the points.

`size` A numeric site variable shown by symbol size (a bubble or attribute plot, as in CANOCO).

`path` A numeric site variable, such as elevation or time: joins the sites in its order with a thin line and small arrowheads, so the reader sees at once how composition changes along the gradient.

`style` Look of every element, from `ord_style()`: colours, symbols, sizes, line widths and fonts. The result is a ggplot, so any scale or theme can also be added afterwards.

`type` The kind of diagram:

- `"biplot"` (default): sites, species, fitted variables and groups;
- `"gradient"`: the ordination read as a gradient: sites (optionally joined in the order of `path`), fitted variables whose arrow width follows their r squared (non-significant ones thin, dashed and grey), the meaning of the first axis written at its ends, and the species sorted along the first axis in a panel below;
- `"response"`: species response curves: each species' abundance against an axis (`axis`) or a site variable (`variable`), fitted by a quadratic Poisson GLM (ter Braak's Gaussian response) or a GAM (`model`), with a confidence band from the model on the link scale, for the `species` best-fitted species, as panels or overlaid (`layout`);
- `"scree"`: variation explained by each axis, with the broken-stick expectation (`n_axes` axes);

- `"shepard"`: Shepard diagram of an NMDS with stress and R squared;
- `"tvalue"`: which species increase or decrease with one site variable (`variable`), the question of CANOCO's t-value biplot: species arrows coloured by an exact t-test of their coefficient in a regression of each (transformed) species on all site variables. Van Dobben circles are not drawn, as they only approximate these tests under one scaling.
- `"vandobben"`: CANOCO's t-value biplot with Van Dobben circles for one numeric site variable (`variable`): species and variables placed by a rank-2 approximation of the matrix of t-values of regression coefficients; the dashed circles mark, approximately, species with a significant positive (red) or negative (blue) coefficient, and the colours give the exact t-tests;
- `"stability"`: how robust each site's position is: 95% ellipses of `nboot` bootstrap ordinations (species resampled), rotated onto the original by Procrustes analysis (unconstrained methods).

`variable`, `axis`, `model`, `layout`, `n_axes`, `nboot`
: Options of `type` (see above).

`inset`
: For the biplot, a small diagnostic panel in a corner: `"scree"` (variance per axis) or `"shepard"` (NMDS fit).

`...`
: Not used.

Details

Groups of sites can be summarised with ellipses, convex hulls or spiders. Ellipses follow `vegan::ordiellipse()`: `"sd"` covers one standard deviation of the points, `"ci"` the region expected to hold `level` of the points, and `"se"` a `level` confidence region for the group centroid.

How to read the lines:

- red arrows from the origin: species (linear methods only);
- dark arrows from the origin: quantitative environmental variables;
- dark triangles: centroids of factor levels (for example years);
- coloured outlines: ellipses or convex hulls around groups of sites, and, with `spider = TRUE`, lines from each group centroid to its sites;
- faint dotted grey lines: label leaders, joining a label that had to be moved to avoid overlap to its point. They carry no information.

Every option is an argument, and the result is an ordinary ggplot that can be changed further with `+`.

Value

A ggplot object.

References

ter Braak, C. J. F. and Smilauer, P. (2012). Canoco reference manual and user's guide: software for ordination (version 5.0). Microcomputer Power, Ithaca, USA.

Examples

```
library(ggplot2)
data("dune", "dune.env", package = "vegan")
d <- community_long(cbind(site = rownames(dune), dune.env, dune),
  cols = dplyr::all_of(names(dune))
)

# RDA triplot of the dune meadows: A1 horizon as an arrow, management
# types as centroids with one-SD ellipses
ord <- ord_fit(d,
  abundance = abundance, method = "rda", transform = "hellinger",
  formula = ~ A1 + Management
)
autoplot(ord, colour = Management, ellipse = "sd")

# Which species respond to the A1 horizon? Van Dobben circles
autoplot(ord, type = "vandobben", variable = "A1")

# NMDS with the A1 horizon as contours and a Shepard inset
nmds <- ord_fit(d, abundance = abundance, method = "nmds", formula = ~A1)
autoplot(nmds, surface = A1, inset = "shepard")
```

---

beta_decay *Distance decay of similarity*

Description

Pairs the pairwise beta diversity from `beta_pair()` with the distance between sites along a gradient, such as elevation, to study how dissimilarity and its components change with distance.

Usage

```
beta_decay(
  data,
  gradient,
  site = site,
  species = species,
  abundance = NULL,
  index = c("sorensen", "jaccard", "bray"),
  .by = NULL,
  index.family = NULL
)
```

Arguments

data	A data frame in long format, with one row per record.

gradient	<tidy-select> One or more numeric site attributes, constant within each site. With one column (such as elevation) the distance is the absolute difference; with several (such as c(x, y) coordinates) it is the Euclidean distance.
site, species	<tidy-select> The column holding site identifiers and the column holding species names.
abundance	<tidy-select> The column holding counts. If NULL (the default), a column called abundance is used when present, otherwise every row counts as one individual.
index	Family of dissimilarity: "sorensen" or "jaccard" (presence and absence) or "bray" (abundance-based Bray-Curtis).
.by	<tidy-select> Optional grouping columns, kept in the output. Grouped data frames are also respected.
index.family	Alias of index, the name used by the betapart package (index.family = "sorensen"), so code written for betapart works unchanged.

Value

A td_beta_decay tibble like the output of beta_pair(), with distance and, for a single gradient column, gradient_1 and gradient_2. Use ggplot2::autoplot() to plot it.

Examples

```
decay <- beta_decay(png_ants, gradient = elevation, .by = season)
decay
```

---

beta_decay_boot *Bootstrap confidence interval for a distance-decay slope*

Description

beta_decay_boot() estimates the slope of dissimilarity against distance (from beta_decay()) and its bootstrap confidence interval. Because pairs of sites share sites and are not independent, sites, not pairs, are resampled: each replicate draws the sites with replacement and refits the slope on every pair of distinct drawn sites. Compare the slopes of two transects (or taxa) by whether their intervals overlap, or by the interval of their difference (compare).

Usage

```
beta_decay_boot(
  decay,
  component = c("total", "turnover", "nestedness"),
  nboot = 999,
  conf = 0.95,
  group = NULL,
  per = 1,
  seed = NULL
)
```

Arguments

decay	A beta_decay() result.
component	Dissimilarity column: "total" (default), "turnover" or "nestedness".
nboot	Number of bootstrap replicates.
conf	Confidence level.
group	Optional column of decay identifying transects or taxa; slopes are estimated within each, and with exactly two groups their difference is added.
per	Scale the slope to dissimilarity per per distance units (for example 1000 for per km when distance is in metres).
seed	Optional random seed.

Value

A tibble with group (if given), slope, lower, upper, n_sites and nboot; with two groups an extra row difference.

Examples

```
d <- beta_decay(png_frogs, elevation)
beta_decay_boot(d, nboot = 200, per = 1000, seed = 1)
```

---

beta_decay_model *Fit and test a distance-decay model*

Description

Fits a nonlinear distance-decay model to the pairwise similarities from beta_decay() and tests it by permutation, as in betapart::decay.model() (Gómez-Rodríguez and Baselga 2018). Similarity is one minus the chosen dissimilarity component. The models are

Usage

```
beta_decay_model(
  x,
  model = c("exponential", "power"),
  component = c("total", "turnover", "nestedness"),
  permutations = 999,
  model.type = NULL
)
```

Arguments

x	A td_beta_decay tibble from beta_decay(). Its groups are kept and fitted separately.
model	"exponential" or "power".
component	Dissimilarity component turned into similarity: "total", "turnover" or "nestedness".
permutations	Number of permutations for the test; 0 skips it.
model.type	Alias of model, the name used by betapart::decay.model() (model.type = "exponential").

Details

- "exponential": $S = ae^{-bd}$ (Nekola and White 1999), and
- "power": $S = ad^{-b}$, which needs positive distances.

Here a is the similarity at distance zero (exponential) or one (power), and b is the rate of decay: positive values mean similarity falls with distance. The fit is by nonlinear least squares, and the fit statistic is the pseudo-R^2, one minus the residual deviance over the deviance of a constant model.

Pairs of sites are not independent, so the test permutes site labels: rows and columns of the similarity matrix are shuffled together while the distances stay in place, as in a Mantel test, and the model is refitted each time. The p-value is $(1 + \#\{R^2_{perm} \geq R^2_{obs}\})/(1 + B)$. betapart instead reports the proportion of permuted values strictly greater than the observed one, which can be zero; the version here cannot.

Value

A tibble with one row per group: model, component, n_pairs, a, b, half_distance (the distance over which similarity halves, $\log 2/b$ for the exponential model and the factor $2^{1/b}$ for the power model), pseudo_r2, aic and p.value.

References

Nekola, J. C. and White, P. S. (1999). The distance decay of similarity in biogeography and ecology. *Journal of Biogeography*, 26, 867-878. doi:10.1046/j.13652699.1999.00305.x

Gómez-Rodríguez, C. and Baselga, A. (2018). Variation among European beetle taxa in patterns of distance decay of similarity suggests a major role of dispersal processes. *Ecography*, 41, 1825-1834. doi:10.1111/ecog.03693

Examples

```
decay <- beta_decay(png_ants, gradient = elevation, .by = season)
set.seed(1)
beta_decay_model(decay, permutations = 199)
```

beta_dist *Pairwise dissimilarity as a distance matrix*

Description

Returns one component of `beta_pair()`, or one dissimilarity of `beta_hill_pair()`, as a stats::dist object, ready for ordination and testing with vegan, for example `vegan::metaMDS()`, `vegan::adonis2()` or `vegan::betadisper()`, or for clustering with `stats::hclust()`.

Usage

```
beta_dist(x, component = NULL, q = NULL)
```

Arguments

`x`	A `td_beta_pair` tibble from `beta_pair()` or a `td_beta_hill_pair` tibble from `beta_hill_pair()`, without groups.
`component`	For `beta_pair()` output: `"total"`, `"turnover"` or `"nestedness"` (default `"total"`). For `beta_hill_pair()` output: one of its `dis_` columns (default `"dis_sorensen"`).
`q`	For `beta_hill_pair()` output with several orders, the order to use.

Value

A `dist` object labelled with the site names, in their order of first appearance.

Examples

```
d <- beta_dist(beta_pair(png_ants[png_ants$season == "wet", ]))
hclust(d)

wet <- png_ants[png_ants$season == "wet", ]
beta_dist(beta_hill_pair(wet, abundance = workers), q = 2)
```

beta_hill *Coverage-standardised beta diversity with Hill numbers*

Description

Gamma, alpha and beta diversity of order q, standardised to a common sample coverage, following Chao et al. (2023) (the iNEXT.beta3D framework). Each `.by` group is a region whose sites are its assemblages. Gamma diversity is the diversity of the pooled region; alpha diversity is computed from the species-by-site data and divided by the number of sites; each is rarefied or extrapolated to the sample size at which it reaches the target coverage. Beta = gamma / alpha is the effective number of completely distinct assemblages (1 to N).

Usage

```
beta_hill(
  data,
  site = site,
  species = species,
  abundance = NULL,
  q = c(0, 1, 2),
  coverage = NULL,
  type = c("TD", "PD", "FD"),
  tree = NULL,
  reftime = NULL,
  pd_unit = c("mean", "total"),
  distance = NULL,
  tau = NULL,
  unit = NULL,
  n_units = NULL,
  nboot = 0,
  conf = 0.95,
  .by = NULL
)
```

Arguments

data	A data frame in long format, with one row per record.
site, species	<tidy-select> The column holding site identifiers and the column holding species names.
abundance	<tidy-select> The column holding counts. If NULL (the default), a column called abundance is used when present, otherwise every row counts as one individual.
q	Orders of diversity.
coverage	Target sample coverage(s) in (0, 1]. By default, the lowest coverage that the pooled data of every region reaches at twice its sample size, so no region is extrapolated too far.
type	Diversity dimension: "TD" (taxonomic, the default), "PD" (phylogenetic, needs tree) or "FD" (functional, needs distance). Give several, such as c("TD", "PD"), or "all" for all three, to analyse the dimensions together: the results are stacked in one tibble with a type column, and autoplot() shows them side by side. All three are Hill numbers of the same order q, in units of effective numbers of species (Chao, Chiu and Jost 2014), and reproduce iNEXT.3D.
tree	For type = "PD", a rooted ape::phylo tree with branch lengths whose tip labels include every species.
reftime	For type = "PD", the reference time (tree depth) at which diversity is measured. Defaults to the age of the root of the tree pruned to the species in data.
pd_unit	For type = "PD", "mean" (the default) divides phylogenetic diversity by reftime, giving the effective number of lineages on the same scale as species; "total" gives effective total branch length.

distance	For type = "FD", a species-by-species distance matrix (or stats::dist object) with names matching the species, scaled to lie between 0 and 1, for example from Gower distances of traits.
tau	For type = "FD", the distance threshold below which species are treated as functionally identical. Defaults to the mean distance between two individuals in the pooled data. Use "auc" to integrate over all thresholds from 0 to 1 (Chao et al. 2019).
unit	For incidence data, the column naming the sampling unit (a trap, bait, plot or survey day) in which each species was detected. Each site is then described by the number of units in which each species was found and the number of units sampled, and every estimate is incidence-based (Chao et al. 2014), for TD, PD and FD alike. Leave NULL for abundance data.
n_units	For incidence data, a column giving the number of units sampled at each site, constant within a site. Use it when some units detected no species and so do not appear in the data. By default the number of distinct units in the data is used.
nboot	Number of bootstrap replicates for confidence intervals; 0 for none.
conf	Confidence level.
.by	<tidy-select> Optional grouping columns, kept in the output. Grouped data frames are also respected.

Details

Standardising by coverage removes the dependence of beta diversity on sample size and completeness, so regions sampled with different effort can be compared (Chao et al. 2023). Four normalised dissimilarities in 0-1 are derived from beta:

- dis_sorensen (1 - C): Sorensen-type non-overlap;
- dis_jaccard (1 - U): Jaccard-type non-overlap;
- dis_turnover_sorensen (1 - V): Sorensen-type turnover;
- dis_turnover_jaccard (1 - S): Jaccard-type turnover.

For q = 0 they reduce to the classic Sorensen and Jaccard family; for q = 1 and 2 they generalise Horn and Morisita-Horn overlap.

Value

A tibble with the .by columns, q, coverage, the sample sizes m_gamma and m_alpha, gamma, alpha, beta, n_sites, the four dissimilarities and, with nboot > 0, their confidence limits.

Dimensions and data types

type = "PD" gives phylogenetic beta diversity from tree, and type = "FD" functional beta diversity from distance, exactly as in iNEXT.beta3D. Sample sizes are always those at which the taxonomic data reach the target coverage. For FD, tau = "auc" integrates gamma, alpha and beta over 30 thresholds between 0 and 1, as iNEXT.beta3D does.

With unit, the data are incidences: each site was surveyed in the same T sampling units (plots, visits, traps), and a unit's label must identify the same unit at every site of a region, because gamma diversity counts a species as present in a unit if it was found there at any site.

Confidence intervals

With `nboot > 0`, every region is resampled as a whole (Chao et al. 2023): each site's community, including estimated undetected species, is redrawn with its sample size (or units) and all quantities are recomputed at the same coverage. Each estimate gets `_lower` and `_upper` columns, the estimate -/+ z x bootstrap standard error, kept within each quantity's range (beta in 1 to N, dissimilarities in 0 to 1).

References

Chao, A., Thorn, S., Chiu, C.-H. et al. (2023). Rarefaction and extrapolation with beta diversity under a framework of Hill numbers: the iNEXT.beta3D standardization. *Ecological Monographs*, 93, e1588. doi:10.1002/ecm.1588

Examples

```
beta_hill(png_ants, .by = season, coverage = 0.9)
```

`beta_hill_pair` *Pairwise Hill-number dissimilarity*

Description

For every pair of sites, `beta_hill_pair()` computes gamma, alpha and beta diversity of order `q` from the observed abundances. It also gives the four normalised dissimilarities of Chao, Chiu and Jost (2014) for two assemblages, with the same names as in `beta_hill()`: `dis_sorensen` ($1 - C_{qN}$), `dis_jaccard` ($1 - U_{qN}$), `dis_turnover_sorensen` ($1 - V_{qN}$) and `dis_turnover_jaccard` ($1 - S_{qN}$). Classic indices are special cases:

Usage

```
beta_hill_pair(
  data,
  site = site,
  species = species,
  abundance = NULL,
  q = c(0, 1, 2),
  weight = c("equal", "size"),
  .by = NULL
)
```

Arguments

`data`	A data frame in long format, with one row per record.
`site, species`	<tidy-select> The column holding site identifiers and the column holding species names.

abundance	<tidy-select> The column holding counts. If NULL (the default), a column called abundance is used when present, otherwise every row counts as one individual.
q	Orders of diversity, between 0 and 3 (non-negative numbers).
weight	"equal" or "size", as described above.
.by	<tidy-select> Optional grouping columns, kept in the output. Grouped data frames are also respected.

Details

Order	dis_sorensen	dis_jaccard
q = 0	Sorensen	Jaccard
q = 1	Horn	Horn
q = 2	Morisita-Horn	Regional overlap

With weight = "equal" (the default) each site's abundances are turned into relative abundances before pooling, so a heavily sampled site does not dominate the pair, and q = 2 gives Morisita-Horn exactly. With weight = "size" raw abundances are pooled, as in beta_hill() and iNEXT.beta3D. At q = 0 the weighting makes no difference.

These are empirical values. Use beta_hill() to compare whole sets of sites at a standardised sample coverage.

Value

A td_beta_hill_pair tibble with the grouping columns, site_1, site_2, q, gamma, alpha, beta and the four dis_ columns. beta_dist() turns one order into a distance matrix, and ggplot2::autoplot() draws it as a heatmap.

References

Chao, A., Chiu, C.-H. and Jost, L. (2014). Unifying species diversity, phylogenetic diversity, functional diversity, and related similarity and differentiation measures through Hill numbers. *Annual Review of Ecology, Evolution, and Systematics*, 45, 297-324. doi:10.1146/annurevecolsys120213-091540

Jost, L. (2007). Partitioning diversity into independent alpha and beta components. *Ecology*, 88, 2427-2439. doi:10.1890/061736.1

Examples

```
beta_hill_pair(png_ants, abundance = workers, .by = season)
```

`beta_indicator` *Indicator species analysis*

Description

Finds species associated with groups of sites (Dufrene and Legendre 1997; De Caceres et al. 2010) using `indicspecies::multipatt()`, and returns a tidy table.

Usage

```
beta_indicator(
  data,
  site = site,
  species = species,
  abundance = NULL,
  group,
  env = NULL,
  func = c("IndVal.g", "IndVal", "r.g", "r"),
  combinations = FALSE,
  adjust = "holm",
  alpha = 0.05,
  permutations = 999,
  duleg = NULL
)
```

Arguments

`data`	A data frame in long format, with one row per record.
`site, species`	<tidy-select> The column holding site identifiers and the column holding species names.
`abundance`	<tidy-select> The column holding counts. If `NULL` (the default), a column called `abundance` is used when present, otherwise every row counts as one individual.
`group`	Site variable (unquoted) defining the groups.
`env`	A data frame of site variables with a column named like the site column. By default, site variables are taken from `data`: every column that is constant within each site.
`func`	`"IndVal.g"` (group-equalised indicator value, the default), `"IndVal"`, `"r.g"` or `"r"` (point-biserial correlation).
`combinations`	Allow species to indicate combinations of groups?
`adjust`	Correction for testing many species at once, passed to `stats::p.adjust()`: `"holm"` (the default, controls the family-wise error rate), `"fdr"` (false discovery rate) or `"none"`.
`alpha`	Return only species with adjusted `p.adj` <= `alpha`; 1 for all.

permutations	Permutations for the tests of passive variables: a number, or "exact" for complete enumeration of all n! orderings of the sites (feasible up to about 9 sites; with 7 sites, 5,040), which gives exact p-values.
duleg	Alias from indicspecies::multipatt(): duleg = TRUE is combinations = FALSE.

Details

Every species is tested, so with many species some reach p.value < 0.05 by chance. Following De Caceres et al. (2010) and Legendre and Legendre (2012), p-values are corrected for multiple testing before species are selected.

Value

A tibble with species, group (the group or combination the species indicates), stat, A (specificity) and B (fidelity; for IndVal only), the permutation p.value and the adjusted p.adj, sorted by group and decreasing stat.

References

Dufrene, M. and Legendre, P. (1997). Species assemblages and indicator species: the need for a flexible asymmetrical approach. *Ecological Monographs*, 67, 345-366. doi:10.1890/00129615(1997)067[0345:SAAIST]2.0.

De Caceres, M., Legendre, P. and Moretti, M. (2010). Improving indicator species analysis by combining groups of sites. *Oikos*, 119, 1674-1684. doi:10.1111/j.16000706.2010.18334.x

Examples

```
ants <- dplyr::mutate(png_ants, unit = paste(site, season, transect),
                      zone = ifelse(elevation <= 700, "lowland", "montane"))
set.seed(1)
beta_indicator(ants, site = unit, group = zone, permutations = 199)
```

---

beta_mantel *Mantel and partial Mantel tests*

Description

beta_mantel() tests whether two distance matrices are correlated, for example community dissimilarity and elevational or geographic distance, with the Mantel test (Mantel 1967). With z, it gives the partial Mantel test of x and y controlling for z (Smouse et al. 1986). Significance comes from permuting the sites of x (vegan's vegan::mantel() and vegan::mantel.partial()).

Pairs of sites are not independent, so do not test distance-decay relationships with ordinary regression p-values; use this test instead. With few sites, use permutations = "exact" to enumerate every ordering of the sites (n!; 5,040 for 7 sites); the p-value is then exact.

alternative sets the tail: "greater" (the default) tests for a positive correlation, "less" for a negative one (for example similarity decreasing with distance), "two.sided" for either.

Usage

```
beta_mantel(
  x,
  y,
  z = NULL,
  method = c("spearman", "pearson", "kendall"),
  permutations = 9999,
  alternative = c("greater", "less", "two.sided"),
  strata = NULL,
  seed = NULL
)
```

Arguments

x, y, z	Distance matrices: `dist` objects (such as the result of `beta_dist()`), square matrices, or, for `y` and `z`, a numeric vector of site values (such as elevation), from which Euclidean distances are computed. All must describe the same sites in the same order; when they carry labels, the labels are matched.
`method`	Correlation: `"spearman"` (default), `"pearson"` or `"kendall"`.
`permutations`	Number of permutations, or `"exact"`.
`alternative`	`"greater"`, `"less"` or `"two.sided"`.
`strata`	Optional grouping of sites within which permutations are restricted.
`seed`	Optional random seed.

Value

A one-row tibble with `statistic` (the Mantel r), `p.value`, `alternative`, `permutations`, `exact`, `method` and `partial`.

References

Mantel, N. (1967). The detection of disease clustering and a generalized regression approach. Cancer Research, 27, 209-220.

Smouse, P. E., Long, J. C. and Sokal, R. R. (1986). Multiple regression and correlation extensions of the Mantel test of matrix correspondence. Systematic Zoology, 35, 627-632.

Examples

```
wet <- dplyr::filter(png_ants, season == "wet")
d_beta <- beta_dist(beta_pair(wet), "turnover")
elev <- dplyr::distinct(wet, site, elevation)
beta_mantel(d_beta, stats::setNames(elev$elevation, elev$site),
  permutations = "exact"
)
```

beta_pair *Partition beta diversity into turnover and nestedness*

Description

`beta_pair()` computes pairwise dissimilarity between sites and splits it into turnover (species replacement) and nestedness (species loss) components following Baselga (2010, 2012). `beta_multi()` computes the multiple-site version for all sites in each group. Both use presence and absence, so any positive abundance counts as presence.

With `index = "sorensen"` the components are Simpson turnover, the nestedness-resultant component and Sorensen dissimilarity. With `index = "jaccard"` they are the Jaccard equivalents. With `index = "bray"` abundances are used: `turnover` is the balanced variation in abundance and `nestedness` the abundance gradient component of Bray-Curtis dissimilarity (Baselga 2013), matching `betapart::beta.pair.abund()` and `betapart::beta.multi.abund()`.

Usage

```
beta_pair(
  data,
  site = site,
  species = species,
  abundance = NULL,
  index = c("sorensen", "jaccard", "bray"),
  .by = NULL,
  index.family = NULL
)

beta_multi(
  data,
  site = site,
  species = species,
  abundance = NULL,
  index = c("sorensen", "jaccard", "bray"),
  .by = NULL,
  index.family = NULL
)
```

Arguments

`data`	A data frame in long format, with one row per record.
`site, species`	<tidy-select> The column holding site identifiers and the column holding species names.
`abundance`	<tidy-select> The column holding counts. If `NULL` (the default), a column called `abundance` is used when present, otherwise every row counts as one individual.

index	Family of dissimilarity: "sorensen" or "jaccard" (presence and absence) or "bray" (abundance-based Bray-Curtis).
.by	<tidy-select> Optional grouping columns, kept in the output. Grouped data frames are also respected.
index.family	Alias of index, the name used by the betapart package (index.family = "sorensen"), so code written for betapart works unchanged.

Value

beta_pair() returns a tibble with the grouping columns, site_1, site_2, turnover, nestedness and total. beta_multi() returns one row per group with n_sites, turnover, nestedness and total.

References

Baselga, A. (2010). Partitioning the turnover and nestedness components of beta diversity. Global Ecology and Biogeography, 19, 134-143. doi:10.1111/j.14668238.2009.00490.x

Baselga, A. (2012). The relationship between species replacement, dissimilarity derived from nestedness, and nestedness. Global Ecology and Biogeography, 21, 1223-1232. doi:10.1111/j.1466-8238.2011.00756.x

Baselga, A. (2013). Separating the two components of abundance-based dissimilarity: balanced changes in abundance vs. abundance gradients. Methods in Ecology and Evolution, 4, 552-557. doi:10.1111/2041210X.12029

Examples

```
beta_pair(png_ants, .by = season)

beta_multi(png_ants, .by = season, index = "jaccard")
```

---

beta_permanova *PERMANOVA and dispersion tests on community composition*

Description

beta_permanova() tests whether composition differs with site variables using permutational multivariate analysis of variance (vegan::adonis2(), Anderson 2001). beta_dispersion() tests whether groups differ in their spread (multivariate dispersion, vegan::betadisper(), Anderson 2006). PERMANOVA is sensitive to differences in dispersion, so report both.

Usage

```
beta_permanova(
  data,
  site = site,
  species = species,
  abundance = NULL,
```

```
  formula,
  env = NULL,
  distance = "bray",
  transform = "none",
  by = c("margin", "terms", "model"),
  strata = NULL,
  permutations = 999,
  method = NULL
)

beta_dispersion(
  data,
  site = site,
  species = species,
  abundance = NULL,
  group,
  env = NULL,
  distance = "bray",
  transform = "none",
  type = c("median", "centroid"),
  permutations = 999,
  method = NULL
)
```

Arguments

data	A data frame in long format, with one row per record.
site, species	<tidy-select> The column holding site identifiers and the column holding species names.
abundance	<tidy-select> The column holding counts. If NULL (the default), a column called abundance is used when present, otherwise every row counts as one individual.
formula	One-sided formula of site variables, such as ~ elevation + season.
env	A data frame of site variables with a column named like the site column. By default, site variables are taken from data: every column that is constant within each site.
distance	Dissimilarity for "pcoa", "nmds" and "dbrda", passed to vegan::vegdist(), such as "bray" (the default) or "jaccard".
transform	Transformation of the community matrix before the analysis: "hellinger" (the default for PCA and RDA, which otherwise overweight abundant species; Legendre and Gallagher 2001), "chord", "log" (log(x) + 1 for x > 0), "sqrt", "pa" (presence-absence) or "none" (the default for the other methods).
by	"margin" (the default: each term tested after all the others, so the result does not depend on the order of terms), "terms" (sequential: each term after those listed before it, so order matters when terms are correlated) or "model" (all terms together). Note that vegan::adonis2() defaults to "terms"; set by = "terms" to reproduce its default output.

strata	Optional site variable (unquoted) within which to restrict permutations, for example a block or transect.
permutations	Permutations for the tests of passive variables: a number, or "exact" for complete enumeration of all n! orderings of the sites (feasible up to about 9 sites; with 7 sites, 5,040), which gives exact p-values.
method	Alias of distance, the name used by vegan::vegdist(), vegan::adonis2() and vegan::betadisper() (method = "bray").
group	Site variable (unquoted) defining the groups.
type	Distance to the group spatial "median" (the default, as in vegan) or "centroid".

Details

Site variables are taken from the data when they are constant within sites, as in ord_fit(), or from env.

Value

A tibble: one row per term, with df, sum_sq, r2, statistic (pseudo-F) and p.value.

For beta_dispersion(), a tibble with one row per site (group, distance to its group centre) and the permutation test as attribute "test" (also printed). Use autoplot() to draw it.

References

Anderson, M. J. (2001). A new method for non-parametric multivariate analysis of variance. *Austral Ecology*, 26, 32-46. doi:10.1111/j.14429993.2001.01070.pp.x

Anderson, M. J. (2006). Distance-based tests for homogeneity of multivariate dispersions. *Biometrics*, 62, 245-253. doi:10.1111/j.15410420.2005.00440.x

Examples

```
ants <- dplyr::mutate(png_ants, unit = paste(site, season, transect))
set.seed(1)
beta_permanova(ants, site = unit, formula = ~ elevation + season,
               permutations = 199)
beta_dispersion(ants, site = unit, group = season, permutations = 199)
```

---

beta_shared *Shared species and similarity corrected for unseen species*

Description

For every pair of sites:

- `beta_shared()` estimates the total number of species the two sites share, including shared species that neither sample detected, with the Chao1-shared estimator for abundance data and the Chao2-shared estimator for incidence data (Chao et al. 2000; Pan et al. 2009).
- `beta_chao()` gives the abundance-based Jaccard and Sørensen similarities of Chao et al. (2005), which correct for shared species missed by incomplete samples, next to the classic indices.

Classic indices underestimate similarity when samples are incomplete, because a species shared by two sites but rare in one of them is often missed. The Chao versions estimate the probability that an individual of one site belongs to a species shared with the other (`u`, `v`), adding the shared species that were seen once in one sample. They are the estimators of EstimateS and SpadeR; the values equal SpadeR's, and `chao_jaccard` equals one minus vegan's `vegdist(method = "chao")`. This is checked in the test suite.

Usage

```
beta_shared(
  data,
  site = site,
  species = species,
  abundance = NULL,
  unit = NULL,
  n_units = NULL,
  .by = NULL
)

beta_chao(
  data,
  site = site,
  species = species,
  abundance = NULL,
  unit = NULL,
  n_units = NULL,
  .by = NULL
)
```

Arguments

`data`	A data frame in long format, with one row per record.
`site`, `species`	<tidy-select> The column holding site identifiers and the column holding species names.
`abundance`	<tidy-select> The column holding counts. If `NULL` (the default), a column called `abundance` is used when present, otherwise every row counts as one individual.

unit	For incidence data, the column naming the sampling unit (a trap, bait, plot or survey day) in which each species was detected. Each site is then described by the number of units in which each species was found and the number of units sampled, and every estimate is incidence-based (Chao et al. 2014), for TD, PD and FD alike. Leave NULL for abundance data.
n_units	For incidence data, a column giving the number of units sampled at each site, constant within a site. Use it when some units detected no species and so do not appear in the data. By default the number of distinct units in the data is used.
.by	<tidy-select> Optional grouping columns, kept in the output. Grouped data frames are also respected.

Value

beta_shared(): a tibble with the grouping columns, site_1, site_2, s_1, s_2 (observed richness), observed (observed shared species) and estimate. beta_chao(): a tibble with site_1, site_2, jaccard, sorensen (classic, on species lists), u, v, chao_jaccard and chao_sorensen (similarities, 0-1).

References

Chao, A., Hwang, W.-H., Chen, Y.-C. and Kuo, C.-Y. (2000). Estimating the number of shared species in two communities. Statistica Sinica, 10, 227-246.

Chao, A., Chazdon, R. L., Colwell, R. K. and Shen, T.-J. (2005). A new statistical approach for assessing similarity of species composition with incidence and abundance data. Ecology Letters, 8, 148-159. doi:10.1111/j.14610248.2004.00707.x

Pan, H.-Y., Chao, A. and Foissner, W. (2009). A nonparametric lower bound for the number of species shared by multiple communities. Journal of Agricultural, Biological and Environmental Statistics, 14, 452-468.

Examples

```
beta_shared(png_ants, .by = season)
beta_chao(png_ants, .by = season)
```

---

beta_turnover_counts *Shared, lost and gained species between pairs of sites*

Description

For each pair of sites, beta_turnover_counts() counts the species they share, the species found only in the first site (lost on moving to the second) and only in the second (gained), and gives the pairwise dissimilarities with Baselga's partitions:

- sorensen = turnover (Simpson dissimilarity, beta.sim) + nestedness (beta.sne) (Baselga 2010);
- jaccard = turnover_j + nestedness_j (Baselga 2012);

- with abundances, `bray` = `balanced` (balanced variation in abundance) + `gradient` (abundance gradient) (Baselga 2013).

The values equal those of the betapart package. Along a gradient, use `pairs = "adjacent"` with `order_by` to compare each site with the next.

A site with no species makes the dissimilarities undefined (0 / 0 when both sites are empty, or a pure loss): such pairs are returned with a warning, their counts are kept and undefined indices are `NA`.

Usage

```
beta_turnover_counts(
  data,
  site = site,
  species = species,
  abundance = NULL,
  pairs = "adjacent",
  order_by = NULL,
  .by = NULL
)
```

Arguments

`data`	A data frame in long format, with one row per record.
`site`, `species`	<tidy-select> The column holding site identifiers and the column holding species names.
`abundance`	<tidy-select> The column holding counts. If `NULL` (the default), a column called `abundance` is used when present, otherwise every row counts as one individual.
`pairs`	`"adjacent"` (each site with the next in `order_by` order), `"all"` (every pair), or a data frame with two columns of site names.
`order_by`	Column giving the order of sites for `"adjacent"` pairs, such as elevation; by default the natural site order.
`.by`	<tidy-select> Optional grouping columns, kept in the output. Grouped data frames are also respected.

Value

A tibble with the grouping columns, `site_1`, `site_2`, `s_1`, `s_2` (richness), `shared`, `lost`, `gained`, `sorensen`, `turnover`, `nestedness`, `jaccard`, `turnover_j`, `nestedness_j`, and with abundances `bray`, `balanced` and `gradient`.

References

Baselga, A. (2010). Partitioning the turnover and nestedness components of beta diversity. Global Ecology and Biogeography, 19, 134-143.

Baselga, A. (2012). The relationship between species replacement, dissimilarity derived from nestedness, and nestedness. Global Ecology and Biogeography, 21, 1223-1232.

Baselga, A. (2013). Separating the two components of abundance-based dissimilarity: balanced changes in abundance vs. abundance gradients. Methods in Ecology and Evolution, 4, 552-557.

Examples

```
wet <- dplyr::filter(png_ants, season == "wet")
beta_turnover_counts(wet, abundance = workers, order_by = elevation)
```

`beta_welch` *Welch-type PERMANOVA robust to unequal dispersion*

Description

Tests whether community composition differs among groups with the W*d statistic of Hamidi et al. (2019), a distance-based generalisation of Welch's ANOVA. Unlike classic PERMANOVA (`beta_permanova()`), it does not assume that groups have the same multivariate dispersion, so a significant result reflects a shift in location even when groups differ in spread or size. For two groups it equals the Tw2 test of Alekseyenko (2016). With one variable and Euclidean distance it equals Welch's F statistic (`stats::oneway.test()` with `var.equal = FALSE`).

Usage

```
beta_welch(
  data,
  site = site,
  species = species,
  abundance = NULL,
  group,
  env = NULL,
  distance = "bray",
  transform = "none",
  permutations = 999,
  method = NULL
)
```

Arguments

`data`	A data frame in long format, with one row per record.
`site, species`	<`tidy-select`> The column holding site identifiers and the column holding species names.
`abundance`	<`tidy-select`> The column holding counts. If `NULL` (the default), a column called `abundance` is used when present, otherwise every row counts as one individual.
`group`	Site variable (unquoted) defining the groups.
`env`	A data frame of site variables with a column named like the site column. By default, site variables are taken from `data`: every column that is constant within each site.

distance	Dissimilarity for "pcoa", "nmds" and "dbrda", passed to vegan::vegdist(), such as "bray" (the default) or "jaccard".
transform	Transformation of the community matrix before the analysis: "hellinger" (the default for PCA and RDA, which otherwise overweight abundant species; Legendre and Gallagher 2001), "chord", "log" (log(x) + 1 for x > 0), "sqrt", "pa" (presence-absence) or "none" (the default for the other methods).
permutations	Number of permutations.
method	Alias of distance, the name used by vegan::vegdist(), vegan::adonis2() and vegan::betadisper() (method = "bray").

Details

The p-value is obtained by permuting group labels.

Value

A one-row tibble with statistic (W*d), n_groups, n_sites and p.value.

References

Hamidi, B. et al. (2019). W*d-test: robust distance-based multivariate analysis of variance. *Microbiome*, 7, 51. doi:10.1186/s4016801906599

Alekseyenko, A. V. (2016). Multivariate Welch t-test on distances. *Bioinformatics*, 32, 3552-3558. doi:10.1093/bioinformatics/btw524

Examples

```
ants <- dplyr::mutate(png_ants, unit = paste(site, season, transect))
set.seed(1)
beta_welch(ants, site = unit, group = season, permutations = 199)
```

---

data_aravo *Alpine plant communities with traits (aravo) in long format*

Description

Converts the aravo data of the ade4 package (Choler 2005) into the long format used by tidydiversity: 82 alpine plant species in 75 sites in the Aravo valley (French Alps), with 6 site variables (including snowmelt date, Snow, and physical disturbance, PhysD) and 8 functional traits. The data are loaded from ade4, not copied into tidydiversity.

Usage

```
data_aravo()
```

Value

A list with `community` (a tibble with `site`, `species`, `cover` (abundance class 0-5) and the site variables) and `traits` (a tibble with `species` and the 8 traits).

References

Choler, P. (2005). Consistent shifts in Alpine plant traits along a mesotopographical gradient. *Arctic, Antarctic, and Alpine Research*, 37, 444-453. doi:10.1657/15230430(2005)037[0444:CSIAPT]2.0.CO;2

Examples

```
aravo <- data_aravo()
aravo$community
aravo$traits
```

`div_boot` *Bootstrap replicates of diversity estimates*

Description

Returns the individual bootstrap replicates behind the standard errors of `div_estimate()` and `div_standardise()`, one row per site, order `q` and replicate. Each replicate is a multinomial sample of the observed size drawn from the estimated community, including undetected species, as in iNEXT (Chao et al. 2014). Use the replicates to carry estimation uncertainty into downstream models, for example by refitting a model to each replicate and pooling the coefficients (see `vignette("modelling")`).

Usage

```
div_boot(
  data,
  site = site,
  species = species,
  abundance = NULL,
  q = c(0, 1, 2),
  type = c("asymptotic", "standardised"),
  coverage = NULL,
  size = NULL,
  nboot = 100,
  .by = NULL
)
```

Arguments

data	A data frame in long format, with one row per record.
site, species	<tidy-select> The column holding site identifiers and the column holding species names.
abundance	<tidy-select> The column holding counts. If NULL (the default), a column called abundance is used when present, otherwise every row counts as one individual.
q	Orders of diversity: any of 0, 1 and 2.
type	"asymptotic" for the asymptotic estimates of div_estimate(), or "standardised" for diversity at a common coverage or size as in div_standardise().
coverage	Target sample coverage between 0 and 1. If both coverage and size are NULL, the target is chosen by level.
size	Target sample size. Supply either coverage or size.
nboot	Number of bootstrap replicates.
.by	<tidy-select> Optional grouping columns, kept in the output. Grouped data frames are also respected.

Value

A tibble with the grouping columns, the site column, q, .draw (replicate number) and diversity. For type = "standardised", also m, the size each site was standardised to (fixed across replicates).

Examples

```
set.seed(1)
draws <- div_boot(png_ants, q = 0, .by = season, nboot = 20)
draws

# The bootstrap standard error of each site
dplyr::summarise(draws, se = sd(diversity), .by = c(season, site, q))
```

---

div_completeness *Sample completeness profiles*

Description

Estimates sample completeness of order q for each site (Chao et al. 2020). Completeness at q = 0 is the proportion of species richness detected, at q = 1 it is the sample coverage, and higher orders give increasing weight to abundant species. Values lie between 0 and 1, except in samples made up only of singletons, where estimates for q > 0 are unreliable and can be negative (as in iNEXT.4steps); a warning flags such sites.

Usage

```
div_completeness(
  data,
  site = site,
  species = species,
  abundance = NULL,
  q = seq(0, 2, 0.2),
  conf = 0.95,
  nboot = 50,
  .by = NULL
)
```

Arguments

data	A data frame in long format, with one row per record.
site, species	<tidy-select> The column holding site identifiers and the column holding species names.
abundance	<tidy-select> The column holding counts. If NULL (the default), a column called abundance is used when present, otherwise every row counts as one individual.
q	Orders of completeness, non-negative numbers. Defaults to a profile from 0 to 2.
conf	Confidence level of the intervals.
nboot	Number of bootstrap replicates. Use 0 to skip them.
.by	<tidy-select> Optional grouping columns, kept in the output. Grouped data frames are also respected.

Value

A td_completeness tibble with the grouping columns, the site column, q, completeness, se, lower and upper.

References

Chao, A., Kubota, Y., Zeleny, D., Chiu, C.-H., Li, C.-F., Kusumoto, B., Yasuhara, M., Thorn, S., Wei, C.-L., Costello, M. J. and Colwell, R. K. (2020). Quantifying sample completeness and comparing diversities among assemblages. Ecological Research, 35, 292-314. doi:10.1111/1440-1703.12102

Examples

```
div_completeness(png_ants, q = c(0, 1, 2), .by = season, nboot = 0)
```

`div_coverage` *Sample coverage*

Description

Estimates the sample coverage of each site, the proportion of the individuals in the community that belong to species already recorded (Chao and Jost 2012).

Usage

```
div_coverage(
  data,
  site = site,
  species = species,
  abundance = NULL,
  unit = NULL,
  n_units = NULL,
  .by = NULL
)
```

Arguments

`data`	A data frame in long format, with one row per record.
`site, species`	<tidy-select> The column holding site identifiers and the column holding species names.
`abundance`	<tidy-select> The column holding counts. If `NULL` (the default), a column called `abundance` is used when present, otherwise every row counts as one individual.
`unit`	For incidence data, the column naming the sampling unit (a trap, bait, plot or survey day) in which each species was detected. Each site is then described by the number of units in which each species was found and the number of units sampled, and every estimate is incidence-based (Chao et al. 2014), for TD, PD and FD alike. Leave `NULL` for abundance data.
`n_units`	For incidence data, a column giving the number of units sampled at each site, constant within a site. Use it when some units detected no species and so do not appear in the data. By default the number of distinct units in the data is used.
`.by`	<tidy-select> Optional grouping columns, kept in the output. Grouped data frames are also respected.

Value

A tibble with the grouping columns, the site column, `n`, `s_obs`, `f1`, `f2` (singletons and doubletons) and `coverage`.

References

Chao, A. and Jost, L. (2012). Coverage-based rarefaction and extrapolation: standardizing samples by completeness rather than size. Ecology, 93, 2533-2547. doi:10.1890/111952.1

Examples

```
div_coverage(png_ants, .by = season)
```

`div_effort` *Additional sampling effort needed*

Description

`div_effort()` answers the planning question "how much more do I need to sample?" for each site. It gives the number of additional individuals (abundance data) or sampling units (incidence data: traps, plots, days) needed to reach

- a target **sample coverage** (`coverage`), from the coverage extrapolation of Chao and Jost (2012); and/or
- a target **fraction of the estimated richness** (`fraction`), that is detecting `fraction` x Chao1 (or Chao2) species, as proposed by Chao et al. (2009). The richness extrapolation of Shen et al. (2003) is solved exactly, so the answer agrees with the extrapolated curves of `div_rarefy()`.

Targets the sample has already reached give zero. A target fraction of 1 cannot be reached with finite effort and is refused; fractions of 0.8 to 0.95 are typical. Extrapolating far beyond about twice or three times the current sample is uncertain, so treat large values as indications of order of magnitude.

Usage

```
div_effort(
  data,
  site = site,
  species = species,
  abundance = NULL,
  unit = NULL,
  n_units = NULL,
  coverage = c(0.9, 0.95),
  fraction = c(0.8, 0.9),
  .by = NULL
)
```

Arguments

`data`	A data frame in long format, with one row per record.
`site, species`	<`tidy-select`> The column holding site identifiers and the column holding species names.

abundance	<tidy-select> The column holding counts. If NULL (the default), a column called abundance is used when present, otherwise every row counts as one individual.
unit	For incidence data, the column naming the sampling unit (a trap, bait, plot or survey day) in which each species was detected. Each site is then described by the number of units in which each species was found and the number of units sampled, and every estimate is incidence-based (Chao et al. 2014), for TD, PD and FD alike. Leave NULL for abundance data.
n_units	For incidence data, a column giving the number of units sampled at each site, constant within a site. Use it when some units detected no species and so do not appear in the data. By default the number of distinct units in the data is used.
coverage	Target coverage levels in (0, 1).
fraction	Target fractions of the estimated richness in (0, 1).
.by	<tidy-select> Optional grouping columns, kept in the output. Grouped data frames are also respected.

Value

A tibble with the grouping columns, the site column, n (current sample size), s_obs, coverage_now, s_est (Chao1 or Chao2), target ("coverage" or "fraction"), level, additional (additional individuals or units, rounded up) and total.

References

Chao, A., Colwell, R. K., Lin, C.-W. and Gotelli, N. J. (2009). Sufficient sampling for asymptotic minimum species richness estimators. Ecology, 90, 1125-1133. doi:10.1890/072147.1

Chao, A. and Jost, L. (2012). Coverage-based rarefaction and extrapolation: standardizing samples by completeness rather than size. Ecology, 93, 2533-2547. doi:10.1890/111952.1

Shen, T.-J., Chao, A. and Lin, C.-F. (2003). Predicting the number of new species in further taxonomic sampling. Ecology, 84, 798-804.

Examples

```
div_effort(png_ants, unit = bait, n_units = baits, .by = season)
```

---

div_estimate *Asymptotic diversity estimates*

Description

Estimates the asymptotic Hill numbers of order 0, 1 and 2 for each site using the Chao1 richness estimator, the Chao et al. (2013) Shannon estimator and the unbiased Simpson estimator. Standard errors and confidence intervals come from bootstrap resampling of the estimated community, as in iNEXT.

Usage

```
div_estimate(
  data,
  site = site,
  species = species,
  abundance = NULL,
  q = c(0, 1, 2),
  conf = 0.95,
  nboot = 50,
  ci = c("lognormal", "normal"),
  type = c("TD", "PD", "FD"),
  tree = NULL,
  reftime = NULL,
  pd_unit = c("mean", "total"),
  distance = NULL,
  tau = NULL,
  unit = NULL,
  n_units = NULL,
  .by = NULL
)
```

Arguments

data	A data frame in long format, with one row per record.
site, species	<tidy-select> The column holding site identifiers and the column holding species names.
abundance	<tidy-select> The column holding counts. If NULL (the default), a column called abundance is used when present, otherwise every row counts as one individual.
q	Orders of diversity: any of 0, 1 and 2.
conf	Confidence level of the intervals.
nboot	Number of bootstrap replicates. Use 0 to skip them. For PD and FD the bootstrap community adds undetected species with estimated branch lengths (PD) or distances (FD); see Details in div_estimate().
ci	How to build the confidence intervals from the bootstrap standard error. With "lognormal" (the default) the richness (q = 0) interval uses the log-transformation of Chao (1987), as in SpadeR, so it is asymmetric and its lower limit never falls below the observed richness; orders 1 and 2 use estimate +/- z * se. With "normal" every order uses estimate +/- z * se, as in iNEXT::iNEXT(). In both cases the lower limit is truncated at the observed diversity, which the asymptotic diversity cannot fall below.
type	Diversity dimension: "TD" (taxonomic, the default), "PD" (phylogenetic, needs tree) or "FD" (functional, needs distance). Give several, such as c("TD", "PD"), or "all" for all three, to analyse the dimensions together: the results are stacked in one tibble with a type column, and autoplot() shows them side by side. All three are Hill numbers of the same order q, in units of effective numbers of species (Chao, Chiu and Jost 2014), and reproduce iNEXT.3D.

`tree`	For `type = "PD"`, a rooted ape::phylo tree with branch lengths whose tip labels include every species.
`reftime`	For `type = "PD"`, the reference time (tree depth) at which diversity is measured. Defaults to the age of the root of the tree pruned to the species in `data`.
`pd_unit`	For `type = "PD"`, `"mean"` (the default) divides phylogenetic diversity by `reftime`, giving the effective number of lineages on the same scale as species; `"total"` gives effective total branch length.
`distance`	For `type = "FD"`, a species-by-species distance matrix (or stats::dist object) with names matching the species, scaled to lie between 0 and 1, for example from Gower distances of traits.
`tau`	For `type = "FD"`, the distance threshold below which species are treated as functionally identical. Defaults to the mean distance between two individuals in the pooled data. Use `"auc"` to integrate over all thresholds from 0 to 1 (Chao et al. 2019).
`unit`	For incidence data, the column naming the sampling unit (a trap, bait, plot or survey day) in which each species was detected. Each site is then described by the number of units in which each species was found and the number of units sampled, and every estimate is incidence-based (Chao et al. 2014), for TD, PD and FD alike. Leave `NULL` for abundance data.
`n_units`	For incidence data, a column giving the number of units sampled at each site, constant within a site. Use it when some units detected no species and so do not appear in the data. By default the number of distinct units in the data is used.
`.by`	<`tidy-select`> Optional grouping columns, kept in the output. Grouped data frames are also respected.

Details

For `type = "PD"` and `type = "FD"` the bootstrap resamples an estimated community that includes undetected species, as for taxonomic diversity. Under PD each undetected species is given its own branch, of length equal to the estimated undetected branch length divided by the estimated number of undetected species (Chao et al. 2015; Hsieh and Chao 2017). Under FD undetected species are assumed to resemble the rarest observed species: their distance to each observed species is the mean distance from the singletons to that species, and their distance to each other is the mean distance among the singletons. The reference time (PD) and threshold `tau` (FD) are held fixed across replicates.

Value

A tibble with the grouping columns, the site column, `q`, `observed`, `estimate`, `se`, `lower` and `upper`.

References

Chao, A., Gotelli, N. J., Hsieh, T. C., Sander, E. L., Ma, K. H., Colwell, R. K. and Ellison, A. M. (2014). Rarefaction and extrapolation with Hill numbers. Ecological Monographs, 84, 45-67. doi:10.1890/130133.1

Chao, A. (1987). Estimating the population size for capture-recapture data with unequal catchability. Biometrics, 43, 783-791. doi:10.2307/2531532

Examples

```
set.seed(1)
div_estimate(png_ants, .by = season, nboot = 20)

# Incidence-based (Chao2) estimates, baits as sampling units
div_estimate(png_ants, unit = bait, n_units = baits, .by = season, nboot = 20)
```

`div_evenness` *Evenness profiles*

Description

Computes the five classes of Hill-number evenness of Chao and Ricotta (2019) for each site, either from the observed sample or from diversity standardised to a common sample coverage, as in iNEXT.4steps.

Usage

```
div_evenness(
  data,
  site = site,
  species = species,
  abundance = NULL,
  q = seq(0, 2, 0.2),
  class = c("E1", "E2", "E3", "E4", "E5"),
  method = c("estimated", "observed"),
  coverage = NULL,
  conf = 0.95,
  nboot = 50,
  .by = NULL
)
```

Arguments

`data`	A data frame in long format, with one row per record.
`site, species`	<tidy-select> The column holding site identifiers and the column holding species names.
`abundance`	<tidy-select> The column holding counts. If `NULL` (the default), a column called `abundance` is used when present, otherwise every row counts as one individual.
`q`	Orders of completeness, non-negative numbers. Defaults to a profile from 0 to 2.
`class`	Evenness classes to return, any of `"E1"` to `"E5"`.
`method`	`"estimated"` (the default) uses diversity standardised to `coverage`; `"observed"` uses the observed sample.

coverage	Target coverage for method = "estimated". Defaults to the lowest coverage that every site in a group reaches at twice its sample size.
conf	Confidence level of the intervals.
nboot	Number of bootstrap replicates. Use 0 to skip them.
.by	<tidy-select> Optional grouping columns, kept in the output. Grouped data frames are also respected.

Value

A td_evenness tibble with the grouping columns, the site column, class, q, evenness, se, lower and upper, and coverage for the estimated method.

References

Chao, A. and Ricotta, C. (2019). Quantifying evenness and linking it to diversity, beta diversity, and similarity. Ecology, 100, e02852. doi:10.1002/ecy.2852

Chao, A. et al. (2020). Quantifying sample completeness and comparing diversities among assemblages. Ecological Research, 35, 292-314. doi:10.1111/14401703.12102

Examples

```
div_evenness(png_ants, q = c(0, 1, 2), .by = season, nboot = 0)
```

---

div_fisher_alpha *Fisher's alpha*

Description

Fisher's log-series diversity index alpha (Fisher, Corbet and Williams 1943), the parameter that satisfies S = alpha log(1 + n / alpha). It is nearly independent of sample size when abundances follow a log-series, which made it the standard index in tropical ecology, and EstimateS reports it. The values equal vegan's fisher.alpha().

Usage

```
div_fisher_alpha(
  data,
  site = site,
  species = species,
  abundance = NULL,
  .by = NULL
)
```

Arguments

data	A data frame in long format, with one row per record.
site, species	<tidy-select> The column holding site identifiers and the column holding species names.
abundance	<tidy-select> The column holding counts. If NULL (the default), a column called abundance is used when present, otherwise every row counts as one individual.
.by	<tidy-select> Optional grouping columns, kept in the output. Grouped data frames are also respected.

Value

A tibble with the grouping columns, the site column, n, s_obs and alpha.

References

Fisher, R. A., Corbet, A. S. and Williams, C. B. (1943). The relation between the number of species and the number of individuals in a random sample of an animal population. Journal of Animal Ecology, 12, 42-58. doi:10.2307/1411

Examples

```
div_fisher_alpha(png_ants, .by = season)
```

---

div_hill *Observed Hill numbers*

Description

Computes Hill numbers of order q for each site from the observed relative abundances. Order 0 is species richness, order 1 is the exponential of Shannon entropy and order 2 is the inverse Simpson index.

Usage

```
div_hill(
  data,
  site = site,
  species = species,
  abundance = NULL,
  q = c(0, 1, 2),
  type = c("TD", "PD", "FD"),
  tree = NULL,
  reftime = NULL,
  pd_unit = c("mean", "total"),
  distance = NULL,
```

```
  tau = NULL,
  unit = NULL,
  n_units = NULL,
  .by = NULL
)
```

Arguments

data	A data frame in long format, with one row per record.
site, species	<tidy-select> The column holding site identifiers and the column holding species names.
abundance	<tidy-select> The column holding counts. If NULL (the default), a column called abundance is used when present, otherwise every row counts as one individual.
q	Orders of diversity, non-negative numbers.
type	Diversity dimension: "TD" (taxonomic, the default), "PD" (phylogenetic, needs tree) or "FD" (functional, needs distance). Give several, such as c("TD", "PD"), or "all" for all three, to analyse the dimensions together: the results are stacked in one tibble with a type column, and autoplot() shows them side by side. All three are Hill numbers of the same order q, in units of effective numbers of species (Chao, Chiu and Jost 2014), and reproduce iNEXT.3D.
tree	For type = "PD", a rooted ape::phylo tree with branch lengths whose tip labels include every species.
reftime	For type = "PD", the reference time (tree depth) at which diversity is measured. Defaults to the age of the root of the tree pruned to the species in data.
pd_unit	For type = "PD", "mean" (the default) divides phylogenetic diversity by reftime, giving the effective number of lineages on the same scale as species; "total" gives effective total branch length.
distance	For type = "FD", a species-by-species distance matrix (or stats::dist object) with names matching the species, scaled to lie between 0 and 1, for example from Gower distances of traits.
tau	For type = "FD", the distance threshold below which species are treated as functionally identical. Defaults to the mean distance between two individuals in the pooled data. Use "auc" to integrate over all thresholds from 0 to 1 (Chao et al. 2019).
unit	For incidence data, the column naming the sampling unit (a trap, bait, plot or survey day) in which each species was detected. Each site is then described by the number of units in which each species was found and the number of units sampled, and every estimate is incidence-based (Chao et al. 2014), for TD, PD and FD alike. Leave NULL for abundance data.
n_units	For incidence data, a column giving the number of units sampled at each site, constant within a site. Use it when some units detected no species and so do not appear in the data. By default the number of distinct units in the data is used.
.by	<tidy-select> Optional grouping columns, kept in the output. Grouped data frames are also respected.

Value

A tibble with the grouping columns, the site column, `q`, `n` (number of individuals, or of sampling units for incidence data) and `diversity`. Phylogenetic and functional results also have `type` and `reftime` or `tau`.

References

Hill, M. O. (1973). Diversity and evenness: a unifying notation and its consequences. Ecology, 54, 427-432. doi:10.2307/1934352

Chao, A., Chiu, C.-H. and Jost, L. (2014). Unifying species diversity, phylogenetic diversity, functional diversity, and related similarity and differentiation measures through Hill numbers. Annual Review of Ecology, Evolution, and Systematics, 45, 297-324. doi:10.1146/annurevecolsys120213-091540

Chao, A., Chiu, C.-H., Villeger, S., Sun, I.-F., Thorn, S., Lin, Y.-C., Chiang, J.-M. and Sherwin, W. B. (2019). An attribute-diversity approach to functional diversity, functional beta diversity, and related (dis)similarity measures. Ecological Monographs, 89, e01343. doi:10.1002/ecm.1343

Examples

```
div_hill(png_ants, .by = season)

# Grouped data frames work too
png_ants |>
  dplyr::group_by(season) |>
  div_hill(q = c(0, 0.5, 1, 2))

# Incidence data: each bait is a sampling unit, 60 baits per site
div_hill(png_ants, unit = bait, n_units = baits, .by = season)

# TD, PD and FD together, here for a functional distance matrix only
d <- as.matrix(stats::dist(c(s1 = 0, s2 = 0.1, s3 = 0.5, s4 = 1)))
toy <- data.frame(
  site = rep(c("a", "b"), each = 4), species = paste0("s", 1:4),
  n = c(10, 5, 2, 1, 3, 3, 1, 1)
)
div_hill(toy, abundance = n, type = c("TD", "FD"), distance = d)
```

`div_homogenisation` *Biotic homogenisation between two surveys*

Description

`div_homogenisation()` asks whether communities have become more alike over time, as when widespread alien species spread and local endemics are lost (biotic homogenisation; McKinney and Lockwood 1999; Olden and Rooney 2006). It computes the mean pairwise similarity among the same sites in two surveys (Jaccard or Sorensen, presence-absence) and the change, $\Delta = \bar{S}_{after} - \bar{S}_{before}$: positive values mean homogenisation, negative values differentiation.

Site pairs share sites, so they are not independent. The confidence interval therefore resamples sites (a site bootstrap): sites are drawn with replacement, and the change is recomputed over the pairs of distinct drawn sites.

With `status`, the change is also computed with native species only: the difference between the two (`alien_contribution`) is the part of the change due to alien species.

Usage

```
div_homogenisation(
  before,
  after,
  site = site,
  species = species,
  status = NULL,
  alien = c("alien", "introduced", "exotic", "non-native", "invasive"),
  index = c("jaccard", "sorensen"),
  nboot = 999,
  conf = 0.95,
  seed = NULL
)
```

Arguments

`before`, `after`	Long-format data of the two surveys.
`site`, `species`	<tidy-select> The column holding site identifiers and the column holding species names.
`status`	Column giving each record's status. Values in `alien` count as alien; all others as native.
`alien`	Values of `status` that mean alien (default `"alien"`, `"introduced"`, `"exotic"`, `"non-native"`, `"invasive"`).
`index`	`"jaccard"` or `"sorensen"` similarity.
`nboot`	Number of bootstrap resamples of sites.
`conf`	Confidence level.
`seed`	Optional random seed.

Value

A tibble with one row per species set (`"all"` and, with `status`, `"native"`): `species_set`, `sites`, `similarity_before`, `similarity_after`, `change`, `lower`, `upper` (bootstrap interval of the change); with `status`, an attribute `alien_contribution`.

References

McKinney, M. L. and Lockwood, J. L. (1999). Biotic homogenization: a few winners replacing many losers in the next mass extinction. Trends in Ecology & Evolution, 14, 450-453.

Olden, J. D. and Rooney, T. P. (2006). On defining and quantifying biotic homogenization. Global Ecology and Biogeography, 15, 113-120.

Examples

```
# A small illustrative example: one alien species spreads to every site
before <- data.frame(
  site = rep(1:4, each = 2),
  species = c("a", "b", "c", "d", "e", "f", "g", "h"),
  status = "native"
)
after <- rbind(before, data.frame(site = 1:4, species = "x", status = "alien"))
div_homogenisation(before, after, status = status, nboot = 99, seed = 1)
```

`div_invasion` *Native and alien species in communities*

Description

`div_invasion()` summarises, for each site (and group), how much of a community is made up of alien (introduced, non-native) species: the number of native and alien species, the alien share of the species and, with abundances, the alien share of the individuals. These are the standard measures of the level of invasion of a community (Catford et al. 2012).

Usage

```
div_invasion(
  data,
  status,
  site = site,
  species = species,
  abundance = NULL,
  alien = c("alien", "introduced", "exotic", "non-native", "invasive"),
  .by = NULL
)
```

Arguments

`data`	A data frame in long format, with one row per record.
`status`	Column giving each record's status. Values in `alien` count as alien; all others as native.
`site, species`	<tidy-select> The column holding site identifiers and the column holding species names.
`abundance`	<tidy-select> The column holding counts. If `NULL` (the default), a column called `abundance` is used when present, otherwise every row counts as one individual.
`alien`	Values of `status` that mean alien (default `"alien"`, `"introduced"`, `"exotic"`, `"non-native"`, `"invasive"`).
`.by`	<tidy-select> Optional grouping columns, kept in the output. Grouped data frames are also respected.

Value

A tibble with one row per site (and group): `site`, `richness`, `native`, `alien`, `alien_share` (of species) and, with abundance, `alien_abundance_share`.

References

Catford, J. A., Vesk, P. A., Richardson, D. M. and Pysek, P. (2012). Quantifying levels of biological invasion: towards the objective classification of invaded and invasible ecosystems. Global Change Biology, 18, 44-62.

Examples

```
# A small illustrative table
d <- data.frame(
  site = rep(c("a", "b"), each = 4),
  species = c("s1", "s2", "s3", "s4", "s1", "s5", "s6", "s7"),
  status = c("native", "native", "native", "alien", "native", "alien", "alien", "native"),
  count = c(10, 5, 3, 2, 4, 20, 6, 1)
)
div_invasion(d, status = status, abundance = count)
```

`div_rarefy` *Rarefaction and extrapolation curves*

Description

Computes size-based rarefaction and extrapolation of Hill numbers for each site, with the matching sample coverage. Rarefaction is exact (Chao et al. 2014) and extrapolation uses the Chao et al. (2014) estimators, reproducing iNEXT.

Usage

```
div_rarefy(
  data,
  site = site,
  species = species,
  abundance = NULL,
  q = c(0, 1, 2),
  size = NULL,
  endpoint = NULL,
  knots = 40,
  conf = 0.95,
  nboot = 0,
  type = c("TD", "PD", "FD"),
  tree = NULL,
  reftime = NULL,
  pd_unit = c("mean", "total"),
```

```
  distance = NULL,
  tau = NULL,
  unit = NULL,
  n_units = NULL,
  .by = NULL
)
```

Arguments

data	A data frame in long format, with one row per record.
site, species	<tidy-select> The column holding site identifiers and the column holding species names.
abundance	<tidy-select> The column holding counts. If NULL (the default), a column called abundance is used when present, otherwise every row counts as one individual.
q	Orders of diversity: any of 0, 1 and 2.
size	Sample sizes at which to compute diversity. If NULL, a grid of knots sizes from 1 to endpoint is used, always including the observed sample size.
endpoint	Largest sample size of the default grid. Defaults to twice the observed sample size of each site, the recommended limit for extrapolating richness. Larger values give a warning.
knots	Number of sizes in the default grid.
conf	Confidence level of the intervals.
nboot	Number of bootstrap replicates for confidence intervals. The default, 0, skips them.
type	Diversity dimension: "TD" (taxonomic, the default), "PD" (phylogenetic, needs tree) or "FD" (functional, needs distance). Give several, such as c("TD", "PD"), or "all" for all three, to analyse the dimensions together: the results are stacked in one tibble with a type column, and autoplot() shows them side by side. All three are Hill numbers of the same order q, in units of effective numbers of species (Chao, Chiu and Jost 2014), and reproduce iNEXT.3D.
tree	For type = "PD", a rooted ape::phylo tree with branch lengths whose tip labels include every species.
reftime	For type = "PD", the reference time (tree depth) at which diversity is measured. Defaults to the age of the root of the tree pruned to the species in data.
pd_unit	For type = "PD", "mean" (the default) divides phylogenetic diversity by reftime, giving the effective number of lineages on the same scale as species; "total" gives effective total branch length.
distance	For type = "FD", a species-by-species distance matrix (or stats::dist object) with names matching the species, scaled to lie between 0 and 1, for example from Gower distances of traits.
tau	For type = "FD", the distance threshold below which species are treated as functionally identical. Defaults to the mean distance between two individuals in the pooled data. Use "auc" to integrate over all thresholds from 0 to 1 (Chao et al. 2019).

`unit`	For incidence data, the column naming the sampling unit (a trap, bait, plot or survey day) in which each species was detected. Each site is then described by the number of units in which each species was found and the number of units sampled, and every estimate is incidence-based (Chao et al. 2014), for TD, PD and FD alike. Leave `NULL` for abundance data.
`n_units`	For incidence data, a column giving the number of units sampled at each site, constant within a site. Use it when some units detected no species and so do not appear in the data. By default the number of distinct units in the data is used.
`.by`	<tidy-select> Optional grouping columns, kept in the output. Grouped data frames are also respected.

Value

A `td_rarefy` tibble with the grouping columns, the site column, `q`, `m` (sample size), `method` (`"rarefaction"`, `"observed"` or `"extrapolation"`), `diversity`, `coverage`, `lower` and `upper`. Use `ggplot2::autoplot()` to draw the curves.

Examples

```
curves <- div_rarefy(png_ants, q = 0, .by = season, knots = 20)
curves

div_rarefy(png_ants, size = c(10, 20, 40), .by = season)
```

`div_richness` *Classic species richness estimators*

Description

`div_richness()` gives the non-parametric richness estimators of EstimateS (Colwell 2013) and SpadeR (Chao et al. 2016) for every site, with bootstrap standard errors and log-normal confidence intervals.

For abundance data (counts of individuals):

- `chao1`: Chao (1984), with the small-sample factor (n - 1)/n.
- `chao1_bc`: the bias-corrected Chao1, finite even without doubletons.
- `ichao1`: the improved Chao1 of Chiu et al. (2014), which also uses tripletons and quadrupletons.
- `ace`, `ace1`: the abundance-based coverage estimators of Chao and Lee (1992); species seen more than `k` times are "abundant" and counted as observed.
- `jack1`, `jack2`: first- and second-order jackknife (Burnham and Overton 1978).

For incidence data (detections in sampling units, given with `unit`):

- `chao2`, `chao2_bc`, `ichao2`: the incidence versions of the above.

- `ice`, `ice1`: the incidence-based coverage estimators of Lee and Chao (1994), with the improved coverage estimate of Chao et al. (2016).
- `jack1`, `jack2`: jackknife estimators on sampling units.
- `bootstrap`: the bootstrap estimator of Smith and van Belle (1984).

Point estimates equal those of SpadeR (and of vegan's `estimateR()` and `specpool()` where they compute the same estimator); this is checked in the test suite. Standard errors come from `nboot` bootstrap samples of the estimated community, as in `div_estimate()`, so every estimator has one.

Which to report? Chao1/Chao2 are lower bounds with good statistical properties and are the usual choice; iChao1/iChao2 are less biased when many species are rare. ACE/ICE and the jackknives are given for comparison with older studies. For comparing sites, prefer coverage-standardised diversity (`div_standardise()`).

Usage

```
div_richness(
  data,
  site = site,
  species = species,
  abundance = NULL,
  unit = NULL,
  n_units = NULL,
  estimator = NULL,
  k = 10,
  conf = 0.95,
  nboot = 100,
  .by = NULL
)
```

Arguments

`data`	A data frame in long format, with one row per record.
`site, species`	<`tidy-select`> The column holding site identifiers and the column holding species names.
`abundance`	<`tidy-select`> The column holding counts. If `NULL` (the default), a column called `abundance` is used when present, otherwise every row counts as one individual.
`unit`	For incidence data, the column naming the sampling unit (a trap, bait, plot or survey day) in which each species was detected. Each site is then described by the number of units in which each species was found and the number of units sampled, and every estimate is incidence-based (Chao et al. 2014), for TD, PD and FD alike. Leave `NULL` for abundance data.
`n_units`	For incidence data, a column giving the number of units sampled at each site, constant within a site. Use it when some units detected no species and so do not appear in the data. By default the number of distinct units in the data is used.
`estimator`	Estimators to compute; by default all for the data type.
`k`	Cut-off for rare (ACE) or infrequent (ICE) species.

conf	Confidence level of the intervals.
nboot	Number of bootstrap samples for the standard errors (0 for none).
.by	<tidy-select> Optional grouping columns, kept in the output. Grouped data frames are also respected.

Value

A tibble with the grouping columns, the site column, estimator, observed, estimate, se, lower and upper.

References

Burnham, K. P. and Overton, W. S. (1978). Estimation of the size of a closed population when capture probabilities vary among animals. Biometrika, 65, 625-633.

Chao, A. (1984). Nonparametric estimation of the number of classes in a population. Scandinavian Journal of Statistics, 11, 265-270.

Chao, A. and Lee, S.-M. (1992). Estimating the number of classes via sample coverage. Journal of the American Statistical Association, 87, 210-217.

Chao, A., Ma, K. H., Hsieh, T. C. and Chiu, C.-H. (2016). SpadeR: Species-richness prediction and diversity estimation with R. R package.

Chiu, C.-H., Wang, Y.-T., Walther, B. A. and Chao, A. (2014). An improved nonparametric lower bound of species richness via a modified Good-Turing frequency formula. Biometrics, 70, 671-682. doi:10.1111/biom.12200

Colwell, R. K. (2013). EstimateS: Statistical estimation of species richness and shared species from samples. Version 9.

Lee, S.-M. and Chao, A. (1994). Estimating population size via sample coverage for closed capture-recapture models. Biometrics, 50, 88-97.

Smith, E. P. and van Belle, G. (1984). Nonparametric estimation of species richness. Biometrics, 40, 119-129.

Examples

```
div_richness(png_ants, .by = season, nboot = 20)
div_richness(png_ants, unit = bait, n_units = baits, .by = season, nboot = 20)
```

---

div_standardise *Standardise diversity by coverage or sample size*

Description

Compares sites at equal sample completeness (coverage) or equal sample size, by rarefying or extrapolating each site to that level (Chao and Jost 2012).

Usage

```
div_standardise(
  data,
  site = site,
  species = species,
  abundance = NULL,
  q = c(0, 1, 2),
  coverage = NULL,
  size = NULL,
  level = c("max_extrapolation", "min_observed"),
  conf = 0.95,
  nboot = 50,
  type = c("TD", "PD", "FD"),
  tree = NULL,
  reftime = NULL,
  pd_unit = c("mean", "total"),
  distance = NULL,
  tau = NULL,
  unit = NULL,
  n_units = NULL,
  .by = NULL
)
```

Arguments

data	A data frame in long format, with one row per record.
site, species	<tidy-select> The column holding site identifiers and the column holding species names.
abundance	<tidy-select> The column holding counts. If NULL (the default), a column called abundance is used when present, otherwise every row counts as one individual.
q	Orders of diversity: any of 0, 1 and 2.
coverage	Target sample coverage between 0 and 1. If both coverage and size are NULL, the target is chosen by level.
size	Target sample size. Supply either coverage or size.
level	Rule for the default target coverage: "max_extrapolation" or "min_observed"; see Details.
conf	Confidence level of the intervals.
nboot	Number of bootstrap replicates. Use 0 to skip them. For PD and FD the bootstrap community adds undetected species with estimated branch lengths (PD) or distances (FD); see Details in div_estimate().
type	Diversity dimension: "TD" (taxonomic, the default), "PD" (phylogenetic, needs tree) or "FD" (functional, needs distance). Give several, such as c("TD", "PD"), or "all" for all three, to analyse the dimensions together: the results are stacked in one tibble with a type column, and autoplot() shows them side

by side. All three are Hill numbers of the same order q, in units of effective numbers of species (Chao, Chiu and Jost 2014), and reproduce iNEXT.3D.

`tree` For `type = "PD"`, a rooted ape::phylo tree with branch lengths whose tip labels include every species.

`reftime` For `type = "PD"`, the reference time (tree depth) at which diversity is measured. Defaults to the age of the root of the tree pruned to the species in `data`.

`pd_unit` For `type = "PD"`, `"mean"` (the default) divides phylogenetic diversity by `reftime`, giving the effective number of lineages on the same scale as species; `"total"` gives effective total branch length.

`distance` For `type = "FD"`, a species-by-species distance matrix (or stats::dist object) with names matching the species, scaled to lie between 0 and 1, for example from Gower distances of traits.

`tau` For `type = "FD"`, the distance threshold below which species are treated as functionally identical. Defaults to the mean distance between two individuals in the pooled data. Use `"auc"` to integrate over all thresholds from 0 to 1 (Chao et al. 2019).

`unit` For incidence data, the column naming the sampling unit (a trap, bait, plot or survey day) in which each species was detected. Each site is then described by the number of units in which each species was found and the number of units sampled, and every estimate is incidence-based (Chao et al. 2014), for TD, PD and FD alike. Leave `NULL` for abundance data.

`n_units` For incidence data, a column giving the number of units sampled at each site, constant within a site. Use it when some units detected no species and so do not appear in the data. By default the number of distinct units in the data is used.

`.by` <tidy-select> Optional grouping columns, kept in the output. Grouped data frames are also respected.

Details

When neither `coverage` nor `size` is given, the target coverage of each group is set by `level`:

- `"max_extrapolation"` (default, the rule of `iNEXT::estimateD()`): the lowest coverage that every site reaches at twice its sample size, the largest extrapolation that is still reliable for richness.
- `"min_observed"`: the lowest observed coverage of any site, so the least complete site is used as observed and the others are rarefied (the most conservative choice; Chao and Jost 2012).

The target is reported in the `target` column. A site whose coverage curve cannot reach the target (for example one with a single singleton and no doubletons) is returned at its highest attainable coverage with a warning, so check `coverage` against `target`. iNEXT rounds the target coverage it prints, so results can differ from iNEXT in the second decimal when a rounded level is passed back to it.

Value

A `td_standardise` tibble with the grouping columns, the site column, `q`, `m`, `method`, `target` (the target coverage, or `NA` for a target size), `coverage` (coverage reached), `diversity`, `se`, `lower` and

upper. A warning is given when a site has to be extrapolated beyond twice its sample size, where richness estimates become unreliable.

Examples

```
set.seed(1)
div_standardise(png_ants, coverage = 0.95, .by = season, nboot = 10)

div_standardise(png_ants, size = 20, .by = season, nboot = 0)
```

`div_table` *Publication-ready tables*

Description

Formats a tidydiversity result as a table for a manuscript or report: rounds numbers, combines estimates and confidence limits into one cell in APA style, for example `12.34 [10.10, 14.50]`, renames columns to readable titles, and can spread the orders q into columns so each site takes one row. The result can be returned as a tibble, or rendered with knitr, gt or flextable for HTML, PDF or Word output.

Usage

```
div_table(
  x,
  digits = 2,
  ci = TRUE,
  ci_style = c("apa", "parentheses", "pm"),
  wide = FALSE,
  output = c("tibble", "kable", "gt", "flextable"),
  caption = NULL
)
```

Arguments

x	A tibble from `div_estimate()`, `div_standardise()`, `div_hill()`, `div_coverage()`, `beta_pair()` or `beta_multi()`.
digits	Number of decimal places.
ci	Combine the estimate with its confidence limits in one cell.
ci_style	Style of the interval: `"apa"` (`[lower, upper]`), `"parentheses"` (`(lower-upper)`) or `"pm"` (`+/- SE`).
wide	Spread the orders q into columns, one row per site.
output	`"tibble"`, `"kable"` (`knitr::kable()`), `"gt"` (`gt::gt()`) or `"flextable"` (`flextable::flextable()`).
caption	Optional table caption for the rendered formats.

Value

A tibble of formatted character columns, or a kable, gt or flextable object.

Examples

```
set.seed(1)
est <- div_estimate(png_ants, .by = season, nboot = 20)
div_table(est)
div_table(est, wide = TRUE, digits = 1)
div_table(est, output = "kable", caption = "Asymptotic diversity")
```

---

model_case_weights *Precision weights from estimated diversity*

Description

Asymptotic diversity estimates differ in precision between sites: a poorly sampled site has a large standard error. `model_case_weights()` turns standard errors from `div_estimate()` or `div_boot()` into inverse-variance weights, normalised to mean 1, as `hardhat::importance_weights()` for use with `workflows::add_case_weights()`.

Usage

```
model_case_weights(se, cap = 100)
```

Arguments

`se`	Numeric vector of standard errors.
`cap`	Largest allowed ratio between weights, to stop sites with an SE near zero from dominating. `Inf` for no cap.

Value

A `hardhat_importance_weights` vector.

Examples

```
est <- div_estimate(png_ants, q = 1, .by = season)
est$w <- model_case_weights(est$se)
est
```

`model_count` *Overdispersed count and proportion models (aods3-free)*

Description

Fits Poisson, quasi-Poisson, negative binomial, binomial or quasi-binomial GLMs with a tidy interface so callers never need **aods3**, and need **MASS** only when `family = "negbin"` (Suggests). Designed for the abundance / richness models in Moses et al. (2021), which switched to negative binomial after detecting overdispersion.

Usage

```
model_count(
  data,
  formula,
  family = c("poisson", "quasipoisson", "negbin", "binomial", "quasibinomial"),
  weights = NULL,
  offset = NULL,
  ...
)

## S3 method for class 'td_count_model'
tidy(x, ...)

## S3 method for class 'td_count_model'
glance(x, ...)
```

Arguments

`data`	A data frame (for example a site table of abundance or Hill richness).
`formula`	A model formula, such as `abundance ~ poly(elevation, 2) * litter`.
`family`	One of `"poisson"`, `"quasipoisson"`, `"negbin"`, `"binomial"`, `"quasibinomial"`.
`weights, offset`	Optional columns (bare names or expressions) passed to `stats::glm()` / `MASS::glm.nb()`.
`...`	Additional arguments forwarded to the fitter.
`x`	A `td_count_model` from `model_count()`.

Value

A `td_count_model` list with `fit`, `family`, `formula`, `data`, and dispersion diagnostics. `generics::tidy()` / `generics::glance()` methods are provided as `tidy.td_count_model()` and `glance.td_count_model()`.

References

Moses, J., Fayle, T. M., Novotny, V. and Klimes, P. (2021). Elevation and leaf litter interact in determining the structure of ant communities on a tropical mountain. *Biotropica*, 53, 906-919. doi:10.1111/btp.12914

Examples

```
ants <- png_ants |>
  dplyr::group_by(site, elevation) |>
  dplyr::summarise(richness = dplyr::n_distinct(species), .groups = "drop")
fit <- model_count(ants, richness ~ poly(elevation, 2), family = "poisson")
tidy(fit)
glance(fit)
```

`model_group_folds` *Grouped cross-validation folds for community data*

Description

Samples from the same site (or transect, plot, ...) are not independent, so random cross-validation folds leak information and overstate model skill (Roberts et al. 2017). `model_group_folds()` creates folds that keep every row of a group together, using `rsample::group_vfold_cv()`.

Usage

```
model_group_folds(data, group, v = NULL, ...)
```

Arguments

`data`	A data frame, typically one row per sampling unit.
`group`	<`tidy-select`> The blocking column, such as `site`.
`v`	Number of folds. Defaults to the number of groups (leave-one-group-out).
`...`	Passed to `rsample::group_vfold_cv()`.

Value

An `rset` object for use with tune.

References

Roberts, D. R. et al. (2017). Cross-validation strategies for data with temporal, spatial, hierarchical, or phylogenetic structure. *Ecography*, 40, 913-929. doi:10.1111/ecog.02881

Examples

```
units <- div_hill(png_ants, q = 0, .by = c(season, transect))
model_group_folds(units, group = site)
```

`ord_advise` *Which ordination and which test? Advice from the data and the question*

Description

`ord_advise()` helps choose an ordination method and the test that goes with it. It looks at the community data (number of sites and species, zeros, abundances or presence-absence, the length of the main gradient) and at the question asked, and recommends, with its reasons and the code to run:

- **unconstrained or constrained.** To *describe* the main patterns of composition, without a hypothesis about their causes, use an unconstrained (indirect) ordination: PCA, CA or NMDS; site variables can be fitted afterwards (arrows, contours) but are not part of the analysis. To *test* whether, and how much, measured variables explain composition, use a constrained (direct, canonical) ordination: RDA, CCA or distance-based RDA, with a permutation test.
- **linear or unimodal.** The length of the first axis of a detrended correspondence analysis, in standard-deviation units of species turnover, measures how heterogeneous the data are: below about 3, most species change monotonically and linear methods (PCA, RDA) fit; above about 4, species rise and fall along the gradient and unimodal methods (CA, CCA) fit; in between, either (ter Braak and Smilauer 2012). Hellinger-transformed PCA and RDA (Legendre and Gallagher 2001) work well whatever the length, and NMDS or distance-based RDA with Bray-Curtis or Jaccard dissimilarity suit any data.
- **the test.** Constrained ordinations are tested by permutation (overall, by axis and by term); with many candidate variables, by forward selection; differences between groups of sites by PERMANOVA, with a test of dispersion alongside; treatments followed over time by principal response curves; and the separate and shared effects of sets of variables by variation partitioning.

The advice is a starting point, not a rule: the question, which only the analyst knows, comes first.

Usage

```
ord_advise(
  data,
  site = site,
  species = species,
  abundance = NULL,
  formula = NULL,
  question = c("auto", "explore", "explain", "groups", "time", "partition"),
  treatment = NULL,
  time = NULL
)
```

Arguments

`data`	A data frame in long format, with one row per record.

site, species	<tidy-select> The column holding site identifiers and the column holding species names.
abundance	<tidy-select> The column holding counts. If NULL (the default), a column called abundance is used when present, otherwise every row counts as one individual.
formula	Optional one-sided formula of site variables (columns of data, one value per site), e.g. ~ elevation + habitat.
question	What is asked: "auto" (from the other arguments), "explore" (describe the main patterns), "explain" (test site variables), "groups" (compare groups of sites), "time" (treatments followed over time) or "partition" (separate the effects of sets of variables).
treatment, time	Optional columns for a treatment followed over time.

Value

A list of class td_ord_advice: checks (a tibble of what was measured in the data, its value and what it means), advice (a tibble of steps: step, recommendation, reason and code) and gradient_length. Printing it shows the advice.

References

ter Braak, C. J. F. and Smilauer, P. (2012). Canoco Reference Manual and User's Guide: Software for Ordination (version 5.0). Microcomputer Power.

Legendre, P. and Gallagher, E. D. (2001). Ecologically meaningful transformations for ordination of species data. Oecologia, 129, 271-280.

Legendre, P. and Legendre, L. (2012). Numerical Ecology, third edition. Elsevier.

Examples

```
data("dune", "dune.env", package = "vegan")
d <- community_long(cbind(site = rownames(dune), dune.env, dune),
  cols = dplyr::all_of(names(dune))
)
# Describe the meadows
ord_advise(d, abundance = abundance)
# Test management and soil
ord_advise(d, abundance = abundance, formula = ~ A1 + Moisture + Management)
```

---

ord_fit *Ordination of community data*

Description

Fits an unconstrained or constrained ordination to long community data and returns its scores as tidy tibbles. The ordination itself is computed by the vegan package, so results are identical to vegan's; tidydiversity adds the data handling, site covariates, tidy output and publication-quality plots (autoplot()).

Usage

```
ord_fit(
  data,
  site = site,
  species = species,
  abundance = NULL,
  method = c("pca", "ca", "dca", "pcoa", "nmds", "rda", "cca", "dbrda"),
  formula = NULL,
  env = NULL,
  transform = NULL,
  distance = "bray",
  scaling = c("species", "sites", "symmetric"),
  k = 2,
  permutations = 999,
  choices = 1:2,
  seed = NULL,
  ...
)
```

Arguments

data	A data frame in long format, with one row per record.
site, species	<tidy-select> The column holding site identifiers and the column holding species names.
abundance	<tidy-select> The column holding counts. If NULL (the default), a column called abundance is used when present, otherwise every row counts as one individual.
method	Ordination method; see Details.
formula	A one-sided formula of site variables, such as ~ elevation + season. Required for constrained methods; for unconstrained methods the variables are fitted passively.
env	A data frame of site variables with a column named like the site column. By default, site variables are taken from data: every column that is constant within each site.
transform	Transformation of the community matrix before the analysis: "hellinger" (the default for PCA and RDA, which otherwise overweight abundant species; Legendre and Gallagher 2001), "chord", "log" (log(x) + 1 for x > 0), "sqrt", "pa" (presence-absence) or "none" (the default for the other methods).
distance	Dissimilarity for "pcoa", "nmds" and "dbrda", passed to vegan::vegdist(), such as "bray" (the default) or "jaccard".
scaling	For PCA, CA, PCoA and the constrained methods, which scores to scale (ter Braak and Smilauer 2012): "species" (the default, vegan scaling 2) preserves the correlations between species and variables; "sites" (scaling 1) preserves the distances between sites; "symmetric" (scaling 3) is a compromise. vegan's numbers 1, 2 and 3 are accepted too.
k	For "nmds", the number of dimensions.

permutations	Permutations for the tests of passive variables: a number, or "exact" for complete enumeration of all n! orderings of the sites (feasible up to about 9 sites; with 7 sites, 5,040), which gives exact p-values.
choices	Axes on which passive variables are fitted and tested, as in vegan::envfit(). The default, the first two, matches the biplot; the axes used are reported in the envfit table.
seed	Optional random seed for the permutation tests.
...	Passed to the vegan function that fits the ordination.

Details

Unconstrained methods summarise the main gradients in species composition: principal components analysis ("pca"), correspondence analysis ("ca"), detrended correspondence analysis ("dca"), principal coordinates analysis ("pcoa") of a dissimilarity, and non-metric multidimensional scaling ("nmds"). Constrained methods find the gradients explained by environmental variables: redundancy analysis ("rda"), canonical correspondence analysis ("cca") and distance-based RDA ("dbrda"). Add Condition(x) to the formula for a partial analysis that removes the effect of x first.

For an unconstrained method, a formula fits the variables passively with vegan::envfit(), so they appear as supplementary arrows and centroids (as in CANOCO) without changing the ordination.

Value

A td_ordination object: a list with tibbles sites (site scores joined to the site variables), species, vectors (environmental arrows), centroids (factor levels) and axes (eigenvalues and proportion of variation explained), plus stress for NMDS, envfit for passive variables and fit, the vegan object. Use tidy() and glance() to extract them and ord_test() for permutation tests.

References

Legendre, P. and Gallagher, E. D. (2001). Ecologically meaningful transformations for ordination of species data. Oecologia, 129, 271-280. doi:10.1007/s004420100716

ter Braak, C. J. F. and Smilauer, P. (2012). Canoco reference manual and user's guide: software for ordination (version 5.0). Microcomputer Power, Ithaca, USA.

Oksanen, J. et al. vegan: Community Ecology Package. https://CRAN.R-project.org/package=vegan

Examples

```
# Tree-nesting ants: one sample per plot and subplot
ants <- png_tree_ants
ants$plot <- paste(ants$site, ants$subplot)

# Constrained: how much of the composition does elevation explain?
ord <- ord_fit(ants,
  site = plot, abundance = nests, method = "rda", formula = ~elevation
)
ord
```

```
ord_test(ord)

# Unconstrained NMDS with elevation fitted passively
nmds <- ord_fit(ants,
  site = plot, abundance = nests, method = "nmds", formula = ~elevation
)
tidy(nmds)
```

`ord_interpret` *Explain and interpret an ordination, its tests and its diagram*

Description

`ord_interpret()` turns the results of an ordination into plain sentences for the reader of a report: how much variation the axes hold (and whether they are worth reading, against the broken-stick expectation), what the permutation tests say, which way each site variable runs across the diagram, where the groups lie, which species define each end of the axes, how faithful an NMDS is (its stress), and how to read the diagram under the scaling used. It works for `ord_fit()` results, for principal response curves (`ord_prc()`) and for variation partitioning (`ord_varpart()`).

The sentences describe the results; whether they answer the question is for the analyst to judge. Fitted variables of an unconstrained ordination are a post hoc description, not a test of a hypothesis.

Usage

```
ord_interpret(x, ...)

## S3 method for class 'td_ordination'
ord_interpret(x, permutations = 999, n_species = 3, ...)

## S3 method for class 'td_prc'
ord_interpret(x, ...)

## S3 method for class 'td_varpart'
ord_interpret(x, ...)
```

Arguments

`x`	An `ord_fit()`, `ord_prc()` or `ord_varpart()` result.
`...`	Not used.
`permutations`	Number of permutations for the tests of a constrained ordination or of fitted variables.
`n_species`	Number of species named at each end of an axis.

Value

A list of class `td_ord_interpretation`: `sections` (a named list of character vectors, one per topic) and `tables` (the tables they are based on). Printing it shows the sentences.

References

Clarke, K. R. (1993). Non-parametric multivariate analyses of changes in community structure. Australian Journal of Ecology, 18, 117-143.

ter Braak, C. J. F. and Smilauer, P. (2012). Canoco Reference Manual and User's Guide: Software for Ordination (version 5.0). Microcomputer Power.

Examples

```
data("dune", "dune.env", package = "vegan")
d <- community_long(cbind(site = rownames(dune), dune.env, dune),
  cols = dplyr::all_of(names(dune))
)
o <- ord_fit(d,
  abundance = abundance, method = "rda", transform = "hellinger",
  formula = ~ A1 + Management
)
ord_interpret(o, permutations = 199)
```

`ord_prc` *Principal response curves*

Description

`ord_prc()` fits principal response curves (PRC; van den Brink and ter Braak 1999), a partial redundancy analysis that shows how the community of each treatment departs from a control over time, with time as a condition. Each treatment's departure at each time is one canonical coefficient, and each species has a weight: species with large positive weights follow the curves, species with large negative weights do the opposite. `autoplot()` draws the classic PRC diagram: the curves of the treatments against time, with the control as the zero line, and the species weights on a scale beside them.

Usage

```
ord_prc(
  data,
  treatment,
  time,
  site = site,
  species = species,
  abundance = NULL,
  control = NULL,
  transform = "log",
  permutations = 999
)

## S3 method for class 'td_prc'
```

```
autoplot(
  object,
  weights = 0.5,
  n_species = 20,
  palette = "okabe_ito",
  style = ord_style(),
  ...
)
```

Arguments

data	A data frame in long format, with one row per record.
treatment, time	Columns giving the treatment and the time (survey year or sampling date) of each site (sample).
site, species	<tidy-select> The column holding site identifiers and the column holding species names.
abundance	<tidy-select> The column holding counts. If NULL (the default), a column called abundance is used when present, otherwise every row counts as one individual.
control	Level of treatment used as the control; default the first level.
transform	Transformation of the abundances (as in ord_fit()); default "log" (log(x + 1)), usual for PRC.
permutations	Number of permutations for the test of the first axis; permutations are restricted within times.
object	An ord_prc() result.
weights	Species weights to show: those with an absolute weight of at least this.
n_species	At most this many species (the largest absolute weights) are labelled.
palette	Colours of the treatments.
style	Look of the diagram, from ord_style().
...	Not used.

Value

A list of class td_prc: curves (a tibble of treatment, time and coefficient), species (a tibble of species and weight), test (the permutation test of the first PRC axis), explained (share of the variation explained by treatment, after time) and the fitted vegan::prc() model.

References

van den Brink, P. J. and ter Braak, C. J. F. (1999). Principal response curves: analysis of time-dependent multivariate responses of biological community to stress. Environmental Toxicology and Chemistry, 18, 138-148.

Examples

```
# Chlorpyrifos in experimental ditches (van den Brink and ter Braak 1999):
# 12 ditches, 11 sampling weeks, five doses; abundances already log-transformed
data("pyrifos", package = "vegan")
ditch <- gl(12, 1, length = 132)
week <- gl(11, 12, labels = c(-4, -1, 0.1, 1, 2, 4, 8, 12, 15, 19, 24))
dose <- factor(rep(c(0.1, 0, 0, 0.9, 0, 44, 6, 0.1, 44, 0.9, 0, 6), 11))
pyr <- data.frame(site = paste(ditch, week, sep = "_"), dose, week, pyrifos)
pyr <- community_long(pyr, cols = dplyr::all_of(names(pyrifos)))
p <- ord_prc(pyr,
  treatment = dose, time = week, abundance = abundance,
  transform = "none", permutations = 99
)
p
ggplot2::autoplot(p, weights = 1)
```

---

ord_select *Forward selection of explanatory variables*

Description

Selects site variables for RDA (or CCA, db-RDA) by forward selection with the double stopping criterion of Blanchet et al. (2008): a variable enters only if it is significant *and* the adjusted R^2 of the model stays below that of the full model. Uses `vegan::ordiR2step()`.

Usage

```
ord_select(
  data,
  site = site,
  species = species,
  abundance = NULL,
  formula,
  env = NULL,
  method = c("rda", "cca", "dbrda"),
  transform = NULL,
  distance = "bray",
  alpha = 0.05,
  permutations = 999
)
```

Arguments

`data`	A data frame in long format, with one row per record.
`site, species`	<tidy-select> The column holding site identifiers and the column holding species names.

abundance	<tidy-select> The column holding counts. If NULL (the default), a column called abundance is used when present, otherwise every row counts as one individual.
formula	One-sided formula of all candidate site variables.
env	A data frame of site variables with a column named like the site column. By default, site variables are taken from data: every column that is constant within each site.
method	"rda", "cca" or "dbrda".
transform	Transformation of the community matrix before the analysis: "hellinger" (the default for PCA and RDA, which otherwise overweight abundant species; Legendre and Gallagher 2001), "chord", "log" (log(x) + 1 for x > 0), "sqrt", "pa" (presence-absence) or "none" (the default for the other methods).
distance	Dissimilarity for "pcoa", "nmds" and "dbrda", passed to vegan::vegdist(), such as "bray" (the default) or "jaccard".
alpha	Significance level for entry.
permutations	Permutations for the tests of passive variables: a number, or "exact" for complete enumeration of all n! orderings of the sites (feasible up to about 9 sites; with 7 sites, 5,040), which gives exact p-values.

Value

A tibble with one row per selected step: step, term, adj_r2 (cumulative), statistic and p.value, and the adjusted R^2 of the full model as attribute "adj_r2_full".

References

Blanchet, F. G., Legendre, P. and Borcard, D. (2008). Forward selection of explanatory variables. *Ecology*, 89, 2623-2632. doi:10.1890/070986.1

Examples

```
ants <- dplyr::mutate(png_ants, unit = paste(site, season, transect))
set.seed(1)
ord_select(ants, site = unit, formula = ~ elevation + season,
           permutations = 199)
```

---

ord_style *Style of ordination diagrams*

Description

ord_style() sets the look of every element of an ordination diagram drawn with autoplot(). The defaults follow the conventions of CANOCO 5 and vegan, tuned for print: filled site symbols with a thin outline (legible in colour and in grey), small crosses for species with italic names, heavier arrows for continuous environmental variables with bold names, triangles for factor centroids and hairlines through the origin. Change any argument; colours of site groups come from palette in autoplot(), and further changes can be made by adding ggplot2 scales and themes to the result.

Usage

```
ord_style(
  font = "serif",
  site_size = 3.4,
  site_stroke = 0.8,
  site_alpha = 1,
  site_outline = "grey15",
  site_fill = "white",
  site_shapes = c(21, 24, 22, 23, 25),
  site_label_colour = "grey10",
  site_fontface = "bold",
  species_colour = "grey45",
  species_label_colour = "grey40",
  species_shape = 3,
  species_size = 1.9,
  species_linewidth = 0.35,
  species_fontface = "italic",
  env_colour = "#1F78B4",
  env_linewidth = 0.9,
  env_fontface = "plain",
  env_stats = TRUE,
  path_colour = "grey60",
  path_linewidth = 0.35,
  centroid_shape = 17,
  centroid_size = 3,
  contour_colour = "#1B7837",
  contour_linewidth = 0.45,
  origin_colour = "grey60",
  origin_linewidth = 0.25,
  label_size = 2.8
)
```

Arguments

`font`	Font family of all text (`"serif"`, `"sans"` or any installed family).
`site_size`, `site_stroke`, `site_alpha`	Size, outline width and opacity of site symbols.
`site_outline`, `site_fill`	Outline colour of site symbols, and their fill when sites are not grouped.
`site_shapes`	Filled symbols (21 to 25) used for site classes.
`site_label_colour`	Colour of site labels.
`site_fontface`	Font face of site labels.
`species_colour`, `species_label_colour`	Colour of species symbols (or arrows) and of their names.
`species_shape`, `species_size`	Symbol and size of species points.

`species_linewidth`
: Line width of species arrows (`species_arrows = TRUE`).

`species_fontface`
: Font face of species names (`"italic"` for binomials).

`env_colour`, `env_linewidth`, `env_fontface`
: Colour, arrow width and font face of environmental variables and centroids.

`env_stats`
: Add the fit of each environmental variable (r squared and p from `envfit`) to its label.

`path_colour`, `path_linewidth`
: Colour and width of the `path` line.

`centroid_shape`, `centroid_size`
: Symbol and size of factor centroids.

`contour_colour`, `contour_linewidth`
: Colour and width of contours.

`origin_colour`, `origin_linewidth`
: Lines through the origin.

`label_size`
: Base size of labels (mm); variable labels are a little larger.

Value

A list for the `style` argument of `autoplot()`.

Examples

```
data("dune", "dune.env", package = "vegan")
d <- community_long(cbind(site = rownames(dune), dune.env, dune),
  cols = dplyr::all_of(names(dune))
)
o <- ord_fit(d, abundance = abundance, method = "pca", transform = "hellinger")
ggplot2::autoplot(o,
  colour = Management,
  style = ord_style(site_size = 3.5, species_colour = "grey20", label_size = 3.2)
)
```

---

ord_test *Permutation tests for constrained ordinations*

Description

Tests the significance of a constrained ordination (`ord_fit()` with `"rda"`, `"cca"` or `"dbrda"`) by permutation, with `vegan::anova.cca()`, and returns a tidy table.

Usage

```
ord_test(x, by = c("model", "axis", "terms", "margin"), permutations = 999)
```

Arguments

x	A td_ordination from a constrained method.
by	"model" (the whole model, the default), "axis" (each constrained axis), "terms" (each term, added sequentially) or "margin" (each term given the others). by = NULL, as in vegan::anova.cca(), means "model".
permutations	Number of permutations.

Value

A tibble with term, df, variance (or chi_square for CCA), statistic (pseudo-F) and p.value.

Examples

```
data("dune", "dune.env", package = "vegan")
d <- community_long(cbind(site = rownames(dune), dune.env, dune),
  cols = dplyr::all_of(names(dune))
)
ord <- ord_fit(d, abundance = abundance, method = "cca", formula = ~ A1 + Management)
ord_test(ord, permutations = 199)
ord_test(ord, by = "margin", permutations = 199)
```

---

ord_varpart *Variation partitioning*

Description

Partitions the variation in community composition among two to four groups of explanatory variables with adjusted R^2 (Borcard et al. 1992; Peres-Neto et al. 2006), using vegan::varpart(). Fractions that are testable (the unique fraction of each group) are tested by permutation with partial RDA.

Usage

```
ord_varpart(
  data,
  ...,
  site = site,
  species = species,
  abundance = NULL,
  env = NULL,
  transform = "hellinger",
  permutations = 999
)

## S3 method for class 'td_varpart'
autoplot(
```

```
  object,
  fill = c("#56B4E9", "#E69F00", "#009E73", "#CC79A7"),
  style = ord_style(),
  ...
)
```

Arguments

data	A data frame in long format, with one row per record.
...	Not used.
site, species	<tidy-select> The column holding site identifiers and the column holding species names.
abundance	<tidy-select> The column holding counts. If NULL (the default), a column called abundance is used when present, otherwise every row counts as one individual.
env	A data frame of site variables with a column named like the site column. By default, site variables are taken from data: every column that is constant within each site.
transform	Transformation before RDA; Hellinger by default.
permutations	Permutations for the tests of passive variables: a number, or "exact" for complete enumeration of all n! orderings of the sites (feasible up to about 9 sites; with 7 sites, 5,040), which gives exact p-values.
object	An ord_varpart() result.
fill	Fill colours of the sets.
style	Look of the diagram, from ord_style().

Value

A tibble of fractions with columns fraction, label (e.g. "[a] climate only"), adj_r2, testable and p.value.

References

Borcard, D., Legendre, P. and Drapeau, P. (1992). Partialling out the spatial component of ecological variation. *Ecology*, 73, 1045-1055. doi:10.2307/1940179

Peres-Neto, P. R. et al. (2006). Variation partitioning of species data matrices: estimation and comparison of fractions. *Ecology*, 87, 2614-2625. doi:10.1890/00129658(2006)87[2614:VPOSDM]2.0.CO;2

Examples

```
ants <- dplyr::mutate(png_ants, unit = paste(site, season, transect))
set.seed(1)
ord_varpart(ants, site = unit,
            elevation = ~elevation, season = ~season,
            permutations = 199)
```

phylo_combine_dist *Combine functional and phylogenetic distances*

Description

Combines a functional and a phylogenetic dissimilarity into one, as `sqrt(w * F^2 + (1 - w) * P^2)` after scaling each to a maximum of 1 (the approach of Cadotte et al. 2013 and de Bello et al. 2017). Only species present in both are kept.

Usage

```
phylo_combine_dist(functional, phylogenetic, weight = 0.5)
```

Arguments

`functional`, `phylogenetic`	`dist` objects labelled by species.
`weight`	Weight `w` of the functional part, between 0 and 1.

Value

A `dist` object.

References

Cadotte, M. W. et al. (2013). The ecology of differences: assessing community assembly with trait and evolutionary distances. *Ecology Letters*, 16, 1234-1244. doi:10.1111/ele.12161

phylo_dist *Phylogenetic distances between species*

Description

Cophenetic (patristic) distances from a phylogeny, scaled to a maximum of 1 by default. Use them as `dist` in `trait_rao()`, `trait_ses()` or `trait_fd_indices()` to measure phylogenetic instead of functional diversity, or combine them with trait distances in `phylo_combine_dist()`.

Usage

```
phylo_dist(tree, scale = TRUE)
```

Arguments

`tree`	A phylogeny of class `phylo` (ape); tip labels must match species names.
`scale`	Scale distances to a maximum of 1?

Value

A dist object labelled by tip labels.

`phylo_signal` *Phylogenetic signal of traits*

Description

Measures how strongly closely related species resemble each other in each trait, with Blomberg's K (Blomberg et al. 2003) or Pagel's lambda (Pagel 1999), using `phytools::phylosig()`. K = 1 and lambda = 1 match Brownian-motion evolution; values near 0 mean no phylogenetic signal.

Usage

```
phylo_signal(
  traits,
  tree,
  species = species,
  cols = tidyselect::everything(),
  method = c("K", "lambda"),
  nsim = 999
)
```

Arguments

`traits`	A data frame with a species column and numeric traits.
`tree`	A phylogeny of class `phylo` (ape); tip labels must match species names.
`species`	<tidy-select> The species column of `traits`.
`cols`	<tidy-select> Numeric trait columns.
`method`	`"K"` or `"lambda"`.
`nsim`	Number of randomisations for the test of K.

Value

A tibble with `trait`, `method`, `n_species` (with data and in the tree), `statistic` and `p.value` (randomisation test for K, likelihood-ratio test against lambda = 0 for lambda).

References

Blomberg, S. P., Garland, T. and Ives, A. R. (2003). Testing for phylogenetic signal in comparative data: behavioral traits are more labile. *Evolution*, 57, 717-745. doi:10.1111/j.0014-3820.2003.tb00285.x

Pagel, M. (1999). Inferring the historical patterns of biological evolution. *Nature*, 401, 877-884. doi:10.1038/44766

plot-partial *Partial-effect and two-predictor surface plots (visreg / plotmo style)*

Description

Publication-ready ggplot replacements for the **visreg** and **plotmo** workflow used in Moses et al. (2021). `plot_partial()` draws a one-predictor conditional (partial) effect with optional residuals and confidence band. `plot_surface()` draws a two-predictor response surface as a filled contour (the pattern that paper built by converting `visreg::visreg2d()` into ggplot by hand). No visreg or plotmo dependency.

Usage

```
plot_partial(
  model,
  x,
  data = NULL,
  by = NULL,
  residuals = FALSE,
  se = TRUE,
  level = 0.95,
  n = 100,
  ...
)

plot_surface(
  model,
  x,
  y,
  data = NULL,
  n = 40,
  type = c("contour", "persp"),
  points = TRUE,
  ...
)
```

Arguments

`model`	A fitted model (`lm`, `glm`, `negbin`, `gam`, or `model_count()` / `td_count_model`).
`x, y`	Focal predictor names (character) or bare names.
`data`	Optional data frame used to build the prediction grid. Defaults to `model.frame(model)`.
`by`	Optional conditioning variable for panels (partial plots).
`residuals`	Overlay partial residuals?
`se`	Draw a pointwise confidence band (partial) or skip (surface)?
`level`	Confidence level for the band.

n	Number of grid points per axis.
...	Passed to predict methods.
type	"contour" (filled contours; default) or "persp" (base graphics::persp() drawn to the current device and returned as a recorded plot list; prefer "contour" for publication ggplot output).
points	Overlay observed (x, y) points on the surface?

Value

A ggplot object (plot_partial, plot_surface with type = "contour"). For type = "persp", a list with the persp transformation matrix and the prediction grid (still no plotmo).

References

Moses, J., Fayle, T. M., Novotny, V. and Klimes, P. (2021). Elevation and leaf litter interact in determining the structure of ant communities on a tropical mountain. *Biotropica*, 53, 906-919. doi:10.1111/btp.12914

Examples

```
ants <- png_ants |>
  dplyr::group_by(site, elevation) |>
  dplyr::summarise(richness = dplyr::n_distinct(species), .groups = "drop")
ants$litter <- 1 + ants$elevation / 2000 + stats::rnorm(nrow(ants), 0, 0.3)
fit <- model_count(ants, richness ~ poly(elevation, 2) * litter,
                   family = "poisson")
plot_partial(fit, elevation, residuals = TRUE)
plot_surface(fit, elevation, litter)
```

---

plot_panel *Combine plots into a publication figure*

Description

plot_panel() arranges ggplots (for example several autoplot() results) into one multi-panel figure with patchwork. It adds panel tags (a, b, c, ...), collects shared legends and applies the tidydiversity theme. The result is an ordinary patchwork object, so you can still add patchwork::plot_annotation() or & theme changes.

Usage

```
plot_panel(
  ...,
  ncol = NULL,
  nrow = NULL,
  tag_levels = "a",
```

```
  tag_prefix = "",
  tag_suffix = "",
  guides = c("collect", "keep"),
  legend_position = "bottom",
  title = NULL,
  subtitle = NULL,
  caption = NULL,
  widths = NULL,
  heights = NULL
)
```

Arguments

`...`	ggplot objects, or one list of them.
`ncol`, `nrow`	Layout grid, passed to `patchwork::wrap_plots()`.
`tag_levels`	Panel tag style: `"a"`, `"A"`, `"1"`, `"i"`, `"I"`, or `NULL` for no tags.
`tag_prefix`, `tag_suffix`	Text around each tag, for example `"("` and `")"`.
`guides`	`"collect"` (default) merges identical legends; `"keep"` leaves them on each panel.
`legend_position`	Position of collected legends.
`title`, `subtitle`, `caption`	Figure annotations.
`widths`, `heights`	Relative panel sizes.

Value

A patchwork object.

See Also

`plot_save()`

Examples

```
r0 <- div_rarefy(png_ants, q = 0, .by = season, knots = 20)
r2 <- div_rarefy(png_ants, q = 2, .by = season, knots = 20)
plot_panel(ggplot2::autoplot(r0), ggplot2::autoplot(r2), ncol = 2)
```

plot_phylo *Plot a phylogeny with species abundances and phylogenetic diversity*

Description

Draws a rectangular phylogram of a `phylo` tree with ggplot2. With `data`, a column of dots per site beside the tips shows each species' relative abundance at that site. With `highlight`, the branches leading to the species found at one site are coloured: their summed length is that site's Faith's phylogenetic diversity (PD; Faith 1992), the quantity behind q = 0 phylogenetic Hill numbers (Chao et al. 2010).

Usage

```
plot_phylo(
  tree,
  data = NULL,
  site = site,
  species = species,
  abundance = NULL,
  highlight = NULL,
  tip_labels = TRUE,
  theme = theme_tidydiversity(grid = "none")
)
```

Arguments

`tree`	A rooted tree of class `phylo`, for example from `ape::read.tree()`. Without branch lengths every branch has length 1.
`data`	Optional data frame of species records by site.
`site`, `species`, `abundance`	Columns of `data`. Without `abundance`, each row counts as one record.
`highlight`	Optional site whose branches to colour.
`tip_labels`	Draw the tip labels (underscores become spaces).
`theme`	A ggplot2 theme, by default `theme_tidydiversity()`.

Value

A ggplot object.

References

Faith, D. P. (1992). Conservation evaluation and phylogenetic diversity. *Biological Conservation*, 61, 1-10. doi:10.1016/00063207(92)912013

Chao, A., Chiu, C.-H. and Jost, L. (2010). Phylogenetic diversity measures based on Hill numbers. *Philosophical Transactions of the Royal Society B*, 365, 3599-3609. doi:10.1098/rstb.2010.0272

Examples

```
# A taxonomic tree (family / genus / species) for the Mt Wilhelm frogs
frogs <- png_frogs
frogs$genus <- sub(" .*", "", frogs$scientific_name)
frogs$tip <- gsub(" ", "_", frogs$scientific_name)
tax <- unique(as.data.frame(frogs[c("family", "genus", "tip")]))
tax[] <- lapply(tax, factor)
tree <- ape::compute.brlen(ape::as.phylo(~ family / genus / tip, data = tax))
plot_phylo(tree, frogs, species = tip, highlight = "MW0200")
```

---

plot_save *Save a figure at journal sizes*

Description

A wrapper around `ggplot2::ggsave()` with journal column widths, a white background and 300 dpi by default. Works for ggplot and patchwork objects.

Usage

```
plot_save(
  plot,
  filename,
  size = c("single", "onehalf", "double"),
  width = NULL,
  height = NULL,
  units = c("mm", "cm", "in", "px"),
  dpi = 300,
  ...
)
```

Arguments

`plot`	A ggplot or patchwork object.
`filename`	File name; the extension sets the format (`.png`, `.pdf`, `.tiff`, `.svg`, ...).
`size`	`"single"` (85 mm wide), `"onehalf"` (120 mm) or `"double"` (175 mm) column width. Ignored if `width` is given.
`width, height`	Size in `units`. `height` defaults to 0.8 x width.
`units`	Units for `width` and `height` (`"mm"`, `"cm"`, `"in"`, or `"px"`).
`dpi`	Resolution for raster formats (default 300).
`...`	Passed to `ggplot2::ggsave()`.

Value

`filename`, invisibly.

See Also

`tidymapr::geo_save()` for map-oriented defaults (path, 180x140 mm).

Examples

```
r <- div_rarefy(png_ants, q = 0, .by = season, knots = 20)
f <- tempfile(fileext = ".png")
plot_save(ggplot2::autoplot(r), f, size = "single")
```

---

plot_trait_space *Plot functional trait space*

Description

Places species in a two-dimensional functional trait space, the first two axes of a principal coordinates analysis (PCoA) of their trait dissimilarities, as is usual for multidimensional functional diversity indices (Villeger et al. 2008). With `data`, each group of sites gets the convex hull of the species it holds, whose area is the two-dimensional functional richness, and its community-weighted mean position (CWM).

Usage

```
plot_trait_space(
  traits,
  data = NULL,
  species = species,
  group = NULL,
  abundance = NULL,
  distance = NULL,
  labels = FALSE,
  palette = "okabe_ito",
  theme = theme_tidydiversity(grid = "none")
)
```

Arguments

`traits`	A data frame with a species column and trait columns.
`data`	Optional data frame of species records by group.
`species`	The species column of `traits` and `data`.
`group`	Column of `data` defining the groups to compare, such as habitats.
`abundance`	Optional abundance column of `data`, used for the CWM.
`distance`	Optional species dissimilarities (a `dist` with species labels). By default Gower distances from `trait_dist()`.
`labels`	Label the species.
`palette`	Colours for the groups: `"okabe_ito"`, `"grey"` or a vector of colours.
`theme`	A ggplot2 theme, by default `theme_tidydiversity()`.

Value

A ggplot object.

References

Villeger, S., Mason, N. W. H. and Mouillot, D. (2008). New multidimensional functional diversity indices for a multifaceted framework in functional ecology. *Ecology*, 89, 2290-2301. doi:10.1890/071206.1

Examples

```
aravo <- data_aravo()
plot_trait_space(aravo$traits, aravo$community, group = ZoogD, abundance = cover)
```

---

plot_venn *Venn diagram of species shared between groups*

Description

Draws a Venn diagram of the species recorded in two to four groups of sites, such as habitats or seasons. Each region shows the number of species found in exactly that combination of groups, and optionally their share of all species. Circles are used for two and three groups and ellipses for four, so that every combination has its own region.

Usage

```
plot_venn(
  data,
  group,
  species = species,
  percent = TRUE,
  palette = "okabe_ito",
  theme = theme_tidydiversity(grid = "none")
)
```

Arguments

data	A data frame with one row per site (or sample) and species.
group	The column defining the groups (two to four levels).
species	The species column.
percent	Also show each region's share of all species.
palette	Colours for the groups: "okabe_ito", "grey" or a vector of colours.
theme	A ggplot2 theme, by default theme_tidydiversity().

Value

A ggplot object. The counts are stored in its `regions` attribute.

Examples

```
plot_venn(png_tree_ants, group = site)
plot_venn(wanang_caterpillars, group = site)
```

`png_ants` *Ants at nutrient baits along Mt Wilhelm, Papua New Guinea*

Description

Real field data: ground-foraging ants recorded at experimental nutrient baits along the Mt Wilhelm elevational gradient in Papua New Guinea, from 200 to 3700 m, in the dry and the wet season (Moses et al. 2023). Baits offering six nutrient types were placed along transects at eight elevations 500 m apart. Ants were found up to 2200 m in both seasons and at 2700 m only once, in the dry season (one species at three baits); that record is left out so both seasons cover the same five elevations (200 to 2200 m) and can be compared directly. Each row is one species recorded at one bait. Only the Papua New Guinea part of the published cross-continental dataset is included, with the lead author's permission.

Usage

```
png_ants
```

Format

A tibble with one row per bait and ant species, and 12 columns:

site Elevation band, such as `"MW0200"` for 200 m.

elevation Elevation band in metres.

season Sampling season, `"dry"` or `"wet"`.

transect Transect number within the elevation band.

bait Bait identifier: the sampling unit for incidence-based estimates (`unit = bait`).

baits Number of baits set at that elevation and season, including baits no ant visited (`n_units = baits`).

bait_type Nutrient offered: `h2o`, `nacl`, `cho` (sucrose), `amino_acid`, `cho_amino_acid` or `lipid`.

species Ant morphospecies code.

genus, subfamily Ant genus and subfamily.

guild Trophic guild.

workers Number of workers at the bait.

Details

Because ants recruit to baits, worker counts measure activity rather than independent individuals. tidydiversity therefore counts each row (one species at one bait) as one occurrence when no abundance column is given, which is the default here: diversity is estimated from the number of baits at which each species was recorded.

Source

Mottl, O., Moses, J. and Klimes, P. (2023). Data and code for "Nutrient use by tropical ant communities varies among three extensive elevational gradients: a cross-continental comparison". Zenodo. doi:10.5281/zenodo.8191946 (open access). Built by `data-raw/png_ants.R` in the package source.

References

Moses, J., Peters, M. K., Tiede, Y., Mottl, O., Donoso, D. A., Farwig, N., Fayle, T. M., Novotny, V., Sanders, N. J. and Klimes, P. (2023). Nutrient use by tropical ant communities varies among three extensive elevational gradients: a cross-continental comparison. *Global Ecology and Biogeography*, 32, 2212-2229. doi:10.1111/geb.13757

Examples

```
png_ants
div_hill(png_ants, q = 0, .by = season)

# Incidence-based: baits as sampling units, including empty baits
div_estimate(png_ants, unit = bait, n_units = baits, .by = season, nboot = 0)
```

---

png_frogs *Frogs along the Mt Wilhelm elevational gradient, Papua New Guinea*

Description

Real field data: 3390 frogs of 55 species recorded at eight sites from 200 to 3700 m on Mt Wilhelm, Papua New Guinea (Dahl et al. 2024). Species not formally described carry a provisional name such as `"Copiula sp 1"`.

Usage

```
png_frogs
```

Format

A tibble with one row per site and species recorded, and seven columns:

site Site, such as `"MW0200"` for 200 m.

elevation Elevation in metres.

species Species code.

scientific_name Scientific or provisional name.

family Frog family.

reproduction Reproductive mode, `"terrestrial"` (direct development) or `"aquatic"`.

abundance Number of individuals recorded.

Source

Dahl, C., Richards, S. J., Basien, I., Mungkaje, A. J. and Novotny, V. (2023). Data from: Local and regional diversity of frog communities along an extensive rainforest elevation gradient in Papua New Guinea. Dryad. doi:10.5061/dryad.3n5tb2rqf, released under CC0 1.0. Built by `data-raw/png_frogs.R`.

References

Dahl, C., Richards, S. J., Basien, I., Mungkaje, A. J. and Novotny, V. (2024). Local and regional diversity of frog communities along an extensive rainforest elevation gradient in Papua New Guinea. *Biotropica*, 56, 90-97. doi:10.1111/btp.13283

Examples

```
png_frogs
div_estimate(png_frogs, nboot = 0)
```

---

png_tree_ants — *Tree-nesting ants in rainforest plots at three elevations, Papua New Guinea*

Description

Real field data: every ant nest in 1254 trees within continuous plots of primary rainforest at Wanang (200 m), Numba (900 m) and Yawan (1800 m), Papua New Guinea (Plowman et al. 2020). Only the 769 trees with at least one nest appear. Each row is one ant species nesting in one tree.

Usage

```
png_tree_ants
```

Format

A tibble with one row per tree and ant species, and nine columns:

site Plot: `"Wanang"`, `"Numba"` or `"Yawan"`.

elevation Elevation of the plot in metres.

subplot Subplot within the plot.

tree Tree identifier.

tree_family, tree_genus Host tree family and genus.

dbh Tree diameter at breast height (cm).

species Ant morphospecies code.

nests Number of nests of the species in the tree.

Source

Plowman, N. S., Mottl, O., Novotny, V., Idigel, C., Philip, F. J. and Klimes, P. (2019). Data from: Nest microhabitats and tree size mediate shifts in ant community structure across elevation in tropical rainforest canopies. Dryad. doi:10.5061/dryad.2fqz612k6, released under CC0 1.0. Built by `data-raw/png_tree_ants.R`.

References

Plowman, N. S., Mottl, O., Novotny, V., Idigel, C., Philip, F. J., Rimandai, M. and Klimes, P. (2020). Nest microhabitats and tree size mediate shifts in ant community structure across elevation in tropical rainforest canopies. *Ecography*, 43, 431-442. doi:10.1111/ecog.04730

Examples

```
png_tree_ants
# Nest counts per plot, trees pooled
div_estimate(png_tree_ants, site = site, abundance = nests, nboot = 0)
# Turnover between trees within each plot
beta_multi(png_tree_ants, site = tree, .by = site)
```

---

portal_rodents *Rodent captures at the Portal Project, Arizona*

Description

Real field data: annual counts of rodent captures on the 24 experimental plots of the Portal Project, a long-term study in the Chihuahuan Desert near Portal, Arizona, USA, from 2016 to 2019. Since 2015 plots have been unmanipulated controls, kangaroo rat (*Dipodomys*) exclosures, or removals of all rodents. Only rodent species that are census targets and identified to species are kept. Each plot and year is one site.

Usage

```
portal_rodents
```

Format

A tibble with one row per site and species captured, and seven columns:

site Site identifier, plot and year, such as `"P01-2016"`.

plot Plot number, 1 to 24.

year Census year.

treatment Experimental treatment: `control`, `krat_exclosure` or `removal`.

species Two-letter species code.

scientific_name Scientific name.

abundance Number of captures in the year.

Source

PortalData, Weecology, https://github.com/weecology/PortalData, released under the CC0 1.0 public domain dedication. Built by `data-raw/portal_rodents.R` in the package source.

References

Ernest, S. K. M., Yenni, G. M., Allington, G., Bledsoe, E. K., Christensen, E. M., Diaz, R. M. et al. (2018). The Portal Project: a long-term study of a Chihuahuan desert ecosystem. bioRxiv. doi:10.1101/332783

Examples

```
portal_rodents
div_hill(portal_rodents, q = 0, .by = c(treatment, year))
```

`range_edges` *Elevational (or other gradient) ranges of species*

Description

`range_edges()` summarises where each species occurs along a gradient, such as elevation: the lower and upper edges (lowest and highest site of occurrence), the midpoint, the breadth and the abundance-weighted optimum (mean position of individuals). Species found at the first or last sampled position have ranges that may extend beyond the transect; `truncated_lower` and `truncated_upper` flag them (Colwell and Hurtt 1994), as their edges and breadth are underestimated.

Usage

```
range_edges(
  data,
  gradient,
  site = site,
  species = species,
  abundance = NULL,
  .by = NULL
)
```

Arguments

data	A data frame in long format, with one row per record.
gradient	Column giving the position of each site on the gradient, such as elevation in metres.
site, species	<tidy-select> The column holding site identifiers and the column holding species names.
abundance	<tidy-select> The column holding counts. If NULL (the default), a column called abundance is used when present, otherwise every row counts as one individual.
.by	<tidy-select> Optional grouping columns, kept in the output. Grouped data frames are also respected.

Value

A tibble with one row per species (and group): species, lower, upper, midpoint, breadth, optimum, n_sites, abundance, truncated_lower, truncated_upper.

References

Colwell, R. K. and Hurtt, G. C. (1994). Nonbiological gradients in species richness and a spurious Rapoport effect. American Naturalist, 144, 570-595.

Examples

```
range_edges(png_frogs, abundance = abundance, gradient = elevation)
```

---

range_mde *Mid-domain null model of richness along a gradient*

Description

If species' ranges are placed at random within a bounded gradient, they overlap most in the middle, so richness peaks at mid-gradient even with no environmental cause: the **mid-domain effect** (Colwell and Hurtt 1994; Colwell and Lees 2000). range_mde() builds this null model from the observed ranges, in the spirit of RangeModel (Colwell 2008), and compares it with the observed richness at each site.

Each species keeps its observed range size (the number of consecutive sampled sites it spans, after range_through() interpolation), and its position is randomised:

- model = "uniform": every position where the range fits is equally likely (Colwell and Hurtt 1994). The expected richness is computed exactly; the interval comes from nsim randomisations.
- model = "spread": the range starts at a random site and spreads to random neighbouring sites (the "spreading dye" model of Jetz and Rahbek 2001, as in the rangemodelR package), which places more ranges near the ends than the uniform model.

Sites are treated as positions in order along the gradient, so the model assumes the sampled sites represent the domain evenly.

Usage

```
range_mde(
  data,
  gradient,
  site = site,
  species = species,
  model = c("uniform", "spread"),
  nsim = 999,
  conf = 0.95,
  interpolate = TRUE,
  .by = NULL
)

## S3 method for class 'td_mde'
glance(x, ...)
```

Arguments

`data`	A data frame in long format, with one row per record.
`gradient`	Column with the position of each site on the gradient (one value per site).
`site, species`	<tidy-select> The column holding site identifiers and the column holding species names.
`model`	`"uniform"` or `"spread"`.
`nsim`	Number of randomisations.
`conf`	Width of the simulation interval.
`interpolate`	Apply `range_through()` first (recommended).
`.by`	<tidy-select> Optional grouping columns, kept in the output. Grouped data frames are also respected.
`x`	A `range_mde()` result.
`...`	Not used.

Value

A tibble of class `td_mde` with the grouping columns, the site and gradient columns, `observed` richness, `expected` richness under the null model, its `sd`, `lower` and `upper` simulation limits, and `outside` (`"above"`, `"below"` or `"within"`). Attributes `r2` (the variance in observed richness explained by the null model, one value per group) and `p_value` (a global test: the proportion of randomisations that deviate from the expectation at least as much as the observed richness) summarise the fit; see `glance()`.

References

Colwell, R. K. and Hurtt, G. C. (1994). Nonbiological gradients in species richness and a spurious Rapoport effect. American Naturalist, 144, 570-595. doi:10.1086/285695

Colwell, R. K. and Lees, D. C. (2000). The mid-domain effect: geometric constraints on the geography of species richness. Trends in Ecology & Evolution, 15, 70-76. doi:10.1016/S0169-5347(99)01767X

Colwell, R. K. (2008). RangeModel: tools for exploring and assessing geometric constraints on species richness (the mid-domain effect) along transects. Ecography, 31, 4-7. doi:10.1111/j.2008.09067590.05347.x

Jetz, W. and Rahbek, C. (2001). Geometric constraints explain much of the species richness pattern in African birds. Proceedings of the National Academy of Sciences, 98, 5661-5666. doi:10.1073/pnas.091100998

Examples

```
set.seed(1)
mde <- range_mde(png_frogs, gradient = elevation, nsim = 199)
mde
glance(mde)
```

`range_partition` *Geometry or environment? Partitioning richness along a gradient*

Description

`range_partition()` splits the variation in observed richness along a gradient into the part explained by the mid-domain null model (geometry), the part explained by environmental variables, the part they share, and the unexplained rest. It is variation partitioning (Borcard et al. 1992; Peres-Neto et al. 2006) with adjusted R^2, applied to the debate on whether geometric constraints or the environment drive richness gradients (Colwell et al. 2004; Dunn et al. 2007).

Unique fractions are tested with an F test of the nested linear models. With few sites (as on most single-mountain transects) tests have little power and adjusted fractions can be negative (read negative values as zero); report them as descriptive. Sites along a gradient are also spatially autocorrelated, which makes the p-values too small.

Usage

```
range_partition(mde, env, data = NULL)
```

Arguments

`mde`	A `range_mde()` result.
`env`	A formula of environmental predictors, e.g. `~ temperature`, evaluated in `mde` joined with `data`.
`data`	Optional site-level data frame with the environmental variables and the site (and grouping) columns of `mde`.

Value

A tibble with the grouping columns and, for each fraction (`geometry only`, `environment only`, `shared`, `residual`), `adj_r2` and, for the unique fractions, `p_value`.

References

Borcard, D., Legendre, P. and Drapeau, P. (1992). Partialling out the spatial component of ecological variation. Ecology, 73, 1045-1055.

Colwell, R. K., Rahbek, C. and Gotelli, N. J. (2004). The mid-domain effect and species richness patterns: what have we learned so far? American Naturalist, 163, E1-E23. doi:10.1086/382056

Dunn, R. R., McCain, C. M. and Sanders, N. J. (2007). When does diversity fit null model predictions? Scale and range size mediate the mid-domain effect. Global Ecology and Biogeography, 16, 305-312.

Examples

```
set.seed(1)
mde <- range_mde(png_frogs, gradient = elevation, nsim = 199)
range_partition(mde, ~elevation)
```

`range_shift_resurvey` *Range shifts between two surveys of the same gradient*

Description

`range_shift_resurvey()` compares species' ranges in a historical survey and a resurvey of the same gradient (for example the same elevational transect decades apart; Moritz et al. 2008; Freeman and Freeman 2014). For each species found in both surveys it gives the shift of the lower edge, the upper edge, the midpoint and the abundance-weighted optimum, and the change in breadth; species found in only one survey are listed as `"lost"` or `"new"`.

`range_shift_test()` asks whether species shifted on average. Species share a gradient and survey design, but their shifts are treated as the units of replication, as in published resurveys: it gives the mean shift of each measure, a bootstrap confidence interval resampling species, and an exact two-sided sign test of whether more species moved up than down. Shifts of edges at the end of the sampled gradient are unreliable, so by default edges flagged as truncated in either survey are left out of the test of that edge.

Usage

```
range_shift_resurvey(
  before,
  after,
  gradient,
  site = site,
  species = species,
  abundance = NULL
```

```
)

range_shift_test(
  x,
  nboot = 999,
  conf = 0.95,
  drop_truncated = TRUE,
  seed = NULL
)
```

Arguments

before, after	Long-format data of the two surveys, with the same columns.
gradient	Column giving the position of each site on the gradient, such as elevation in metres.
site, species	<tidy-select> The column holding site identifiers and the column holding species names.
abundance	<tidy-select> The column holding counts. If NULL (the default), a column called abundance is used when present, otherwise every row counts as one individual.
x	Result of range_shift_resurvey().
nboot	Number of bootstrap resamples of species.
conf	Confidence level.
drop_truncated	Leave out edges truncated at the ends of the sampled gradient?
seed	Optional random seed.

Value

range_shift_resurvey(): a tibble with one row per species: species, status ("both", "lost", "new"), the edges in both surveys (lower_before, lower_after, ...), the shifts (shift_lower, shift_upper, shift_midpoint, shift_optimum, change_breadth) and truncated (an edge truncated in either survey).

range_shift_test(): a tibble with one row per measure: measure, n (species), mean, lower, upper (bootstrap interval), n_up, n_down, p_sign (exact sign test).

References

Freeman, B. G. and Class Freeman, A. M. (2014). Rapid upslope shifts in New Guinean birds illustrate strong distributional responses of tropical montane species to global warming. PNAS, 111, 4490-4494.

Moritz, C. et al. (2008). Impact of a century of climate change on small-mammal communities in Yosemite National Park, USA. Science, 322, 261-264.

Examples

```
# Two virtual surveys: every species moves 100 m upslope
before <- png_frogs
after <- png_frogs
after$elevation <- after$elevation + 100
shifts <- range_shift_resurvey(before, after,
  gradient = elevation, abundance = abundance
)
range_shift_test(shifts, nboot = 199, seed = 1)
```

`range_shift_validate` *Validate projected range shifts against resurveys (hindcasting)*

Description

A range-shift model earns trust when it predicts shifts that have already happened. `range_shift_validate()` compares the shifts observed between a historical survey and a resurvey (`range_shift_resurvey()`) with the shifts a model predicted for the same species and period, run from the climate of the first survey to that of the second (hindcasting).

For each measure (lower edge, upper edge, midpoint, optimum) it reports:

- `bias`: mean of predicted minus observed shift (m);
- `mae`, `rmse`: mean absolute and root mean square error (m);
- `r`, `rho`: Pearson and Spearman correlation across species;
- `slope`: calibration slope of observed on predicted (1 is ideal; below 1, the model overstates differences between species);
- `direction`: share of species whose shift has the predicted sign;
- `tracking`: mean observed shift divided by mean predicted shift, the share of the expected shift that species realised (1 = full tracking; below 1, a lag), with a bootstrap interval resampling species.

`range_shift_expected()` gives the simplest prediction, the isotherm shift $\Delta T/\gamma$ for warming ΔT and lapse rate γ (Colwell et al. 2008; Freeman and Freeman 2014). It is the baseline any model must beat.

Usage

```
range_shift_validate(
  observed,
  predicted,
  drop_truncated = TRUE,
  nboot = 999,
  conf = 0.95,
  seed = NULL
)

range_shift_expected(warming, lapse_rate)
```

Arguments

observed	Result of range_shift_resurvey(), or a data frame with species and observed shift_* columns.
predicted	Data frame with species and predicted shifts, in columns named like those of observed (shift_lower, shift_upper, shift_midpoint, shift_optimum), or a single column shift used for every measure.
drop_truncated	Leave out edges truncated at the ends of the sampled gradient (needs the truncated_* columns of observed).
nboot	Bootstrap resamples for the tracking interval.
conf	Confidence level.
seed	Optional random seed.
warming	Warming in degrees C between the surveys (or scenarios).
lapse_rate	Temperature lapse rate in degrees C per km.

Value

range_shift_validate(): a tibble with one row per measure.

range_shift_expected(): the expected upslope shift in metres.

References

Colwell, R. K., Brehm, G., Cardelús, C. L., Gilman, A. C. and Longino, J. T. (2008). Global warming, elevational range shifts, and lowland biotic attrition in the wet tropics. Science, 322, 258-261.

Freeman, B. G. and Class Freeman, A. M. (2014). Rapid upslope shifts in New Guinean birds illustrate strong distributional responses of tropical montane species to global warming. PNAS, 111, 4490-4494.

Examples

```
# Virtual resurvey: species realise 60% of a 200 m expected shift
before <- png_frogs
after <- png_frogs
after$elevation <- after$elevation + 120
obs <- range_shift_resurvey(before, after, gradient = elevation, abundance = abundance)
pred <- data.frame(species = obs$species, shift = range_shift_expected(1, 5))
range_shift_validate(obs, pred, nboot = 199, seed = 1)
```

range_through *Range-through interpolation along a gradient*

Description

range_through() assumes that a species recorded at two positions on a gradient (elevation, latitude, depth) is also present at every sampled position between them, and adds those records. This is the standard first step for elevational and latitudinal richness patterns (Grytnes and Vetaas 2002; Colwell 2008), because a species missed at an intermediate site is far more likely an undetected than an absent one.

Interpolation fills gaps only *inside* each species' observed range, so it raises richness most at mid-gradient sites. It therefore creates part of a mid-domain peak by itself; compare the result with range_mde(), which applies the same interpolation to its null model.

Usage

```
range_through(data, gradient, site = site, species = species, .by = NULL)
```

Arguments

data	A data frame in long format, with one row per record.
gradient	Column with the position of each site on the gradient (one value per site).
site, species	<tidy-select> The column holding site identifiers and the column holding species names.
.by	<tidy-select> Optional grouping columns, kept in the output. Grouped data frames are also respected.

Value

A tibble with the grouping columns, the site and gradient columns, species and interpolated (TRUE for added records), one row per species and site where it is present or interpolated.

References

Colwell, R. K. (2008). RangeModel: tools for exploring and assessing geometric constraints on species richness (the mid-domain effect) along transects. Ecography, 31, 4-7. doi:10.1111/j.2008.09067590.05347.x

Grytnes, J.-A. and Vetaas, O. R. (2002). Species richness and altitude: a comparison between null models and interpolated plant species richness along the Himalayan altitudinal gradient, Nepal. American Naturalist, 159, 294-304. doi:10.1086/338542

Examples

```
rt <- range_through(png_frogs, gradient = elevation)
dplyr::count(rt, site, elevation, interpolated)
```

stems *ForestGEO-style stem and tree census verbs*

Description

Tidyverse-style helpers for ForestGEO / CTFS plot inventories. They accept data frames with the usual columns (`plot`, `census`, `treeID`, `stemID`, `gx`, `gy`, `dbh`, `species` / `sp`, `status`, optionally `date`) and return tibbles. Callers stay inside tidydiversity instead of using the fgeo or CTFS packages.

Typical workflow: standardise names with `stems_as_trees()` (main stem per tree) or keep stems; summarise `stems_abundance()` and `stems_basal_area()`; size classes with `stems_size_class()`; demography between two censuses with `stems_demography()`; feed species counts to `div_hill()` via `stems_as_community()`.

Spatial stem verbs (quadrats, habitat joins, neighbourhoods, the torus-translation habitat test) and stem maps over topography are in the companion package tidymapr (`tidymapr::stems_quadrat()`, `tidymapr::stems_habitat_test()`, `tidymapr::plot_topo()`).

Basal area for one stem is `\eqn{\pi/4 \times \mathrm{dbh}^2}` (Condit 1998; fgeo.analyze). With `dbh_unit = "cm"` the result is cm^2; with `"mm"` it is mm^2. Convert afterwards if you need m^2 ha^{-1}.

Usage

```
stems_as_trees(data, hom = NULL)

stems_abundance(
  data,
  .by = NULL,
  alive_only = TRUE,
  alive = "A",
  level = c("tree", "stem")
)

stems_basal_area(
  data,
  .by = NULL,
  dbh_unit = c("mm", "cm"),
  alive_only = TRUE,
  alive = "A"
)

stems_size_class(
  data,
  breaks = c(10, 20, 50, 100, Inf),
  labels = NULL,
  dbh_unit = c("mm", "cm"),
  .by = NULL,
  alive_only = TRUE,
```

```
  alive = "A"
)

stems_demography(
  census1,
  census2,
  mindbh = 0,
  alive = "A",
  time = NULL,
  growth = c("absolute", "relative"),
  dbh_unit = c("mm", "cm"),
  .by = NULL
)

stems_as_community(
  data,
  site = NULL,
  alive_only = TRUE,
  alive = "A",
  level = c("tree", "stem")
)
```

Arguments

`data`	A ForestGEO-like data frame.
`hom`	Optional height-of-measure column; the largest HOM row is kept when several measures share a stem.
`.by`	Optional grouping columns (species, quadrat, habitat).
`alive_only`	Keep only stems/trees whose `status` is in `alive`?
`alive`	Character codes of `status` that mean alive (default `"A"`).
`level`	`"tree"` (collapse to one row per `treeID`) or `"stem"`.
`dbh_unit`	`"mm"` or `"cm"` for the dbh column.
`breaks`	Breakpoints for DBH size classes (in the same unit as dbh).
`labels`	Optional labels for size classes.
`census1, census2`	Two census tables for demography (tree-level preferred).
`mindbh`	Minimum DBH included in demography counts.
`time`	Mean census interval in years. When `NULL`, estimated from `date` columns when present, otherwise required.
`growth`	`"absolute"` for (dbh2-dbh1)/time or `"relative"` for (log dbh2 - log dbh1)/time.
`site`	Column to use as the site identifier in the community table (default `quadrat`, as added by `tidymapr::stems_quadrat()`, or `plot`).

References

Condit, R. (1998). *Tropical Forest Census Plots*. Springer. Anderson-Teixeira, K. J. et al. (2015). CTFS-ForestGEO: a worldwide network monitoring forests in an era of global change. *Global Change Biology*, 21, 528-549. doi:10.1111/gcb.12712. Condit, R. et al. (2014). Data and database standards for permanent forest plots in a global network. *Forest Ecology and Management*. ForestGEO (Smithsonian Tropical Research Institute). Analysis conventions also follow the fgeo.analyze package (rOpenSci / ForestGEO).

Examples

```
# The column layout of a ForestGEO stem table (illustrative rows; use
# your own census, or tidymapr::geo_example_stems())
stems <- tibble::tibble(
  treeID = 1:6,
  stemID = 1:6,
  gx = c(12.1, 40.5, 77.0, 23.9, 61.2, 88.4),
  gy = c(8.3, 55.0, 31.7, 90.2, 14.6, 66.1),
  dbh = c(105, 342, 58, 220, 77, 410),
  species = c("Pometia_pinnata", "Intsia_bijuga", "Pometia_pinnata",
    "Pterocarpus_indicus", "Intsia_bijuga", "Pometia_pinnata"),
  status = "A"
)
stems_abundance(stems, .by = species)
stems_basal_area(stems, dbh_unit = "mm", .by = species)
```

---

step_hill *Hill number features in a recipe*

Description

`step_hill()` is a recipes step that replaces wide species columns (one column per species, abundance or incidence values) with Hill numbers of chosen orders, so community diversity can be used as a predictor in tidymodels workflows.

Usage

```
step_hill(
  recipe,
  ...,
  q = c(0, 1, 2),
  prefix = "hill_q",
  keep_original_cols = FALSE,
  role = "predictor",
  trained = FALSE,
  columns = NULL,
  skip = FALSE,
  id = NULL
)
```

Arguments

recipe	A recipe.
...	Selectors for the species columns.
q	Orders of diversity; one output column per order, named prefix + q.
prefix	Prefix for the new columns.
keep_original_cols	Keep the species columns?
role, trained, columns, skip, id	Standard recipe step arguments.

Value

An updated recipe.

Examples

```
wide <- data.frame(elev = c(200, 700, 1200),
                   sp1 = c(5, 0, 2), sp2 = c(1, 3, 0), sp3 = c(2, 2, 2))
recipes::recipe(elev ~ ., data = wide) |>
  step_hill(dplyr::starts_with("sp"), q = c(0, 2)) |>
  recipes::prep() |>
  recipes::bake(new_data = NULL)
```

---

theme_tidydiversity *Publication theme and colour palettes*

Description

`theme_tidydiversity()` is a clean ggplot2 theme for journal figures: white background, light horizontal grid, black text, a legend at the bottom and plain facet strips. Every autoplot() method uses it by default, and you can add it to any ggplot.

`plot_palette()` returns colour-blind-safe colours: the Okabe-Ito palette (Okabe and Ito 2008) by default, switching to the viridis palette when more than eight colours are needed, or a greyscale ramp for journals that print in black and white. `scale_colour_tidydiversity()` and `scale_fill_tidydiversity()` are the matching discrete ggplot2 scales.

Usage

```
theme_tidydiversity(
  base_size = 11,
  base_family = "",
  legend_position = "bottom",
  grid = c("y", "x", "xy", "none")
)
```

```
plot_palette(n, palette = c("okabe_ito", "grey"))

scale_colour_tidydiversity(palette = c("okabe_ito", "grey"), ...)

scale_fill_tidydiversity(palette = c("okabe_ito", "grey"), ...)
```

Arguments

base_size	Base font size in points.
base_family	Base font family.
legend_position	Legend position, passed to ggplot2::theme().
grid	Draw major grid lines: "y", "x", "xy" or "none".
n	Number of colours.
palette	"okabe_ito" or "grey".
...	Passed to ggplot2::discrete_scale().

Value

A ggplot2 theme, a character vector of colours or a ggplot2 scale.

References

Okabe, M. and Ito, K. (2008). Color universal design (CUD): how to make figures and presentations that are friendly to colour-blind people. https://jfly.uni-koeln.de/color/

Examples

```
library(ggplot2)
plot_palette(3)
ggplot(mtcars, aes(wt, mpg, colour = factor(cyl))) +
  geom_point() +
  scale_colour_tidydiversity() +
  theme_tidydiversity(base_size = 9)
```

---

tidy-tidydiversity *Tidy and model-ready output*

Description

Every tidydiversity result is already a tidy tibble with one row per site and order q. The generics::tidy() methods rename the columns to the broom conventions (estimate, std.error, conf.low, conf.high) and add an inverse-variance weight (weight, 1 / se^2, or NA when the standard error is zero, as at a site with a single species), so results stack with other broom output and can go straight into weighted models. generics::glance() gives one row per site with its sample size, observed richness and coverage.

add_site_data() joins site-level covariates, such as elevation, habitat or treatment, from the original data or a separate site table, so diversity can be modelled against them.

Usage

```
## S3 method for class 'td_estimate'
tidy(x, ...)

## S3 method for class 'td_standardise'
tidy(x, ...)

## S3 method for class 'td_rarefy'
tidy(x, ...)

## S3 method for class 'td_boot'
tidy(x, ...)

## S3 method for class 'td_beta_pair'
tidy(x, ...)

## S3 method for class 'td_coverage'
glance(x, ...)

add_site_data(x, data, site = site, cols = NULL)
```

Arguments

x	A tidydiversity result.
...	Not used.
data	A data frame holding the site column and the covariates, such as the community data the result came from, or a site table.
site	<tidy-select> The site column of data, matched to the site column of x.
cols	<tidy-select> Covariates to join. By default every column that is constant within each site and not already in x.

Value

A tibble.

Examples

```
set.seed(1)
est <- div_estimate(png_ants, q = 0, nboot = 20)
generics::tidy(est)

# Join elevation and fit a weighted model
est <- add_site_data(est, png_ants)
fit <- lm(estimate ~ elevation,
  data = generics::tidy(est),
  weights = weight
)
coef(fit)
```

`tidy.td_ordination` *Tidy and glance methods for ordinations*

Description

`tidy()` returns one set of ordination scores as a tibble, and `glance()` a one-row summary of the ordination.

Usage

```
## S3 method for class 'td_ordination'
tidy(
  x,
  display = c("sites", "species", "vectors", "centroids", "axes", "envfit"),
  ...
)

## S3 method for class 'td_ordination'
glance(x, ...)
```

Arguments

`x`	A `td_ordination` from `ord_fit()`.
`display`	Scores to return: `"sites"`, `"species"`, `"vectors"` (environmental arrows), `"centroids"` (factor levels), `"axes"` (eigenvalues and variation explained) or `"envfit"` (fit of passive variables).
`...`	Not used.

Value

A tibble.

Examples

```
ord <- ord_fit(png_frogs, method = "ca")
tidy(ord, "species")
glance(ord)
```

`tidyvisor` *tidyvisor: advice before the analysis, interpretation after it*

Description

`tidyvisor()` is the package's adviser. Give it **data** and a question, and it recommends the analysis, the test, the plots and the code to run (see `ord_advise()`). Give it a **result**, and it explains the tables and the diagram in plain sentences (see `ord_interpret()`).

Its advice follows the standard references of community ecology, and it always gives its reasons, so it can be checked and overruled: the question, which only the analyst knows, comes first.

Usage

```
tidyvisor(x, ...)
```

Arguments

`x`	Community data in long format (to be advised), or the result of `ord_fit()`, `ord_prc()` or `ord_varpart()` (to be interpreted).
`...`	Passed to `ord_advise()` (data) or `ord_interpret()` (results).

Value

A `td_ord_advice` or a `td_ord_interpretation` object.

Examples

```
data("dune", "dune.env", package = "vegan")
d <- community_long(cbind(site = rownames(dune), dune.env, dune),
  cols = dplyr::all_of(names(dune))
)
# Before: which analysis?
tidyvisor(d, abundance = abundance, formula = ~ A1 + Management)
# After: what does it show?
o <- ord_fit(d,
  abundance = abundance, method = "rda", transform = "hellinger",
  formula = ~ A1 + Management
)
tidyvisor(o, permutations = 199)
```

`trait_cwm` *Community-weighted mean traits*

Description

The abundance-weighted mean of each numeric trait in each site, and the abundance-weighted proportion of each level of categorical traits (Garnier et al. 2004; de Bello et al. 2021, ch. 5). Species with a missing trait value are left out of that trait's mean, and the weights of the remaining species are rescaled.

Usage

```
trait_cwm(
  data,
  traits,
  site = site,
  species = species,
  abundance = NULL,
  cols = tidyselect::everything()
)
```

Arguments

`data`	A data frame in long format, with one row per record.
`traits`	A data frame of traits, one row per species, with a column whose name matches the `species` column of `data`.
`site, species`	<tidy-select> The column holding site identifiers and the column holding species names.
`abundance`	<tidy-select> The column holding counts. If `NULL` (the default), a column called `abundance` is used when present, otherwise every row counts as one individual.
`cols`	<tidy-select> Trait columns of `traits`.

Value

A tibble with one row per site and trait: the site column, `trait` and `cwm`. Site variables (constant within sites) are kept.

References

Garnier, E. et al. (2004). Plant functional markers capture ecosystem properties during secondary succession. *Ecology*, 85, 2630-2637. doi:10.1890/030799

Examples

```
aravo <- data_aravo()
trait_cwm(aravo$community, aravo$traits, abundance = cover)
```

`trait_cwm_test` *Test CWM-environment relationships with the max test*

Description

Relating community-weighted means to site variables with an ordinary test inflates Type I error, because CWMs of different sites share the same species traits (Zeleny 2018; Peres-Neto et al. 2017). The *max test* (ter Braak et al. 2012) combines a row-based permutation (shuffle sites) with a column-based permutation (shuffle traits among species) and reports the larger p-value. It controls Type I error whether the null hypothesis concerns the environment or the traits.

Usage

```
trait_cwm_test(
  data,
  traits,
  env_vars,
  site = site,
  species = species,
  abundance = NULL,
  cols = tidyselect::everything(),
  permutations = 999
)
```

Arguments

`data`	A data frame in long format, with one row per record.
`traits`	A data frame of traits, one row per species, with a column whose name matches the `species` column of `data`.
`env_vars`	<tidy-select> Numeric site variables in `data` (constant within sites) to test.
`site, species`	<tidy-select> The column holding site identifiers and the column holding species names.
`abundance`	<tidy-select> The column holding counts. If `NULL` (the default), a column called `abundance` is used when present, otherwise every row counts as one individual.
`cols`	<tidy-select> Trait columns of `traits`.
`permutations`	Number of permutations.

Value

A tibble with one row per numeric trait x variable: `trait`, `variable`, Pearson `r` between CWM and the variable, `p_row`, `p_col` and `p_max`.

References

Zeleny, D. (2018). Which results of the standard test for community-weighted mean approach are too optimistic? *Journal of Vegetation Science*, 29, 953-966. doi:10.1111/jvs.12688

ter Braak, C. J. F., Cormont, A. and Dray, S. (2012). Improved testing of species traits-environment relationships in the fourth-corner problem. *Ecology*, 93, 1525-1526. doi:10.1890/120126.1

Examples

```
aravo <- data_aravo()
set.seed(1)
trait_cwm_test(aravo$community, aravo$traits, abundance = cover,
            cols = c(Height, SLA), env_vars = Snow, permutations = 199)
```

`trait_dist` *Trait dissimilarity between species*

Description

Computes functional dissimilarity between species from traits of any type (numeric, ordinal, categorical, binary, with missing values). `"gower"` uses Gower distance with Podani's extension (`FD::gowdis()`); `"gawdis"` uses `gawdis::gawdis()`, which weights traits so that each contributes equally to the dissimilarity, an improvement on Gower when trait types are mixed (de Bello et al. 2021).

Usage

```
trait_dist(
  traits,
  species = species,
  cols = tidyselect::everything(),
  method = c("gower", "gawdis"),
  groups = NULL,
  ...
)
```

Arguments

`traits`	A data frame of traits, one row per species.
`species`	<`tidy-select`> The column of species names.
`cols`	<`tidy-select`> Trait columns; all other columns by default.
`method`	`"gower"` or `"gawdis"`.
`groups`	For `"gawdis"`, an integer vector giving a group to each trait column, so that traits measuring the same thing (e.g. fuzzy-coded life forms) count as one.
`...`	Passed to `gawdis::gawdis()`.

Value

A `dist` object labelled with species names, usable as `dist` in `div_hill()` and other FD functions.

References

de Bello, F. et al. (2021). Towards a more balanced combination of multiple traits when computing functional differences between species. *Methods in Ecology and Evolution*, 12, 443-448. doi:10.1111/2041210X.13537

Examples

```
data(aravo, package = "ade4")
tr <- data.frame(species = rownames(aravo$traits), aravo$traits)
d <- trait_dist(tr)
as.matrix(d)[1:3, 1:3]
```

`trait_fd_indices` *Classic functional diversity indices*

Description

Functional richness (FRic), evenness (FEve), divergence (FDiv), dispersion (FDis) and Rao's quadratic entropy (RaoQ) for each site (Villeger et al. 2008; Laliberte and Legendre 2010), computed by `FD::dbFD()` and returned as a tidy table. These complement the Hill-number functional diversity of `div_hill()`.

Usage

```
trait_fd_indices(
  data,
  traits,
  site = site,
  species = species,
  abundance = NULL,
  cols = tidyselect::everything(),
  dist = NULL,
  indices = c("FRic", "FEve", "FDiv", "FDis", "RaoQ"),
  m = "max"
)
```

Arguments

`data`	A data frame in long format, with one row per record.
`traits`	A data frame of traits, one row per species, with a column whose name matches the `species` column of `data`.

site, species	<tidy-select> The column holding site identifiers and the column holding species names.
abundance	<tidy-select> The column holding counts. If NULL (the default), a column called abundance is used when present, otherwise every row counts as one individual.
cols	<tidy-select> Trait columns of traits.
dist	Optional species dissimilarity (e.g. from trait_dist()); computed with Gower distance from traits if NULL.
indices	Indices to return.
m	Number of PCoA axes for FRic and FDiv: "max" or "min" (see FD::dbFD()), or a number.

Value

A tibble with the site column, nbsp (species richness), index and value.

References

Villeger, S., Mason, N. W. H. and Mouillot, D. (2008). New multidimensional functional diversity indices for a multifaceted framework in functional ecology. *Ecology*, 89, 2290-2301. doi:10.1890/071206.1

Laliberte, E. and Legendre, P. (2010). A distance-based framework for measuring functional diversity from multiple traits. *Ecology*, 91, 299-305. doi:10.1890/082244.1

Examples

```
aravo <- data_aravo()
trait_fd_indices(aravo$community, aravo$traits, abundance = cover,
              indices = c("FDis", "RaoQ"))
```

---

trait_fourth_corner *Fourth-corner analysis*

Description

Tests every trait x environment association with the fourth-corner method (Legendre et al. 1997; Dray and Legendre 2008) and the max test (model 6, ter Braak et al. 2012), using ade4::fourthcorner(), and returns a tidy table.

Usage

```
trait_fourth_corner(
  data,
  traits,
  env_vars,
  site = site,
  species = species,
  abundance = NULL,
  cols = tidyselect::everything(),
  permutations = 999,
  adjust = "fdr"
)
```

Arguments

data	A data frame in long format, with one row per record.
traits	A data frame of traits, one row per species, with a column whose name matches the species column of data.
env_vars	<tidy-select> Numeric site variables in data (constant within sites) to test.
site, species	<tidy-select> The column holding site identifiers and the column holding species names.
abundance	<tidy-select> The column holding counts. If NULL (the default), a column called abundance is used when present, otherwise every row counts as one individual.
cols	<tidy-select> Trait columns of traits.
permutations	Number of permutations.
adjust	Method for adjusting p-values across all tests, passed to stats::p.adjust(); "none" for no adjustment.

Value

A tibble with trait, variable, the test statistic (Pearson r for two numeric variables, otherwise as in ade4), p.value and p.adj.

References

Dray, S. and Legendre, P. (2008). Testing the species traits-environment relationships: the fourth-corner problem revisited. *Ecology*, 89, 3400-3412. doi:10.1890/080349.1

Examples

```
aravo <- data_aravo()
set.seed(1)
trait_fourth_corner(aravo$community, aravo$traits, abundance = cover,
                cols = c(Height, SLA), env_vars = c(Snow, PhysD),
                permutations = 199)
```

`trait_itv` *Intra- and interspecific trait variability*

Description

Decomposes variation in community trait means into the part caused by species turnover (interspecific) and the part caused by trait change within species (intraspecific), following Leps et al. (2011). It compares the *specific* average (CWM from site-specific species means) with the *fixed* average (CWM from species means over all sites).

Usage

```
trait_itv(
  data,
  traits,
  site = site,
  species = species,
  abundance = NULL,
  cols,
  formula = NULL
)
```

Arguments

`data`	A data frame in long format, with one row per record.
`traits`	A data frame of trait values measured per site: columns for site, species and the trait(s). Several rows per site and species (individuals) are averaged.
`site, species`	<tidy-select> The column holding site identifiers and the column holding species names.
`abundance`	<tidy-select> The column holding counts. If `NULL` (the default), a column called `abundance` is used when present, otherwise every row counts as one individual.
`cols`	<tidy-select> Trait columns of `traits`.
`formula`	Optional one-sided formula of site variables. If given, the sums of squares are those explained by the formula; otherwise the total sums of squares are decomposed.

Value

A tibble with one row per trait and component (`"turnover"`, `"intraspecific"`, `"covariation"`, `"total"`), the sum of squares `ss` and its share `prop` of the total.

References

Leps, J. et al. (2011). Community trait response to environment: disentangling species turnover vs intraspecific trait variability effects. *Ecography*, 34, 856-863. doi:10.1111/j.16000587.2010.06904.x

`trait_rao` *Rao's quadratic entropy: alpha, beta and gamma functional diversity*

Description

Partitions functional diversity across sites with Rao's quadratic entropy, using the equivalent-numbers correction of de Bello et al. (2010) so that beta = gamma / alpha and is not underestimated. Sites are weighted equally. Dissimilarities are scaled to a maximum of 1.

Usage

```
trait_rao(
  data,
  traits = NULL,
  site = site,
  species = species,
  abundance = NULL,
  cols = tidyselect::everything(),
  dist = NULL
)
```

Arguments

`data`	A data frame in long format, with one row per record.
`traits`	A data frame of traits, one row per species, with a column whose name matches the `species` column of `data`.
`site, species`	<tidy-select> The column holding site identifiers and the column holding species names.
`abundance`	<tidy-select> The column holding counts. If `NULL` (the default), a column called `abundance` is used when present, otherwise every row counts as one individual.
`cols`	<tidy-select> Trait columns of `traits`.
`dist`	Species dissimilarity; computed with Gower distance from `traits` if `NULL`. Use a dissimilarity of 1 between all species for taxonomic (Simpson) diversity.

Value

A tibble with `component` (`"alpha"`, `"beta"`, `"gamma"`), `rao` (Rao's Q, for alpha and gamma) and `equivalent` (equivalent numbers, 1 / (1 - Q)), plus `beta_prop`, the proportional turnover `1 - alpha / gamma` (in equivalent numbers).

References

de Bello, F. et al. (2010). The partitioning of diversity: showing Theseus a way out of the labyrinth. *Journal of Vegetation Science*, 21, 992-1000. doi:10.1111/j.16541103.2010.01195.x

Examples

```
aravo <- data_aravo()
trait_rao(aravo$community, aravo$traits, abundance = cover)
```

`trait_ses` *Standardised effect sizes from null models*

Description

Compares an observed functional diversity index with its distribution under a null model, to ask whether communities are more (divergence, e.g. limiting similarity) or less (convergence, e.g. environmental filtering) functionally diverse than expected by chance (de Bello et al. 2021, ch. 7). SES = (observed - mean null) / sd null.

Usage

```
trait_ses(
  data,
  traits,
  site = site,
  species = species,
  abundance = NULL,
  cols = tidyselect::everything(),
  dist = NULL,
  index = c("RaoQ", "FDis", "MPD"),
  null = c("traits", "richness", "swap"),
  times = 999
)
```

Arguments

`data`	A data frame in long format, with one row per record.
`traits`	A data frame of traits, one row per species, with a column whose name matches the `species` column of `data`.
`site, species`	<tidy-select> The column holding site identifiers and the column holding species names.
`abundance`	<tidy-select> The column holding counts. If `NULL` (the default), a column called `abundance` is used when present, otherwise every row counts as one individual.
`cols`	<tidy-select> Trait columns of `traits`.
`dist`	Optional species dissimilarity (e.g. from `trait_dist()`); computed with Gower distance from `traits` if `NULL`.
`index`	Index to test: `"RaoQ"`, `"FDis"` or `"MPD"` (abundance- weighted mean pairwise dissimilarity).
`null`	Null model, see Details.
`times`	Number of randomisations.

Details

Null models:

- `"traits"`: shuffle trait values among species (keeps every site's species richness and abundances; tests trait structure).
- `"richness"`: shuffle abundances among species within each site (keeps richness and the abundance distribution of each site).
- `"swap"`: independent swap of the site x species matrix, keeping site and species totals (`vegan::nullmodel()`, `"quasiswap_count"`).

Value

A tibble with one row per site: `observed`, `null_mean`, `null_sd`, `ses` and `p.value` (two-sided rank-based).

Examples

```
aravo <- data_aravo()
set.seed(1)
trait_ses(aravo$community, aravo$traits, abundance = cover, times = 49)
```

`trait_tpd` *Functional diversity from trait probability densities*

Description

The trait probability density (TPD) approach (Carmona et al. 2016, 2019) describes each species by the probability density of its trait values, estimated from individual measurements, so intraspecific variability is part of every index. Community TPDs are the abundance-weighted sums of species TPDs. From them, `trait_tpd()` returns functional richness, evenness and divergence (`TPD::REND()`) and functional redundancy (`TPD::redundancy()`) per site, and `trait_tpd_dissim()` returns pairwise functional dissimilarity between sites, split into shared and non-shared parts (`TPD::dissim()`).

Usage

```
trait_tpd(
  data,
  traits,
  site = site,
  species = species,
  abundance = NULL,
  cols,
  alpha = 0.95,
  n_divisions = NULL
)
```

```
trait_tpd_dissim(
  data,
  traits,
  site = site,
  species = species,
  abundance = NULL,
  cols,
  alpha = 0.95,
  n_divisions = NULL
)
```

Arguments

data	A data frame in long format, with one row per record.
traits	A data frame with one row per measured individual: a species column (named as in data) and one to four numeric trait columns. Traits are often log-transformed first.
site, species	<tidy-select> The column holding site identifiers and the column holding species names.
abundance	<tidy-select> The column holding counts. If NULL (the default), a column called abundance is used when present, otherwise every row counts as one individual.
cols	<tidy-select> The trait columns (one to four).
alpha	Proportion of the probability density kept for each species (removes outliers).
n_divisions	Number of grid divisions per trait axis; defaults to those of TPD::TPDs().

Value

trait_tpd(): a tibble with the site column, FRic, FEve, FDiv and redundancy. trait_tpd_dissim(): a tibble of site pairs with dissimilarity, its parts due to shared (p_shared) and non-shared (p_nonshared) trait space.

References

Carmona, C. P. et al. (2016). Traits without borders: integrating functional diversity across scales. *Trends in Ecology & Evolution*, 31, 382-394. doi:10.1016/j.tree.2016.02.003

Carmona, C. P. et al. (2019). Trait probability density (TPD): measuring functional diversity across scales based on TPD with R. *Ecology*, 100, e02876. doi:10.1002/ecy.2876

wanang_butterflies *Butterflies in the Wanang forest dynamics plot, Papua New Guinea*

Description

Real field data: butterflies counted on 36 sampling days, three days a month over 12 months spread across three years, in the Wanang 50 ha forest dynamics plot, Papua New Guinea (Luk et al. 2019). Each sampling day is one site, which makes the data suited to sample-based accumulation and to partitioning diversity across days, months and years. Only the Wanang part of the published multi-site dataset is included.

Usage

```
wanang_butterflies
```

Format

A tibble with one row per day and species recorded, and six columns:

site Sampling day, `"day01"` to `"day36"`.

day Sampling day number, 1 to 36.

month Sampling month, 1 to 12, in order of sampling.

year Sampling year, 1 to 3, in order of sampling.

species Butterfly species.

abundance Number of individuals counted.

Source

Luk, C.-L., Basset, Y., Kongnoo, P., Hau, B. C. H. and Bonebrake, T. C. (2019). Data from: Inter-annual monitoring improves diversity estimation of tropical butterfly assemblages. Dryad. doi:10.5061/dryad.j35fq7r, released under CC0 1.0. Built by `data-raw/wanang_butterflies.R`.

References

Luk, C.-L., Basset, Y., Kongnoo, P., Hau, B. C. H. and Bonebrake, T. C. (2019). Inter-annual monitoring improves diversity estimation of tropical butterfly assemblages. *Biotropica*, 51, 519-528. doi:10.1111/btp.12671

Examples

```
wanang_butterflies
div_hill(wanang_butterflies, q = 0, .by = year) |> head()
```

wanang_caterpillars *Caterpillars reared from trees in primary and secondary forest at Wanang*

Description

Real field data: Lepidoptera larvae collected from the foliage of trees in a primary and a secondary forest plot in the Wanang Conservation Area, Madang Province, Papua New Guinea, between 2003 and 2009, and reared to adults where possible by the New Guinea Binatang Research Centre (Butterill and Redmond 2023). Each plot has five sampling units (`WP1`-`WP5` in primary and `WS1`-`WS5` in secondary forest). Counts are the number of individuals of each morphospecies reared from each host plant species. Taxa are morphospecies codes identified to different ranks (species, genus, family or order), so treat them as morphospecies, not as named species. The host plant makes the data suited to species-overlap diagrams (`plot_venn()`) and host-use comparisons.

Usage

```
wanang_caterpillars
```

Format

A tibble with one row per sampling unit, host plant and morphospecies, and eight columns:

site Forest type, `"primary"` or `"secondary"`.

plot Sampling unit, `"WP1"`-`"WP5"` and `"WS1"`-`"WS5"`.

host Host plant species (`NA` when not recorded).

species Morphospecies code (GBIF `taxonID`).

scientific_name Most precise name given for the morphospecies.

rank Rank of `scientific_name`.

family Lepidoptera family (`NA` when only identified to order).

abundance Number of individuals.

Source

Butterill, P. and Redmond, C. (2023). Plot-based sampling and rearing of adult lepidoptera from lowland rainforest in Wanang, Papua New Guinea. Version 1.1. New Guinea Binatang Research Centre. Occurrence dataset, GBIF. doi:10.15468/7e96ht, released under CC BY 4.0 (`https://creativecommons.org/licenses/by/4.0/`): cite this source when you use the data. Built from all 9,354 GBIF occurrence records by `data-raw/wanang_caterpillars.R`.

Examples

```
wanang_caterpillars
div_hill(wanang_caterpillars, site = plot, q = 0:2) |> head()
```

Index

∗ **datasets**
    png_ants, 83
    png_frogs, 84
    png_tree_ants, 85
    portal_rodents, 86
    wanang_butterflies, 115
    wanang_caterpillars, 116

add_site_data *(*tidy-tidydiversity*)*, 100
ade4::fourthcorner(), *108*
ape::phylo, *19*, *42*, *46*, *51*, *56*
ape::read.tree(), *79*
as_community, 4
autoplot(), *62*, *99*
autoplot-tidydiversity, 5
autoplot.td_beta_decay
        *(*autoplot-tidydiversity*)*, 5
autoplot.td_beta_hill
        *(*autoplot-tidydiversity*)*, 5
autoplot.td_beta_hill_pair
        *(*autoplot-tidydiversity*)*, 5
autoplot.td_beta_multi
        *(*autoplot-tidydiversity*)*, 5
autoplot.td_beta_pair
        *(*autoplot-tidydiversity*)*, 5
autoplot.td_completeness
        *(*autoplot-tidydiversity*)*, 5
autoplot.td_coverage
        *(*autoplot-tidydiversity*)*, 5
autoplot.td_dispersion
        *(*autoplot-tidydiversity*)*, 5
autoplot.td_estimate
        *(*autoplot-tidydiversity*)*, 5
autoplot.td_evenness
        *(*autoplot-tidydiversity*)*, 5
autoplot.td_hill
        *(*autoplot-tidydiversity*)*, 5
autoplot.td_mde
        *(*autoplot-tidydiversity*)*, 5
autoplot.td_ordination, 10
autoplot.td_prc *(*ord_prc*)*, 66
autoplot.td_rarefy
        *(*autoplot-tidydiversity*)*, 5
autoplot.td_standardise
        *(*autoplot-tidydiversity*)*, 5
autoplot.td_varpart *(*ord_varpart*)*, 72

beta_chao *(*beta_shared*)*, 29
beta_decay, 14
beta_decay(), *9*, *15–17*
beta_decay_boot, 15
beta_decay_model, 16
beta_dispersion *(*beta_permanova*)*, 27
beta_dist, 18
beta_dist(), *22*, *25*
beta_hill, 18
beta_hill(), *9*, *21*, *22*
beta_hill_pair, 21
beta_hill_pair(), *18*
beta_indicator, 23
beta_mantel, 24
beta_multi *(*beta_pair*)*, 26
beta_multi(), *9*, *57*
beta_pair, 26
beta_pair(), *9*, *14*, *15*, *18*, *57*
beta_permanova, 27
beta_permanova(), *33*
beta_shared, 29
beta_turnover_counts, 31
beta_welch, 33
betapart::beta.multi.abund(), *26*
betapart::beta.pair.abund(), *26*
betapart::decay.model(), *16*, *17*

community_long *(*as_community*)*, 4
community_wide *(*as_community*)*, 4

data_aravo, 34
div_boot, 35
div_boot(), *58*

div_completeness, 36
div_coverage, 38
div_coverage(), *9*, *57*
div_effort, 39
div_estimate, 40
div_estimate(), *9*, *35, 36*, *41*, *53*, *55*, *57, 58*
div_evenness, 43
div_fisher_alpha, 44
div_hill, 45
div_hill(), *9*, *57*, *96*, *107*
div_homogenisation, 47
div_invasion, 49
div_rarefy, 50
div_rarefy(), *9*, *39*
div_richness, 52
div_standardise, 54
div_standardise(), *9*, *35, 36*, *53*, *57*
div_table, 57

FD::dbFD(), *107, 108*
FD::gowdis(), *106*
flextable::flextable(), *57*

gawdis::gawdis(), *106*
generics::glance(), *59*, *100*
generics::tidy(), *59*, *100*
ggplot2::autoplot(), *15*, *22*, *52*
ggplot2::discrete_scale(), *100*
ggplot2::facet_wrap(), *9*
ggplot2::ggsave(), *9*, *80*
ggplot2::theme(), *100*
glance(), *64*, *89*
glance.td_count_model *(*model_count*)*, 59
glance.td_coverage
*(*tidy-tidydiversity*)*, 100
glance.td_mde *(*range_mde*)*, 88
glance.td_ordination
*(*tidy.td_ordination*)*, 102
graphics::persp(), *77*
gt::gt(), *57*

hardhat::importance_weights(), *58*

indicspecies::multipatt(), *23, 24*
iNEXT::estimateD(), *56*
iNEXT::iNEXT(), *41*

knitr::kable(), *57*

MASS::glm.nb(), *59*
model_case_weights, 58
model_count, 59
model_count(), *59*, *76*
model_group_folds, 60

ord_advise, 61
ord_advise(), *103*
ord_fit, 62
ord_fit(), *11*, *29*, *65*, *67*, *71*, *102, 103*
ord_interpret, 65
ord_interpret(), *103*
ord_prc, 66
ord_prc(), *65*, *103*
ord_select, 68
ord_style, 69
ord_style(), *12*, *67*, *73*
ord_test, 71
ord_test(), *64*
ord_varpart, 72
ord_varpart(), *65*, *103*

patchwork::wrap_plots(), *78*
phylo_combine_dist, 74
phylo_combine_dist(), *74*
phylo_dist, 74
phylo_signal, 75
phytools::phylosig(), *75*
plot-partial, 76
plot_palette *(*theme_tidydiversity*)*, 99
plot_panel, 77
plot_partial *(*plot-partial*)*, 76
plot_partial(), *76*
plot_phylo, 79
plot_save, 80
plot_save(), *78*
plot_surface *(*plot-partial*)*, 76
plot_surface(), *76*
plot_trait_space, 81
plot_venn, 82
plot_venn(), *116*
png_ants, 83
png_frogs, 84
png_tree_ants, 85
portal_rodents, 86

range_edges, 87
range_mde, 88
range_mde(), *90*, *95*
range_partition, 90

range_shift_expected
(range_shift_validate), 93
range_shift_resurvey, 91
range_shift_resurvey(), 93, 94
range_shift_test
(range_shift_resurvey), 91
range_shift_validate, 93
range_through, 95
range_through(), 88, 89
rsample::group_vfold_cv(), 60

scale_colour_tidydiversity
(theme_tidydiversity), 99
scale_fill_tidydiversity
(theme_tidydiversity), 99
stats::dist, 18, 20, 42, 46, 51, 56
stats::glm(), 59
stats::hclust(), 18
stats::oneway.test(), 33
stats::p.adjust(), 23, 109
stems, 96
stems_abundance (stems), 96
stems_abundance(), 96
stems_as_community (stems), 96
stems_as_community(), 96
stems_as_trees (stems), 96
stems_as_trees(), 96
stems_basal_area (stems), 96
stems_basal_area(), 96
stems_demography (stems), 96
stems_demography(), 96
stems_size_class (stems), 96
stems_size_class(), 96
step_hill, 98

theme_tidydiversity, 99
theme_tidydiversity(), 8, 9, 12, 79, 81, 82
tidy(), 64
tidy-tidydiversity, 100
tidy.td_beta_pair (tidy-tidydiversity),
100
tidy.td_boot (tidy-tidydiversity), 100
tidy.td_count_model (model_count), 59
tidy.td_estimate (tidy-tidydiversity),
100
tidy.td_ordination, 102
tidy.td_rarefy (tidy-tidydiversity), 100
tidy.td_standardise
(tidy-tidydiversity), 100
tidyvisor, 103
TPD::dissim(), 113
TPD::redundancy(), 113
TPD::REND(), 113
TPD::TPDs(), 114
trait_cwm, 104
trait_cwm_test, 105
trait_dist, 106
trait_dist(), 81, 108, 112
trait_fd_indices, 107
trait_fd_indices(), 74
trait_fourth_corner, 108
trait_itv, 110
trait_rao, 111
trait_rao(), 74
trait_ses, 112
trait_ses(), 74
trait_tpd, 113
trait_tpd_dissim (trait_tpd), 113

vegan::adonis2(), 18, 27–29, 34
vegan::anova.cca(), 71, 72
vegan::betadisper(), 18, 27, 29, 34
vegan::envfit(), 64
vegan::mantel(), 24
vegan::mantel.partial(), 24
vegan::metaMDS(), 18
vegan::nullmodel(), 113
vegan::ordiellipse(), 13
vegan::ordiR2step(), 68
vegan::varpart(), 72
vegan::vegdist(), 28, 29, 34, 63, 69

wanang_butterflies, 115
wanang_caterpillars, 116