Package 'nodiv'

July 22, 2025

Type Package

Title Compares the Distribution of Sister Clades Through a Phylogeny

Version 1.4.2

Date 2023-08-21

Author Michael Krabbe Borregaard

Maintainer Michael Krabbe Borregaard <`mkborregaard@sund.ku.dk`>

Description An implementation of the nodiv algorithm, see Borregaard, M.K., Rahbek, C., Fjeldsaa, J., Parra, J.L., Whittaker, R.J. & Graham, C.H. 2014. Node-based analysis of species distributions. Methods in Ecology and Evolution 5(11): 1225-1235. <DOI:10.1111/2041-210X.12283>. Package for phylogenetic analysis of species distributions. The main function goes through each node in the phylogeny, compares the distributions of the two descendant nodes, and compares the result to a null model. This highlights nodes where major distributional divergence have occurred. The distributional divergence for these nodes is mapped.

Depends R (>= 3.0)

Imports picante, raster, ape, sp, vegan, utils

Suggests RColorBrewer, parallel, testthat, colorspace

License MIT + file LICENSE

URL `https://github.com/mkborregaard/nodiv`

NeedsCompilation no

Repository CRAN

Date/Publication 2023-08-21 08:12:36 UTC

Contents

nodiv-package 2
add_shape 3
basal_node 4
coquettes 5
distrib_data 5
gridData 7
Nodesig 8

Node_analysis 9
Node_size 11
Nspecies 12
occurrences 13
phyplot 14
plot_nodes_phylo 17
plot_points 18
plot_richness 20
sitestat 21
SOS 22
species_stat 23
subsample 24
two_color_map 25
update_object 26

Index **27**

`nodiv-package` *nodiv - Node-based analysis of species distributions*

Description

The package implements Borregaard et al. (2014) method for identifying nodes in a phylogeny associated with divergent distributions. The main algorithm goes through each node in the phylogeny and relates node overlap to a null model. The package also provides functions for preparing the data sets, for exploratory plots and further analysis.

Details

The workflow starts by creating a `nodiv_data` data object, which makes sure that the phylogeny, the species distributions and the spatial coordinates of sites are matched correctly. The function takes a number of data types. There are several functions for exploratory data analysis, including `plot`, `summary`, `richness`, `Node_occupancy`, `Node_size` etc. The core of the package are the functions `Nodesig`, which compares the distributions of the two clades descending from a node, and `Node_analysis`, which applies this function to all nodes in the phylogeny and summarizes the results as a `nodiv_result` object. There is a set of functions for interpretation of the results, including `plot`, `plotSOS`, and `summary`. The package also provides basic functions for plotting and manipulating data sets that combine spatial distributions with phylogenies, e.g. `subsample`, `plot_grid` and `plot_points`. Note that the package may still be unstable - if you experience problems, try cloning the github repository instead: `library(devtools); install_github("mkborregaard/nodiv")`, or feel free to contact the maintainer.

Author(s)

Michael Krabbe Borregaard

Maintainer: Michael Krabbe Borregaard <mkborregaard@sund.ku.dk>

References

Borregaard, M.K., Rahbek, C., Fjeldsaa, J., Parra, J.L., Whittaker, R.J. & Graham, C.H. 2014. Node-based analysis of species distributions. Methods in Ecology and Evolution 5(11): 1225-1235. <DOI:10.1111/2041-210X.12283>

Examples

```
data(coquettes)
res <- Node_analysis(coquettes, 50, "rdtable")
opar <- par(mfrow = c(1,2))
plot(res)
plotSOS(res, 28, shapefill = terrain.colors(64))
summary(res)
par(opar)
```

`add_shape` *Add a shapefile to a* `nodiv` *data object*

Description

Takes a shapefile, e.g. as points or polygons, and adds it to the data object

Usage

```
add_shape(distrib_data, shape)
```

Arguments

`distrib_data`	An object of type `distrib_data`, `nodiv_data` or `nodiv_result`
`shape`	A GIS shapefile, from e.g. packages `maptools` or `rgdal`

Value

Returns the object with the new shapefile

Author(s)

Michael Krabbe Borregaard

See Also

`distrib_data`

Examples

```
# remove the shapefile from coquettes
data(coquettes)
sh <- coquettes$shape
coquettes$shape <- NULL
# and add it back in
coquettes <- add_shape(coquettes, sh)
```

`basal_node` *Identify nodes in a phylogeny*

Description

Returns the internal (ape) node numbers corresponding to certain nodes in the phylogeny

Usage

```
basal_node(tree)
nodenumbers(tree)
nodes(tree, all = FALSE)
Descendants(node, tree)
Parent(node, tree)
Sister(node, tree)
MostRecentAncestor(tips, tree)
```

Arguments

`tree`	An object of type `phylo`, or alternatively an object of type `nodiv_data`
`node`	The internal node number of a focal node in the phylogeny
`all`	Give all node labels in order of occurrence. If FALSE, only non-empty node labels will be shown in alphabetical order
`tips`	The species to find the most recent common ancestor for

Details

`Parent` gives the code immediately basal to the focal node, `Descendants` gives the nodes descending from the focal node, and `Sister` gives the sister node(s). `MostRecentAncestor` gives the most recent common node for the given tips.

Value

The internal (ape) node number of the node

Author(s)

Michael Krabbe Borregaard

`coquettes` *Distribution of coquette hummingbirds in Northern South America*

Description

Phylogeny and distributions for the coquette clade of hummingbirds in Ecuador, Peru and Colombia

Usage

```
data(coquettes)
```

Format

a `nodiv_data` object with the phylogeny and distributions of all species

Source

Graham, C.H., Parra, J.L., Tinoco, B.A., Stiles, F.G. & McGuire, J.A. (2012). Untangling the influence of ecological and evolutionary factors on trait variation across hummingbird assemblages. Ecology, 93, S99-S111.

Examples

```
data(coquettes)
plot(coquettes, shapefill = terrain.colors(64), pch = 21)
```

`distrib_data` *Data objects for node-based analysis*

Description

Functions to collect and display data with matching species distributions and phylogenies

Usage

```
distrib_data(commatrix, coords = NULL, proj4string_in = CRS(as.character(NA)),
  type = c("auto", "grid", "points"), shape = NULL)
nodiv_data(phylo, commatrix, coords, proj4string_in = CRS(as.character(NA)),
  type = c("auto", "grid", "points"), shape = NULL)
## S3 method for class 'distrib_data'
summary(object, ...)
## S3 method for class 'nodiv_data'
summary(object, ...)
## S3 method for class 'distrib_data'
plot(x, ...)
## S3 method for class 'nodiv_data'
```

```
plot(x, ...)
## S3 method for class 'distrib_data'
identify(x, ...)
```

Arguments

`phylo`	A phylogeny in ape's phylo format, with tip.labels that correspond to the species names in commatrix
`commatrix`	A community matrix or data.frame of species distributions, with species as columns, sites as rows, and 0/1 denoting the absence or presence of a species in a given site. Alternatively, the functions accept a `data.frame` in the phylocom format, i.e. with three columns containing plot id, abundance (ignored) and species id. The `nodiv_data` function also accepts an object of type `distrib_data`, in which case `coords` is ignored. The `nodiv_data` function matches the species in the matrix to the tips of `phylo`.
`coords`	The coordinates of all sites in the data. The input can be of a variety of different formats. Can be a `data.frame` or `matrix` with two columns. If more columns are present, the function will search for typical coordinate names such as X/Y or Lon/Lat. Alternatively, an object inheriting from `SpatialPoints` from package `sp` is accepted. The coordinates are transformed to `SpatialPoints` using the proj4string argument. Site names can be given as a third column of the data.frame, as rownames for the matrix or omitted. If site names are given, and `commatrix` has row.names, these are matched.
`proj4string_in`	Projection string as defined in package `sp`
`type`	A descriptor of whether the sites are point (community) sites, or the centres of regular grid cells. If omitted or set to `auto`, the function will guess the type from `coords`.
`shape`	An optional shapefile to overlay plots upon.
`object`	An object of type distrib_data or nodiv_data
`x`	An object of type distrib_data or nodiv_data
`...`	Further arguments to be passed to the plotting function

Details

When making an object of type 'grid', the coordinates will be stored as `SpatialPixels` from package `sp`. This may result in a warning if there are rows or columns in the grid without data. The function removes these. This may affect plotting, but can otherwise be ignored.

Value

The `distrib_data` function produces an object of type `distrib_data`, which has the following components:

`coords`	A `SpatialPointsDataFrame` or `SpatialPixelsDataFrame` (from `sp`) with spatial coordinates, and one column named `sites` containing the site names
`comm`	The community matrix
`type`	A character vector describing the type of data

`species`	A list of species names
`shape`	An optional slot detailing a shapefile object for plotting the data

The `nodiv_data` function produces an object of type `nodiv_data`, which contains the additional component:

`phylo`	A phylogeny of type `phylo` from package `ape`

Author(s)

Michael Krabbe Borregaard

See Also

`Node_analysis`

`gridData` *Apply a grid (2D bins) to nodiv objects of type 'points'*

Description

Takes an object of type `distrib_data` or `nodiv_data`, and puts the point distributions on a grid. Can also be used for resampling grid data at a coarser scale. If the comm matrix has abundances (any one value larger than 1), they are summed across points in the grid cell.

Usage

```
gridData(dist_data, cellsize_x = 1, cellsize_y = cellsize_x, xll_corner, yll_corner)
```

Arguments

`dist_data`	An object of type `distrib_data` or `nodiv_data`
`cellsize_x`	The size of the bins along the X (longitudinal) axis, in units of the spatial coordinates of the original points
`cellsize_y`	The size of the bins along the Y (latitudinal) axis, in units of the spatial coordinates of the original points
`xll_corner`	The x coordinate of the lower left corner of the lower left grid cell of the intended grid. The corner must be left of all points in the data set
`yll_corner`	The y coordinate of the lower left corner of the lower left grid cell of the intended grid. The corner must be below all points in the data set

Value

The return value is the original `dist_data` object with gridded distribution data

Author(s)

Michael Krabbe Borregaard

See Also

`distrib_data`, ~~~

Examples

```
## Not run:
data(coquettes)
new <- gridData(coquettes, 2)
plot_richness(new)
## End(Not run)
```

`Nodesig` *Compare the distributions of a pair of sister clades*

Description

Compares the overlap of species richness of a pair of sister clades to the expectation from a null model. By default, the function compares the two descendants from the basal node of the phylogeny

Usage

```
Nodesig(nodiv_data, Node_sp = NULL, repeats = 100, method = c("rdtable", "quasiswap"),
  show = T)
```

Arguments

`nodiv_data`	An object of type `nodiv_data`
`Node_sp`	A character or numeric vector indicating which species that belong to one of the clades. If NULL, the species of a descendant from the basal node of nodiv_data is chosen randomly
`repeats`	The number of repeats to use for the null model
`method`	The null model algorithm used in the calculations
`show`	Should a progress bar be printed during calculations?

Details

The \"rdtable\" algorithm is efficient, but may underestimate the deviation between nodes. It should primarily be used for exploratory data analysis. Note that the more robust \"quasiswap\" algorithm may take hours to run on data sets with a few hundred species.

Value

The return value is a list with 5 elements:

`SR`	A numeric vector of the standardized difference between the empirical and simulated species richness for one node in each site
`rval`	The rank of the empirical species richness of one node in the distribution of simulated values
`nodeemp`	The numerical species richness in each site
`nodemeans`	The mean simulated species richness of one node in each site
`nodesds`	The standard deviation of the simulated species richness of one node in each site

Author(s)

Michael Krabbe Borregaard

References

Borregaard, M.K., Rahbek, C., Fjeldsaa, J., Parra, J.L., Whittaker, R.J. & Graham, C.H. 2014. Node-based analysis of species distributions. Methods in Ecology and Evolution 5(11): 1225-1235. DOI: 10.1111/2041-210X.12283

See Also

`Node_analysis`

Examples

```
data(coquettes)
res <- Nodesig(coquettes, repeats = 50)
hist(res$rval, xlim = 0:1)
```

`Node_analysis` *Calculate GND and SOS scores for a phylogeny and community data set*

Description

Goes through each node in the phylogeny, and compares the distribution overlap of sister clades to that expected from a null mode that considers node size, species richness and range size.

Usage

```
Node_analysis(nodiv_data, repeats = 100, method = c("rdtable", "quasiswap"),
                cores = 1, log_parallel_progress = FALSE)
## S3 method for class 'nodiv_result'
summary(object, ...)
## S3 method for class 'nodiv_result'
print(x, printlen = 4, ...)
## S3 method for class 'nodiv_result'
plot(x, label = nodenumbers(x), zlim = 0:1, ...)
```

Arguments

`nodiv_data`	An object created by the `nodiv_data` function
`repeats`	Number of null communities to create at each node
`method`	The null model algorithm used to simulate random communities
`cores`	The number of cores to use in the computation. Note that multi-core processing is yet unimplemented.
`log_parallel_progress`	If this is TRUE, the results from each node will be saved as an '.rda' file in the working directory. Useful for large parallel computations.
`object`	The `nodiv_result` object
`x`	The `nodiv_result` object
`printlen`	The number of species to print to screen
`label`	The text to use for labelling nodes in the plot
`zlim`	The values of GND to use for the extremes of the color scale on the plot
`...`	Further arguments to be passed to the function

Details

The "rdtable" algorithm is efficient, but may underestimate the deviation between nodes. It should primarily be used for exploratory data analysis. Note that the more robust "quasiswap" algorithm may take days to run on larger data sets.

The `plot` function for `nodiv_result` plots the phylogenetic tree with colored symbols at the nodes. The color and size the nodes are proportional with the GND value of the node.

Value

The result is an object of type `nodiv_result`, which has the following components:

`GND`	A numeric vector between 0 and 1 describing the distributional divergence associated with a node
`SOS`	A numeric matrix given the SOS of values, which expresses the difference between empirical and expected species richness of each clade at each site

It also includes all the elements of the `nodiv_data` object:

`coords`	A `SpatialPointsDataFrame` or `SpatialPixelsDataFrame` (from `sp`) with spatial coordinates, and one column named `sites` containing the site names
`comm`	The community matrix
`type`	A character vector describing the type of data
`species`	A list of species names
`shape`	An optional slot detailing a shapefile object for plotting the data
`phylo`	A phylogeny of type `phylo` from package `ape`

Author(s)

Michael Krabbe Borregaard

References

Borregaard, M.K., Rahbek, C., Fjeldsaa, J., Parra, J.L., Whittaker, R.J. & Graham, C.H. 2014. Node-based analysis of species distributions. Methods in Ecology and Evolution 5(11): 1225-1235. DOI: 10.1111/2041-210X.12283

See Also

`plotSOS`

Examples

```
data(coquettes)
res <- Node_analysis(coquettes, 50, "rdtable")
opar <- par(mfrow = c(1,2))
plot(res)
plotSOS(res, 28, shapefill = terrain.colors(64))
summary(res)
par(opar)
```

`Node_size` *Attributes of individual clades*

Description

Functions that summarize the species that descend from a node, and their distribution.

Usage

```
Node_size(nodiv_data, node = NULL)
Node_sites(nodiv_data, node, names = TRUE)
Node_species(nodiv_data, node, names = TRUE)
Node_occupancy(nodiv_data, node = NULL)
```

Arguments

`node`	A character or numeric variable specifying one or more nodes in the phylogeny
`nodiv_data`	An object of type `nodiv_data` or `nodiv_result`. `Node_species` also accepts objects of type `phylo`
`names`	If `TRUE`, return a character vector of species names; otherwise return the species index numbers in the community matrix

Details

If no `node` is specified in `Node_size` or `Node_occupancy`, the function will calculate the value for all nodes in the phylogeny

Value

For `Node_size` and `Node_occupancy` the number of species or number of occupied sites of the nodes; or a vector giving this value for all nodes. For `Node_species` or `Node_sites` a character vector of names of species descending from a node, or the sites where they occur.

Author(s)

Michael Krabbe Borregaard

`Nspecies` *Basic attributes of* `distrib_data` *objects*

Description

Calculates the species richness and site occupancy of a `distrib_data` object

Usage

```
Nspecies(distrib_data)
Nsites(distrib_data)
richness(distrib_data, sites = NULL)
occupancy(distrib_data, species = NULL)
sites(distrib_data)
species(distrib_data)
coords(distrib_data)
```

Arguments

`distrib_data`	An object of type `distrib_data`, `nodiv_data` or `nodiv_result`
`sites`	The site for which to return the richness
`species`	The species for which to return the occupancy

Value

`Nspecies` gives the number of species in the dataset, and `Nsites` the number of sites. `richness` gives a numeric vector with the species richness of all sites. `occupancy` gives the number of sites occupied by all species. `sites` gives a list of site names in the object. `species` gives a list of species names in the object. `nodes` gives a node labels or node numbers `coords` returns the spatial coordinates of points or grid cell centers

Author(s)

Michael Krabbe Borregaard

`occurrences` *Access functions to the community matrix of* `distrib_data` *objects*

Description

Gives the species in a site, or the sites occupied by a species, in a `distrib_data` object

Usage

```
occurrences(distrib_data, species, value = c("index", "names", "logical", "raw"))
assemblage(distrib_data, site, value = c("index", "names", "logical", "raw"))
```

Arguments

`distrib_data`	An object of type `distrib_data`, `nodiv_data` or `nodiv_result`
`species`	The number or name of a species in the data set
`site`	The number or name of a site in the data set
`value`	The type of data to return, see below.

Value

`type = "01"` gives a vector of 0s and 1s from the community matrix `type = "logical"` gives a logical vector identifying species or sites `type = "index"` gives the index number of species or sites in the community matrix

Author(s)

Michael Krabbe Borregaard

`phyplot` *Plot Phylogenies*

Description

This is just the ape::plot.phylo method, which is imported and exported directly by the nodiv package. These functions plot phylogenetic trees on the current graphical device.

Usage

```
phyplot(x, type = "phylogram", use.edge.length = TRUE,
  node.pos = NULL, show.tip.label = TRUE, show.node.label = FALSE,
  edge.color = "black", edge.width = 1, edge.lty = 1, font = 3,
  cex = par("cex"), adj = NULL, srt = 0, no.margin = FALSE,
  root.edge = FALSE, label.offset = 0, underscore = FALSE,
  x.lim = NULL, y.lim = NULL, direction = "rightwards",
  lab4ut = NULL, tip.color = "black", plot = TRUE,
 rotate.tree = 0, open.angle = 0, node.depth = 1, align.tip.label = FALSE, ...)
```

Arguments

`x`	an object of class `"phylo"`
`type`	a character string specifying the type of phylogeny to be drawn; it must be one of "phylogram" (the default), "cladogram", "fan", "unrooted", "radial" or any unambiguous abbreviation of these.
`use.edge.length`	a logical indicating whether to use the edge lengths of the phylogeny to draw the branches (the default) or not (if `FALSE`). This option has no effect if the object of class `"phylo"` has no 'edge.length' element.
`node.pos`	a numeric taking the value 1 or 2 which specifies the vertical position of the nodes with respect to their descendants. If `NULL` (the default), then the value is determined in relation to 'type' and 'use.edge.length' (see details).
`show.tip.label`	a logical indicating whether to show the tip labels on the phylogeny (defaults to `TRUE`, i.e. the labels are shown).
`show.node.label`	a logical indicating whether to show the node labels on the phylogeny (defaults to `FALSE`, i.e. the labels are not shown).
`edge.color`	a vector of mode character giving the colours used to draw the branches of the plotted phylogeny. These are taken to be in the same order than the component edge of phy. If fewer colours are given than the length of edge, then the colours are recycled.
`edge.width`	a numeric vector giving the width of the branches of the plotted phylogeny. These are taken to be in the same order than the component `edge` of `phy`. If fewer widths are given than the length of `edge`, then these are recycled.

`edge.lty`	same than the previous argument but for line types; 1: plain, 2: dashed, 3: dotted, 4: dotdash, 5: longdash, 6: twodash.
`font`	an integer specifying the type of font for the labels: 1 (plain text), 2 (bold), 3 (italic, the default), or 4 (bold italic).
`cex`	a numeric value giving the factor scaling of the tip and node labels (Character EXpansion). The default is to take the current value from the graphical parameters.
`adj`	a numeric specifying the justification of the text strings of the labels: 0 (left-justification), 0.5 (centering), or 1 (right-justification). This option has no effect if `type = "unrooted"`. If `NULL` (the default) the value is set with respect of `direction` (see details).
`srt`	a numeric giving how much the labels are rotated in degrees (negative values are allowed resulting in clock-like rotation); the value has an effect respectively to the value of `direction` (see Examples). This option has no effect if `type = "unrooted"`.
`no.margin`	a logical. If `TRUE`, the margins are set to zero and the plot uses all the space of the device (note that this was the behaviour of `plot.phylo` up to version 0.2-1 of 'ape' with no way to modify it by the user, at least easily).
`root.edge`	a logical indicating whether to draw the root edge (defaults to FALSE); this has no effect if 'use.edge.length = FALSE' or if 'type = "unrooted"'.
`label.offset`	a numeric giving the space between the nodes and the tips of the phylogeny and their corresponding labels. This option has no effect if `type = "unrooted"`.
`underscore`	a logical specifying whether the underscores in tip labels should be written as spaces (the default) or left as are (if `TRUE`).
`x.lim`	a numeric vector of length one or two giving the limit(s) of the x-axis. If `NULL`, this is computed with respect to various parameters such as the string lengths of the labels and the branch lengths. If a single value is given, this is taken as the upper limit.
`y.lim`	same than above for the y-axis.
`direction`	a character string specifying the direction of the tree. Four values are possible: "rightwards" (the default), "leftwards", "upwards", and "downwards".
`lab4ut`	(= labels for unrooted trees) a character string specifying the display of tip labels for unrooted trees (can be abbreviated): either `"horizontal"` where all labels are horizontal (the default if `type = "u"`), or `"axial"` where the labels are displayed in the axis of the corresponding terminal branches. This option has an effect if `type = "u"`, `"f"`, or `"r"`.
`tip.color`	the colours used for the tip labels, eventually recycled (see examples).
`plot`	a logical controlling whether to draw the tree. If `FALSE`, the graphical device is set as if the tree was plotted, and the coordinates are saved as well.
`rotate.tree`	for "fan", "unrooted", or "radial" trees: the rotation of the whole tree in degrees (negative values are accepted).
`open.angle`	if `type = "f"` or `"r"`, the angle in degrees left blank. Use a non-zero value if you want to call `axisPhylo` after the tree is plotted.

`node.depth`	an integer value (1 or 2) used if branch lengths are not used to plot the tree; 1: the node depths are proportional to the number of tips descending from each node (the default and was the only possibility previously), 2: they are evenly spaced.
`align.tip.label`	a logical value or an integer. If TRUE, the tips are aligned and dotted lines are drawn between the tips of the tree and the labels. If an integer, the tips are aligned and this gives the type of the lines (lty).
`...`	further arguments to be passed to `plot` or to `plot.phylo`.

Details

The font format of the labels of the nodes and the tips is the same.

If `no.margin = TRUE`, the margins are set to zero and are not restored after plotting the tree, so that the user can access the coordinates system of the plot.

The option ‘node.pos’ allows the user to alter the vertical position (i.e., ordinates) of the nodes. If `node.pos = 1`, then the ordinate of a node is the mean of the ordinates of its direct descendants (nodes and/or tips). If `node.pos = 2`, then the ordinate of a node is the mean of the ordinates of all the tips of which it is the ancestor. If `node.pos = NULL` (the default), then its value is determined with respect to other options: if `type = "phylogram"` then ‘node.pos = 1’; if `type = "cladogram"` and `use.edge.length = FALSE` then ‘node.pos = 2’; if `type = "cladogram"` and `use.edge.length = TRUE` then ‘node.pos = 1’. Remember that in this last situation, the branch lengths make sense when projected on the x-axis.

If `adj` is not specified, then the value is determined with respect to `direction`: if `direction = "leftwards"` then `adj = 1` (0 otherwise).

If the arguments `x.lim` and `y.lim` are not specified by the user, they are determined roughly by the function. This may not always give a nice result: the user may check these values with the (invisibly) returned list (see “Value:”).

If you resize manually the graphical device (windows or X11) you may need to replot the tree.

Value

`plot.phylo` returns invisibly a list with the following components which values are those used for the current plot:

`type`
`use.edge.length`

`node.pos`

`node.depth`

`show.tip.label`
`show.node.label`

`font`

`cex`

`adj`

`srt`

`no.margin`

`label.offset`

`x.lim`

`y.lim`

`direction`

`tip.color`

`Ntip`

`Nnode`

`root.time`

Note

The argument `asp` cannot be passed with `...`.

Author(s)

Emmanuel Paradis

See Also

`plot.phylo`

`plot_nodes_phylo` *Plot a phylogeny with colored node labels*

Description

A wrapper for the plot.phylo function defined in package ape, which is useful for plotting the values at each node

Usage

```
plot_nodes_phylo(variable, tree, label = variable, main = deparse(substitute(variable)),
  zlim = NULL, col, show.legend = TRUE, sig.cutoff, nodes,
  roundoff = TRUE, show.tip.label = NULL, cex = NULL, ...)
```

Arguments

`variable`	The variable to be plotted
`label`	The text to write in each node label
`tree`	The phylogeny to plot, in the ape `phylo` format
`main`	The title text
`zlim`	The values to use for the most extreme colors of the color scale
`col`	The color palette used to create the color scale
`show.legend`	Should a legend for the color scale be plotted?
`sig.cutoff`	The minimum value for nodes to be plotted on the tree, useful for trees with many nodes
`nodes`	The nodes to be plotted on the tree
`roundoff`	Whether to round values when writing them as nodelabel text
`show.tip.label`	Show tip labels on the tree
`cex`	Controls the size of the node symbols
`...`	Further arguments to pass to plot.phylo

Author(s)

Michael Krabbe Borregaard

Examples

```
data(coquettes)
Clade_occupancy <- Node_occupancy(coquettes)
plot_nodes_phylo(Clade_occupancy, coquettes$phylo, cex = 0.7)
```

`plot_points` *Plot a variable on spatial points or a spatial grid using a color scale.*

Description

Plots a variable either as colored points or as a colored raster. `plot_sitestat` is a wrapper function that dispatches the other functions according to the `type` of `distrib_data`.

Usage

```
plot_points(x, coords, col, shape = NULL,
  shapefill = "grey",  zlim= NULL,  zoom_to_points = FALSE,
  pch = 16, bg = par("bg"), legend = TRUE, ...)
plot_grid(x, coords, col, shape = NULL,
  shapefill = "grey",  shapeborder = NA, zlim = NULL,
  zoom_to_points = FALSE, legend = TRUE, gridcol, gridlwd,
  gridsites, overlay_shape = FALSE,
  colscale = c("equal_interval","quantiles"), legendlabels = NULL, ...)
plot_species(distrib_data, species, col = c("darkgreen", "red"), ...)
```

Arguments

`x`	A numerical variable to be plotted using a color scale
`coords`	A matrix or `data.frame` with two columns, or an object inheriting from `SpatialPoints`, of the same length as `x`
`col`	The color palette to use for the color scale. Can be given as color names, a vector of hex colors from a call to a color palette (e.g., `terrain.colors(64)`), or as the name of one of nodiv's internal palettes. Possibillities are `"parula"`, `"jet"`, `"blackbody"`, `"HMblueyellow"`, `"HMrainbow"`,`"HMlinear_optimal"`, `"HMoptimal_scale"`, `"cube1"`, `"cubeyf1"`. Alternatively `col` can be one of `"auto"`, `"ramp"`, `"monochrome"`, `"divergent"` or `"individual"`, allowing `nodiv` to choose a palette of the given type
`shape`	A shape file giving a map to use as a background for the plot
`shapefill`	If shape is a polygon, the color to use to fill the polygons. If shape is a line shapefile, the color to use for plotting lines
`shapeborder`	If shape is a polygon, the color to use for plotting the line border
`zlim`	The values to use for the extremes of the color scale
`zoom_to_points`	Given a shapefile, the function will set the outer limits of the plot to the bounding box of the shapefile. If `zoom_to_points` is `TRUE`, the outer limits will be set by the bounding box of the data points instead.
`pch`	The point character to use for plotting points
`bg`	The central color of points when using a pch value between 21 and 25
`distrib_data`	An object of type `distrib_data`, `nodiv_data` or `nodiv_result` to use for plotting x.
`legend`	Whether to plot a color legend
`species`	The number or name of the species to be plotted
`gridcol`	If specified, plots the sampling grid cell borders in this color
`gridlwd`	If specified, set the line width of overlaid grid cell borders
`gridsites`	If specified, which grid cells to draw cell borders around
`overlay_shape`	Whether to overlay the shape file border on top of the points - only works if shape is a vector shapefile
`colscale`	Specifies whether to define the colors on an equal interval or a quantile-based color scale
`legendlabels`	Specifies custom labels for the colorbar legend
`...`	Additional arguments to pass to plot

Details

Note that if `shape` is a `raster` object, the points may shift relative to the background when resizing the image. This seems to be due to a bug in the `raster` package.

Author(s)

Michael Krabbe Borregaard

`plot_richness` *Plot a richness map for an object of type distrib_data, nodiv_data or nodiv_result*

Description

Create a raster or point plot showing the species richness of an object. Useful for comparing the species richness of sister clades

Usage

```
plot_richness(distrib_data, ...)
plot_node(nodiv_data, node = basal_node(nodiv_data), sites = NULL, ...)
```

Arguments

`distrib_data`	An object of types `distrib_data`, `nodiv_data` or `nodiv_result`
`nodiv_data`	An object of types `nodiv_data` or `nodiv_result`
`node`	The name or internal number of the node to be plotted
`sites`	If `sites = "all"` the plotting region retains all sites (including sites unoccupied by that node)
`...`	Further arguments to pass to the plotting function

Details

The plot will be done as a spatial grid or as colored points, depending on the `type` of the `nodiv_result` object. `plot_richness` plots the richness of the entire dataset, whereas `plot_node` plots the species richness of a given node.

Value

If `distrib_data` has `type = raster`, a raster object of the SOS values is returned.

Author(s)

Michael Krabbe Borregaard

`sitestat` *Site statistics of* `nodiv` *data objects*

Description

Adds, plots or reads site statistics from objects in the `nodiv` package

Usage

```
add_sitestat(distrib_data, site_stat, site = NULL)
plot_sitestat(distrib_data, x, shape = NULL, type = c("auto",
                 "points", "grid"), ...)
sitestat(distrib_data, statname = NULL, site = NULL)
```

Arguments

`distrib_data`	An object of type `distrib_data`, `nodiv_data` or `nodiv_result`
`site_stat`	A site statistic, such as species richness or mean temperature. This can be a vector or a data.frame. If `site` is not specified, the function tries to match `site_stat` to the site names of the `distrib_data` object. The function looks for vector names, rownames, or variables in the data.frame that match the site names. If sitenames are absent and the number of sitestats match, the sitestat is assumed to be in the correct order.
`site`	A character or numeric variable specifying the sites in which to enter or read sitestat
`statname`	The name of a sitestatistic in the `distrib_data` object. If statname is not given, a list of names of sitestatistics is given
`x`	A variable to plot on sites, can either be a numeric variable, or the name of a sitestatistic in the `distrib_data` object
`shape`	A shape file giving a map to use as a background for the plot
`type`	A character vector describing the type of data
`...`	Further arguments to the plotting function

Value

`add_sitestat` returns the object with the new sitestat inside. `sitestat` returns the named sitestatistic.

Author(s)

Michael Krabbe Borregaard

See Also

`distrib_data`

`SOS` *Gives the SOS values for a given node.*

Description

`SOS` gives the SOS value for the node, used in the interpretation of node-based analysis of species distributions after running `Node_analysis`. `plotSOS` plots the values on a map. `GND` gives the GND values.

Usage

```
SOS(nodiv_result, node)
plotSOS(nodiv_result, node, zlim, ...)
GND(nodiv_result, node = NULL)
```

Arguments

`nodiv_result`	The result of running `Node_analysis`
`node`	The node label or node number specifying the node to plot. Optional for `GND`
`zlim`	The values to use for the extremes of the color palette
`...`	Further arguments to pass to the plotting function

Details

The first descendant clade (i.e. the first returned by a call to `Descendants`) is associated with positive SOS values, the second descendant clade is associated with negative values. The plot will be done as a spatial grid or as colored points, depending on the `type` of the `nodiv_result` object. If the `nodiv_result` object has a `shape` object of type `raster`, this will be plotted normally in the background - but resizing the window will cause the plots/pixels and the underlying raster object to lose alignment, due to a bug in the `raster` package.

Value

`SOS` returns a numerical vector with the SOS values. If `nodiv_result` has `type` `raster`, a raster object of the SOS values is returned by `plotSOS`.

Author(s)

Michael Krabbe Borregaard

References

Borregaard, M.K., Rahbek, C., Fjeldsaa, J., Parra, J.L., Whittaker, R.J. & Graham, C.H. 2014. Node-based analysis of species distributions. Methods in Ecology and Evolution 5(11): 1225-1235. DOI: 10.1111/2041-210X.12283

See Also

`Node_analysis`

`species_stat` *Species statistics of* `nodiv` *data objects*

Description

Adds, or reads species statistics from objects in the `nodiv` package

Usage

```
add_species_stat(distrib_data, species_stat, specs = NULL)
species_stat(distrib_data, statname = NULL, specs = NULL)
```

Arguments

`distrib_data`	An object of type `distrib_data`, `nodiv_data` or `nodiv_result`
`species_stat`	A species statistic, such as body size or cold tolerance. This can be a vector or a data.frame. If `specs` is not specified, the function tries to match `species_stat` to the site names of the `distrib_data` object. The function looks for vector names, rownames, or variables in the data.frame that match the species names. If species names are absent and the number of species_stats match, the species_stat is assumed to be in the correct order.
`specs`	A character or numeric variable specifying the species of interest, or for which species_stat is specified
`statname`	The name of a species statistic in the `distrib_data` object. If statname is not given, a list of names of species statistics is given

Value

`add_species_stat` returns the object with the new species_stat inside. `species_stat` returns the named sitestatistic.

Author(s)

Michael Krabbe Borregaard

See Also

`distrib_data`

`subsample` *Subsampling data objects to certain species or sites*

Description

Takes an object of type `distrib_data` or `nodiv_data`, and subsamples it to contain certain species or sites, while keeping the integrity of the data.

Usage

```
subsample(x, ...)
## S3 method for class 'distrib_data'
subsample(x, sites = NULL, species = NULL, ...)
## S3 method for class 'nodiv_data'
subsample(x, sites = NULL, species = NULL, node = NULL, ...)
## S3 method for class 'nodiv_result'
subsample(x, node = NULL, ...)
```

Arguments

`x`	An object of type `distrib_data` or `nodiv_data`
`sites`	A numeric or character vector identifying the sites to keep in the new object. If specified, the function will subsample the data object to only include the defined sites, and the species that exist there. If sites == \"all\", all sites will be kept when subsampling for species, even if no species now exist in the sites.
`species`	A numeric or character vector identifying the species to keep in the new object. If specified, the function will subsample the data object to only include the defined species, and the sites where they exist.
`node`	A numeric or character vector identifying a node in the phylogeny. Only species descending from this node will be kept in the new object, and the sites where they exist.
`...`	Further arguments to the subsample function

Value

The return value is a new object of the same type as `x`

Author(s)

Michael Krabbe Borregaard

See Also

`distrib_data`, ~~~

Examples

```
## Not run:
data(coquettes)
new <- subsample(coquettes, sites = which(richness(coquettes) > 3))
plot(new, show.tip = FALSE)
## End(Not run)
```

`two_color_map` *Plot the relative amounts of two variables using a purple-green color scheme*

Description

Plots site statistics from objects in the `nodiv` package

Usage

```
two_color_map(distrib_data, vec1, vec2, res = 10, showlegend = T,
  legend_coords = c(0.2, 0.26, 0.36, 0.44), type = c("auto", "grid", "points"),
  colour = c("green_purple", "green_red_purple_cyan"), ...)
```

Arguments

`distrib_data`	An object of type `distrib_data`, `nodiv_data` or `nodiv_result`, or, alternatively, an object of spatial points from package `sp`, or a matrix of xy values in two columns
`vec1`	A site statistic to be plotted, such as species richness or mean temperature. This can be a vector of values or the name of a site statistic in distrib_data.
`vec2`	The oter site statistic to be plotted.
`res`	The resolution of colors on the color legend
`showlegend`	A boolean specifying whether to show the legend
`legend_coords`	A vector x1, x2, y1, y2 specifying the location and extent of the color legend
`type`	To specify whether to plot as spatial points or a grid
`colour`	What colour scheme to use for plotting
`...`	Further arguements to the plotting function

Value

A data.frame giving codes and indices of the colors shown in the plot

Author(s)

Michael Krabbe Borregaard

See Also

`distrib_data` `plot_points`

`update_object` *Update* `distrib_data` *objects to the* `nodiv` *version >= 1.1 format*

Description

Creates the `species_stats` element

Usage

```
update_object(distrib_data)
```

Arguments

`distrib_data`	An object of type `distrib_data`, `nodiv_data` or `nodiv_result`

Value

An updated version of the object

Author(s)

Michael Krabbe Borregaard

Index

∗ **datasets**
    coquettes, 5
∗ **package**
    nodiv-package, 2

add_shape, 3
add_sitestat (sitestat), 21
add_species_stat (species_stat), 23
assemblage (occurrences), 13
axisPhylo, *15*

basal_node, 4

coords (Nspecies), 12
coquettes, 5

Descendants (basal_node), 4
distrib_data, *3*, 5, *8*, *21*, *23*, *24*, *26*

GND (SOS), 22
gridData, 7

identify.distrib_data (distrib_data), 5

MostRecentAncestor (basal_node), 4

Node_analysis, *7*, *9*, 9, *23*
Node_occupancy (Node_size), 11
Node_sites (Node_size), 11
Node_size, 11
Node_species (Node_size), 11
nodenumbers (basal_node), 4
nodes (basal_node), 4
Nodesig, 8
nodiv (nodiv-package), 2
nodiv-package, 2
nodiv_data (distrib_data), 5
nodiv_result (Node_analysis), 9
Nsites (Nspecies), 12
Nspecies, 12

occupancy (Nspecies), 12
occurrences, 13

Parent (basal_node), 4
phyplot, 14
plot.distrib_data (distrib_data), 5
plot.nodiv_data (distrib_data), 5
plot.nodiv_result (Node_analysis), 9
plot.phylo, *17*
plot_grid (plot_points), 18
plot_node (plot_richness), 20
plot_nodes_phylo, 17
plot_points, 18, *26*
plot_richness, 20
plot_sitestat (sitestat), 21
plot_species (plot_points), 18
plotSOS, *11*
plotSOS (SOS), 22
print.distrib_data (distrib_data), 5
print.nodiv_data (distrib_data), 5
print.nodiv_result (Node_analysis), 9
print.summary_nodiv_result (Node_analysis), 9

richness (Nspecies), 12

Sister (basal_node), 4
sites (Nspecies), 12
sitestat, 21
SOS, 22
species (Nspecies), 12
species_stat, 23
subsample, 24
summary.distrib_data (distrib_data), 5
summary.nodiv_data (distrib_data), 5
summary.nodiv_result (Node_analysis), 9

two_color_map, 25

update_object, 26