Package 'phylorug'

November 6, 2026

Title Visualize Clade Recovery and Support Across Phylogenetic Trees

Version 0.1.0

Description Generates rug plot visualizations directly on tree nodes to illustrate clade recovery and support from multiple phylogenetic and phylogenomic analyses. As modern studies routinely employ diverse datasets and inference techniques producing potentially conflicting trees, 'phylorug' simplifies the comparison among these trees by consolidating clade recovery and support on a reference tree.

License MIT + file LICENSE

URL https://github.com/mdrifathahamed/phylorug, https://mdrifathahamed.github.io/phylorug/

BugReports https://github.com/mdrifathahamed/phylorug/issues

Depends R (>= 4.1.0)

Imports ape, graphics, grDevices, phangorn, stats, tools, utils

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

Config/testthat/edition 3

Encoding UTF-8

LazyData true

RoxygenNote 8.0.0

VignetteBuilder knitr

NeedsCompilation no

Author Md Rifath Ahamed [aut, cre, cph] (ORCID: <https://orcid.org/0009-0005-6158-8658>), J. Salvador Arias [aut] (ORCID: <https://orcid.org/0000-0002-3717-435X>), Sergei Tarasov [aut] (ORCID: <https://orcid.org/0000-0002-1737-9403>)

Maintainer Md Rifath Ahamed <rifathahamed777@gmail.com>

Repository CRAN

Date/Publication 2026-11-06 16:50:02 UTC

Contents

add_tree 2
check_taxa 4
node_presence_matrix 6
plot_phylorug 8
prune_to_shared 13
read_trees 14
sample_trees 16
translate_tips 17

Index **19**

`add_tree` *Add a comparison tree to an existing node presence matrix*

Description

Adds a new comparison tree to an existing node presence matrix without rerunning the whole pipeline from scratch. The function computes clade recovery (presence) and support for the new tree only and attaches the results as a new column in every matrix of the npm. All existing columns remain unchanged.

Usage

```
add_tree(
  npm,
  backbone,
  new_tree,
  name,
  support_col = 1,
  support_type = NULL,
  pool_threshold = 1
)
```

Arguments

`npm`	A node presence matrix list, as returned by `node_presence_matrix()` or a previous call to `add_tree()`.
`backbone`	The backbone tree used to create `npm`. Must be the same `phylo` object (same topology and tip labels) that was passed to `node_presence_matrix()`.
`new_tree`	A `phylo` or `multiPhylo` object representing the new comparison analysis to add. Must be rooted.
`name`	Character string. Column name for the new tree in the npm matrices. Required. Must be a single non-empty string and must not duplicate an existing column name.

`support_col` Integer or vector of integers (max 3). Which support value(s) to extract from the new tree's node labels, matching the `support_col` used when the original `npm` was built. Default is `1`. For example, if the new tree is from IQ-TREE with compound labels `"80/95"`, passing `c(1, 2)` extracts both metrics.

`support_type` Optional character string naming the support metric of the new tree (e.g. `"ufboot"`, `"lpp"`, `"jackknife"`). If supplied, it is appended to the npm's `"support_type"` attribute so that `plot_phylorug()` can apply metric-specific thresholds automatically. If `NULL`, the new tree's support values will be auto-normalized (values >1 divided by 100) and binned against universal thresholds at plot time. For this reason, raw Bremer support (an unbounded integer) is not supported; use `"bremer_ratio"` instead, which is bounded between 0 and 1.

`pool_threshold` Numeric between 0 and 1. Controls how pools are scored. Default `1.0` (strict consensus). Should match the value used when the original npm was built. See `node_presence_matrix()` for details. This parameter is only applicable when evaluating parsimony-based `multiPhylo` objects (e.g., Most Parsimonious Trees generated via TNT or similar software).

Details

Three taxon scenarios are handled automatically:

Identical The new tree shares all backbone taxa exactly. Every clade is evaluated normally.

Superset The new tree contains all backbone taxa plus extras. The extra taxa are pruned internally so that clade keys match the backbone's. Without this pruning, a backbone clade `(A, B)` would fail to match a new-tree node grouping `(A, B, X)`, producing a false absence. A message reports the pruning. The user's original tree object is not modified.

Missing The new tree lacks one or more backbone taxa. Any backbone clade containing a missing taxon cannot be evaluated in this tree and is set to `NA` in both the presence and support matrices. These appear as red ("not computed") cells in the rug plot. A message lists the missing taxa and suggests rebuilding the npm from scratch with `prune_to_shared()` if the red cells are undesirable. Unlike `node_presence_matrix()`, which errors on missing taxa, `add_tree()` is deliberately permissive: when adding a tree after the initial analysis, different taxon sampling is a legitimate use case, not a data error.

The new tree must be rooted. If it is a `multiPhylo` object (a pool of equally optimal trees from one search), clade presence is determined by `pool_threshold`: the default (`1.0`, strict consensus) requires the clade in every pool tree; `0.5` applies majority rule; `0` records the raw proportion. Support extraction is skipped for pools because averaging node labels across equally optimal trees is not scientifically valid; support cells for that column will be `NA`. To include support values from a parsimony analysis, compute the consensus tree with bootstrap or jackknife support upstream (e.g. in TNT) and pass that consensus as a single `phylo` object.

If `support_type` is supplied, it is appended to the `"support_type"` attribute stored in the npm object, keyed by the new tree's name. This allows `plot_phylorug()` to apply the correct metric-specific thresholds without the user re-declaring the full vector.

Value

The updated npm list with the new tree appended as an additional column in the `presence` matrix and each `support_*` matrix. The `"pool_sizes"` attribute is updated, and `"support_type"` is

extended if the argument was supplied.

See Also

node_presence_matrix() to build the initial npm, plot_phylorug() to visualise the result, and check_taxa() to diagnose taxon overlap before adding.

Examples

```
# Build an npm with 3 of the 5 shipped trees:
backbone <- sample_trees[["70p_uce"]]
others   <- sample_trees[c("70p_partition_entropy", "70p_ghost")]
npm      <- node_presence_matrix(backbone, others)
ncol(npm$presence)   # 2 comparison trees

# Add a fourth tree without re-running the pipeline:
npm <- add_tree(npm, backbone,
                sample_trees[["70p_ASTRAL_uce"]],
                name = "ASTRAL_uce")
ncol(npm$presence)   # now 3

# Add the fifth tree from a file, with a declared support type.
# Here a shipped tree is first written to a temporary file; with your own
# data, pass the path to your tree file. The tree must be rooted
# (see ape::root()).
tree_file <- tempfile(fileext = ".tre")
ape::write.tree(sample_trees[["70p_ASTRAL_partition_entropy"]], tree_file)
new_tree <- ape::read.tree(tree_file)
npm <- add_tree(npm, backbone, new_tree,
                name = "ASTRAL_part_entropy",
                support_type = "lpp")
ncol(npm$presence)   # now 4
attr(npm, "support_type")
unlink(tree_file)
```

---

check_taxa *Diagnose taxon consistency between backbone and comparison trees*

Description

Checks whether the backbone and comparison trees share the same set of taxa or not. If a comparison tree is missing a backbone taxon, every backbone clade that includes that taxon would be falsely scored as absent, node_presence_matrix() prevents this by stopping with an error, but this function helps you diagnose the mismatch before that happens. If you already know all trees share the same taxa, skip straight to node_presence_matrix().

Usage

```
check_taxa(backbone, trees, verbose = TRUE)
```

Arguments

backbone	The reference tree, as a "phylo" object. The rug is drawn on this tree, and its clades are searched for in each comparison tree.
trees	A named list of "phylo" or "multiPhylo" objects, one per comparison, as returned by read_trees().If the backbone is present in the list it is removed automatically before comparison, so you can pass the full list from read_trees() without subsetting first.
verbose	Logical. If TRUE (default), reports the outcome and names any mismatched comparison trees.

Details

This function reports; it does not modify the trees. If the backbone is present in comparison trees it is silently removed before comparison. Three outcomes are

identical All comparison trees share the backbone's taxa exactly.

superset One or more comparison trees contain every backbone taxon plus additional taxa. All backbone clades can still be evaluated; the extra taxa are ignored by node_presence_matrix().

missing One or more comparison trees lack backbone taxa. A clade cannot be searched in a tree that does not contain all of its taxa, so node_presence_matrix() will stop with an error. Use prune_to_shared() to reduce all trees to their common taxon set before building the matrix.

Where a comparison tree is a pool of several equally optimal trees, every tree in the pool is checked. A pool whose trees disagree about their own taxa is a data problem, and is an error.

Value

A single logical value: TRUE if every comparison tree shares the backbone's taxa exactly. The value carries a "diagnostics" attribute, a data frame with one row per comparison tree giving its status ("identical", "superset" or "missing") and the taxa missing from, or extra to, the backbone. Diagnostics are attached whatever verbose is set to.

See Also

read_trees() to load the trees, translate_tips() to harmonize naming conventions, and prune_to_shared() to reduce trees to a common taxon set.

Examples

```
# sample_trees is a named list of 5 phylo objects shipped with phylorug,
# the same structure you get from read_trees(). Pick one analysis as the
# backbone, and the rest as comparison trees:
backbone <- sample_trees[["70p_uce"]]
others   <- sample_trees[names(sample_trees) != "70p_uce"]

# Diagnose taxon consistency (verbose = TRUE by default reports the outcome):
result <- check_taxa(backbone, others)

# If any tree is missing or has extra taxa, inspect the full report:
attr(result, "diagnostics")
```

`node_presence_matrix` *Tabulate clade recovery and support across trees*

Description

Compares a set of phylogenetic trees against a reference topology, the `"backbone"`. For each internal node of the backbone, the function asks whether the same clade appears in each tree, and records either its presence or its support value.

Usage

```
node_presence_matrix(
  backbone,
  trees,
  support_col = 1,
  support_type = NULL,
  pool_threshold = 1
)
```

Arguments

`backbone`	The reference tree, as a `"phylo"` object. The rug is drawn on this tree, and its clades are searched for in each comparison tree.
`trees`	A named list of `"phylo"` or `"multiPhylo"` objects, one per comparison, as returned by `read_trees()`.If the backbone is present in the list it is removed automatically before comparison, so you can pass the full list from `read_trees()` without subsetting first.
`support_col`	Integer or vector of integers (max 3). Specifies which value(s) to extract from multi-metric node labels. For example, if an IQ-TREE tree stores SH-aLRT and UFBoot as "80/95", passing `1` extracts only the first metric into a `support_1` matrix. Passing `c(1, 2)` efficiently extracts both simultaneously into `support_1` and `support_2` matrices, allowing user to easily switch between them during plotting without recalculating. Default is `1` .
`support_type`	Optional named character vector mapping comparison trees to their support metrics (e.g. `"ufboot"`, `"sh_alrt"`, `"lpp"`, `"jackknife"`). Stored as an attribute of the returned list and used automatically by `plot_phylorug()` when `mode = "support"`. If omitted, `plot_phylorug()` will auto-normalize support values and apply universal thresholds.
`pool_threshold`	Numeric between 0 and 1. Controls how pools of equally optimal trees are scored. `1.0` (default) = strict consensus, `0.5` = majority rule, `0` = raw proportion. Only applies to `multiPhylo` comparisons (e.g. MPTs from TNT or PAUP*).

Details

This is the core function of the phylorug pipeline. It can be called directly after `read_trees()`. Running `check_taxa()` first is recommended but not required `node_presence_matrix()` will stop with an error if any comparison tree is missing backbone taxa.

Every tree must include the complete set of backbone taxa. Because a tree lacking a backbone taxon cannot be assessed for the presence of a clade containing that taxon, scoring such a clade as 'absent' would introduce a false negative. The function therefore enforces strict taxon matching. Use `check_taxa()` to diagnose the discrepancies and `prune_to_shared()` to harmonize the backbone and comparison tree taxa.

The function always computes both a presence matrix and a support matrix in a single pass. The plotting function `plot_phylorug()` decides which to use based on the visualisation context.

The `presence` matrix records clade recovery: `1` where the bipartition is found, `0` where absent. For multiple equally most parsimonious trees (MPTs), the behaviour depends on `pool_threshold`: the default (`1.0`, strict consensus) requires the clade to appear in every pool tree; `0.5` applies majority rule; `0` records the raw proportion.

Each `support_*` matrix records the raw support value from the node label of the tree that recovered the clade (e.g. bootstrap, posterior probability). Support values are not computed for pools of equally optimal trees (`multiPhylo`), only presence mode is available for these, and their support cells will be `NA`. If you need support mode for a parsimony analysis, build a consensus tree with support values mapped onto it (e.g. bootstrap on the strict consensus) in your upstream analysis, and pass that single tree to phylorug instead.

Value

A named list with one row per internal backbone node and one column per comparison tree:

presence Clade presence: `1` where recovered, `0` where absent. For pools, the value depends on `pool_threshold` (see above).

```
\item{support matrices}{One matrix per value in `support_col`
 (named `support_1`, `support_2`, etc.). Raw support values where
 the clade was recovered, `NA` where absent or where the clade was
 recovered but no support value could be parsed from the node label.}
```

See Also

`check_taxa()` to diagnose taxon mismatches before building the matrix, `plot_phylorug()` to visualize the result, and `add_tree()` to append additional comparison trees to an existing matrix without recomputing it from scratch.

Examples

```
# phylorug ships `sample_trees`: a named list of 5 phylo objects (beetles),
# the same structure you get from read_trees(). Just type `sample_trees`.
# Pick one analysis as the backbone, and the rest as comparison trees:
backbone <- sample_trees[["70p_uce"]]
others   <- sample_trees[names(sample_trees) != "70p_uce"]

# Optional: diagnose taxon overlap before building the matrix.
# This is not required, node_presence_matrix() enforces matching internally.
check_taxa(backbone, others)

# --- Presence/absence matrix ------------------------------------------------
# Just pass the backbone and comparison trees (support_col defaults to 1).
```

```
# For pools of MPTs, pool_threshold defaults to 1.0 (strict consensus):
npmatrix <- node_presence_matrix(backbone, others)
npmatrix$presence     # clade presence/absence (1/0)
npmatrix$support_1    # support values (from column 1 of the node labels)

# --- Multiple support metrics at once -------------------------------------
# IQ-TREE stores SH-aLRT and UFBoot2 as "80/95". Pass their column positions
# to support_col to extract both in a single pass:
rugmt <- node_presence_matrix(backbone, others, support_col = c(1, 2))
rugmt$support_1       # first metric (SH-aLRT)
rugmt$support_2       # second metric (UFBoot2)

# --- With support_type (stored in npm, used by plot_phylorug) --------------
# Declaring support_type lets plot_phylorug() apply metric-specific
# thresholds automatically. If omitted, universal 0-1 thresholds are used.
support_type <- c(
  "70p_ASTRAL_partition_entropy" = "lpp",
  "70p_ASTRAL_uce"               = "lpp",
  "70p_ghost"                    = "ufboot",
  "70p_partition_entropy"        = "ufboot"
)
npm_typed <- node_presence_matrix(backbone, others,
                                  support_type = support_type)
attr(npm_typed, "support_type")   # stored as an attribute
```

---

plot_phylorug *Draw a phylorug: a backbone tree with rug plots on every internal node*

Description

plot_phylorug() overlays rug plots on a backbone phylogeny, compares how multiple analyses treat each internal node. In presence mode, cells are black (recovered) or white (absent). In support mode, cells are shaded by binned support strength. The function handles canvas sizing, legend placement, and font scaling automatically.

Usage

```
plot_phylorug(
  backbone,
  npm,
  file = NULL,
  width = NULL,
  height = NULL,
  mode = c("presence", "support"),
  support_idx = 1,
  thresholds = NULL,
  n_rows = NULL,
  n_cols = NULL,
```

```
  include_backbone = FALSE,
  nodes = NULL,
  legend = TRUE,
  show_support = TRUE,
  show_support_idx = NULL,
  cell_scale = 0.45,
  x_offset = 0,
  y_offset = 0,
  rug_position = c("inside", "outside"),
  dot_identical = TRUE,
  dot_col = "black",
  dot_cex = NULL,
  support_label_cex = NULL,
  support_label_col = "red",
  dot_on_very_high = TRUE,
  hide_unsupported = TRUE,
  ...
)
```

Arguments

backbone	A phylo object representing the backbone tree.
npm	A named list containing the presence and support matrices, returned by node_presence_matrix().
file	Optional character string specifying the output file name or full directory path. If left as NULL (the default), the plot renders in the active graphics device (e.g., RStudio) for quick drafting. To avoid aspect ratio distortion caused by GUI exports and to generate perfectly scaled, publication-ready figures, provide a file path here (must end in .pdf, .png, or .jpg) to utilize the package's internal scaling engine.
width, height	Numeric. Optional canvas dimensions in inches, applied only when exporting to a file.If left as NULL (the default), the package's internal engine dynamically calculates the optimal canvas dimensions based on the tree size and legend layout.
mode	One of "presence" (default) or "support".
support_idx	Integer (1, 2, or 3). Default is 1. Specifies which single support matrix from the npm list to visualize. For example, if you generated the npm using support_col = c(1, 2), passing 2 here tells the plotting engine to physically shade the grid cells using the second metric (stored in your list as support_2).
thresholds	Optional list overriding the built-in bin thresholds used in support mode. Support values are binned into four levels: very high, high, moderate, and low. Each entry supplies the lower bounds for the top three levels as a named numeric vector with exactly three names: very_high, high, and moderate. Any value below moderate is automatically binned as low, so no fourth cutpoint is needed. Key each entry by metric name when support_type was declared in node_presence_matrix(), or by "universal" when it was not. Examples: thresholds = list(ufboot = c(very_high = 97, high = 85, moderate = 50)) means UFBoot2 values >= 97 are very high, 85–96 are high, 50–84 are

moderate, and < 50 are low. `thresholds = list(universal = c(very_high = 0.90, high = 0.70, moderate = 0.50))` applies a single scale to all trees. If `NULL` (default) and `support_type` was declared, default thresholds for each metric are used. If neither `support_type` nor `thresholds` is provided, universal thresholds (0.95 / 0.80 / 0.50) are applied.

`n_rows, n_cols` Integer. Grid shape for the rug at each node. If `NULL` (the default), a roughly square grid is chosen automatically.

`include_backbone`

Logical or character. If `FALSE` (default), the backbone does not occupy a rug cell. If `TRUE`, the backbone is added as cell 1 using universal thresholds in support mode (with a message). If a character string naming a support metric (`include_backbone = "ufboot"`), the backbone is added as cell 1 and binned against that metric's thresholds.

`nodes` Optional. Restricts the plot to specific backbone nodes. Three ways to specify:

- **By taxa (recommended):** a list of character vectors, each with >= 2 tip labels. phylorug finds the MRCA of each set. Example: `nodes = list(c("sp_A", "sp_B"), c("sp_C", "sp_D"))`.
- **By node ID (advanced):** a numeric vector of ape internal node IDs, as shown in `rownames(npm$presence)`. To see which taxa belong to a node, use `ape::extract.clade(backbone, node_id)$tip.label`.
- **By node label:** a character vector matching `backbone$node.label`.

If `NULL` (default), all internal nodes are plotted. When supplied, only selected nodes get dots, rugs, or support labels.

`legend` Logical. Default `TRUE`. Controls both legends: the position legend (top-left, numbered grid mapping each cell to its analysis) and, in support mode, the threshold legend (top-right, colour key for the four support bins). Set to `FALSE` to suppress both, for example when annotating the figure manually in Inkscape or Illustrator.

`show_support` Logical. If `TRUE` (default), backbone node support labels are drawn beside each node, and the postion legend includes a colored line naming what those numbers are.

`show_support_idx`

Optional. Controls what appears in the label on the tree AND what the legend calls it. Accepts several shapes:

- `NULL` (default): the raw `node.label` is drawn on the tree and the legend says `<value> (backbone support)` without naming metrics.
- An integer (`1` or `2`): only that slot of a compound label (e.g. "80/95") is drawn, and the legend shows just that number.
- A named integer, e.g. `c("1" = "sh_alrt")`: same slot behaviour, and the legend adds the metric name (e.g. "SH-aLRT").
- Two named integers, e.g. `c("1" = "sh_alrt", "2" = "ufboot")`: the full compound is drawn and the legend names both metrics.
- A plain string, e.g. `"UFBoot2"`: legal only when the tree has single-value labels (no `/`). On compound labels, this is ambiguous (which slot?) and the function falls back to the raw-label behaviour with a message explaining the shape it accepts.

Recognised metric keys for named-integer form: "ufboot", "sh_alrt", "lpp", "posterior", "jackknife", "bootstrap", "bremer_ratio", "transfer_boot". Any other key is shown verbatim in the legend.

`cell_scale` Numeric multiplier on cell height. Default 0.45.

`x_offset, y_offset`
Numeric grid shift. Default 0.

`rug_position` One of "inside" (default) or "outside". Controls where the node rug grid is placed relative to the backbone node: "inside" tucks the grid into the crook above-left (toward the root), while "outside" places the grid to the right of the node (toward the tips).

`dot_identical` Logical. Default TRUE. **Presence mode only.** draws a small dot on backbone nodes where every comparison tree is identical in recovering the clade. Has no effect in support mode; see dot_on_very_high for the support-mode equivalent.

`dot_col, dot_cex`
Colour and size of the identical-clade dot. Defaults are "black" and NULL (auto-scales). Shared by both dot_identical (presence mode) and dot_on_very_high (support mode).

`support_label_cex`
Numeric or NULL. Size of the backbone support labels. Default NULL auto-scales with tree size.

`support_label_col`
Colour of the backbone support labels. Default "red".

`dot_on_very_high`
Logical. Default TRUE. **Support mode only.** Controls what a dot means there. When TRUE, a dot is drawn only where a clade is recovered by every comparison analysis AND every one of those analyses independently rates it very-high support. Anything less (not recovered everywhere, or recovered everywhere but not uniformly very-high) is drawn as a rug instead. Set to FALSE to turn off dots entirely in support mode: every node is then drawn as a rug, however strong or weak its support, the most conservative and transparent option. Has no effect in presence mode; see dot_identical for the presence-mode equivalent.

`hide_unsupported`
Logical. Default TRUE. Nodes where no comparison tree recovers the clade are left bare, the absence of both a dot and a rug signals that the clade is unique to the backbone topology. Set to FALSE to draw all-white rugs at these nodes, which can be useful when you want every internal node to carry a visible grid for annotation or figure editing.

`...` Additional arguments passed to ape::plot.phylo(), such as cex, edge.width, font, or label.offset. These override the automatic scaling when provided.

Details

Each internal node is drawn as either a dot (compact summary) or a rug plot (full per-analysis grid). What a dot means depends on the mode. In presence mode, a dot means every comparison analysis recovered the clade; control this with dot_identical, setting it to FALSE to show rug plots on every node, even unanimous ones. In support mode, a dot means every comparison analysis recovered the

clade and every one independently rates it very-high support, control this with `dot_on_very_high`, setting it to `FALSE` to show rug plots on every node regardless of support level. Dot appearance (`dot_col`, `dot_cex`) is shared across both modes.

Value

Invisibly, the file path if a file was written, or `NULL` if plotted directly to the active graphics device.

See Also

`node_presence_matrix()` to build the input data, `check_taxa()` to verify taxon sets, and `add_tree()` to append trees to an existing matrix.

Examples

```
# Build the plotting input from real trees:
backbone <- sample_trees[["70p_uce"]]
others   <- sample_trees[names(sample_trees) != "70p_uce"]
support_type <- c(
  "70p_ASTRAL_partition_entropy" = "lpp",
  "70p_ASTRAL_uce"               = "lpp",
  "70p_ghost"                    = "ufboot",
  "70p_partition_entropy"        = "ufboot"
)
npm_st <- node_presence_matrix(backbone, others,
                               support_col = 1,
                               support_type = support_type)

# --- Presence mode ------------------------------------------------------
# Each cell shows whether an analysis recovered the backbone clade.
tmp <- tempfile(fileext = ".pdf")
plot_phylorug(backbone, npm_st, file = tmp)
unlink(tmp)

# --- Support mode -------------------------------------------------------
tmp2 <- tempfile(fileext = ".pdf")
plot_phylorug(backbone, npm_st,
              file         = tmp2,
              mode         = "support",
              support_idx  = 1)
unlink(tmp2)

# --- Some optional controls ---------------------------------------------
tmp3 <- tempfile(fileext = ".pdf")
plot_phylorug(backbone, npm_st,
              file             = tmp3,
              include_backbone = TRUE,
              rug_position     = "outside")
unlink(tmp3)
# --- Restrict to specific nodes -----------------------------------------
# By taxa: name two or more tips whose MRCA is the node you want.
# Useful when you know which species define a clade of interest.
```

```
tmp4 <- tempfile(fileext = ".pdf")
plot_phylorug(backbone, npm_st,
              file  = tmp4,
              nodes = list(c("Sisyphus_schaefferi_STL44",
                             "Sisyphus_muricatus__STL5")))
unlink(tmp4)

# By node ID: useful when exploring interactively after inspecting
# rownames(npm_st$presence) or ape::nodelabels() output.
tmp5 <- tempfile(fileext = ".pdf")
plot_phylorug(backbone, npm_st,
              file  = tmp5,
              nodes = as.integer(rownames(npm_st$presence))[1:2])
unlink(tmp5)
```

`prune_to_shared` *Trim the backbone and its comparison trees to their shared set of taxa*

Description

First finds the taxa that the backbone and all comparison trees share, then trims everything else from the backbone and from every comparison tree.

Usage

```
prune_to_shared(backbone, trees, verbose = TRUE)
```

Arguments

`backbone`	A phylogenetic tree of class `"phylo"`.
`trees`	A named list of `"phylo"` or `"multiPhylo"` objects, as returned by `read_trees()`.
`verbose`	Logical. If `TRUE` (default), reports how many taxa were dropped and names them.

Details

A clade cannot be compared across trees that were not run on the same taxa. Where a comparison tree is missing a backbone taxon, every backbone clade containing that taxon becomes unevaluable in that tree, and `node_presence_matrix()` refuses to proceed. Trimming to the shared taxa resolves this by making every tree contain the same taxa.

The cost is that the questions become narrower. Any clade containing a dropped taxon no longer exists on the backbone and cannot be reported, even where most comparison trees recovered it. Check the report from `check_taxa()` before trimming: if only one or two taxa are involved the loss is minimal, but if several comparison trees each lack a different taxon, the shared set can shrink quickly.

Value

A list with two elements: `backbone`, the pruned backbone tree, and `trees`, the pruned comparison trees, retaining their names and their original single-tree or pooled structure. The list carries a `"dropped"` attribute naming the taxa that were removed.

See Also

`check_taxa()` to see which taxa are missing and from where, before deciding to prune.

Examples

```
# Build a backbone and comparison trees that do NOT all share the same taxa,
# so there is something to trim. (In practice these come from read_trees().)
backbone <- ape::read.tree(
  text = "((((A,B),(C,D)),((E,F),(G,H))),(I,J));"
)
others <- list(
  # All ten taxa, same as the backbone:
  tree_1 = ape::read.tree(text = "((((A,B),(C,D)),((E,F),(G,H))),(I,J));"),
  # Missing J:
  tree_2 = ape::read.tree(text = "((((A,B),(C,D)),((E,F),(G,H))),I);"),
  # Missing I:
  tree_3 = ape::read.tree(text = "((((A,B),(C,D)),((E,F),(G,H))),J);")
)

# See which taxa are missing, and from which trees, before trimming:
check_taxa(backbone, others)

# Trim the backbone and every comparison tree to their shared taxa:
shared <- prune_to_shared(backbone, others)

# Which taxa were dropped:
attr(shared, "dropped")

# The pruned trees are ready for node_presence_matrix():
shared$backbone
shared$trees
```

---

read_trees *Import phylogenetic trees from a directory*

Description

Scans a directory for phylogenetic tree files, parses each one, and returns them as a named list. Format is detected from file content, so a directory may mix Newick and NEXUS files.

Usage

```
read_trees(
  dir = ".",
  ext = c("tre", "tree", "treefile", "nwk", "newick", "nex", "nexus", "contree"),
  format = c("auto", "newick", "nexus"),
  verbose = TRUE
)
```

Arguments

dir	Path to the directory containing the tree files. Defaults to the working directory.
ext	Character vector of file extensions to look for, without the leading dot (e.g. "tre", not ".tre"). Case-insensitive. The default covers common tree extensions (tre, tree, treefile, nwk, nex, nexus, contree). Pass your own to narrow or widen the search.
format	The parsing strategy to use. Defaults to "auto", which detects the format of each file individually, so a directory may mix formats. Use "newick" or "nexus" to force one parser for all files.
verbose	Logical. If TRUE (default), reports trees read, files skipped, and pooled analyses. Errors are always reported.

Details

Each file is one analysis. A file holding a single tree is returned as a "phylo" ; a file holding several equally optimal trees from one search (as from POY, TNT, or PAUP*) is returned as a "multiPhylo" and is scored as a pool. How many pool trees must recover a clade before it counts as present is controlled by the pool_threshold argument in node_presence_matrix().

Do not supply posterior samples, bootstrap replicates, or sets of gene trees. These are distributions rather than analyses and must be summarised in the upstream analysis before use (e.g. into a consensus tree). Files holding more than 100 trees are rejected with an error.

Support values stored as numeric node labels in the tree file (e.g. ((A,B)95,(C,D)100);) are read normally. BEAST-style bracket annotations ([&posterior=0.98]) are discarded by **ape**; the topology is unaffected, and a message is emitted. Support values are imported exactly as written in the tree file; read_trees() does not recompute or verify them.

Files matching ext that contain no tree, such as a NEXUS character matrix, are skipped with a message. A tree file that fails to parse is an error.

Value

A named list, one element per tree file, named after the file names with extensions removed. Single-tree files yield "phylo" objects; multi-tree files yield "multiPhylo". Carries a "pool_sizes" attribute giving the number of trees per analysis.

Examples

```
# phylorug ships example raw tree files in its `extdata` directory.
# Point read_trees() at one of those folders:
```

```
dir <- system.file("extdata", "beetles_70p", package = "phylorug")

# With only the directory, read_trees() detects each file's format from its
# contents (NEXUS or Newick, the two formats this version supports) and
# matches the default extension filter (tre, tree, treefile, nwk, ...).

trees <- read_trees(dir)

# The result is a named list, one element per analysis, ready to pass to
# [check_taxa()] or [node_presence_matrix()]:

names(trees)

# If a folder holds many files and you want only some, narrow `ext`
# to one extension, or to a set of them:
trees <- read_trees(dir, ext = "tre")
trees <- read_trees(dir, ext = c("tre", "tree", "treefile"))

# For full control you can also force a single parser. Note this applies
# one format to every matched file, so a file of a different format would
# not be read.
trees <- read_trees(dir, format = "newick")
```

`sample_trees` *Sample beetle phylogenies (15-taxon subset)*

Description

A named list (the same structure returned by `read_trees()`) of five phylogenetic trees from a dung beetle phylogenomic study, Montanaro, Lopes et al. (2026), pruned to 15 taxa for compact demonstration of the `phylorug` workflow. The trees are already rooted, outgroup-removed, and tip labels translated from museum codes to species names. Users can select a backbone tree and comparison trees, then pass them directly to `check_taxa()`, `node_presence_matrix()` and `plot_phylorug()`.

Usage

```
sample_trees
```

Format

A named list of 5 objects of class `"phylo"`, each with 15 tips. Node labels contain support values: SH-aLRT/UFBoot2 for IQ-TREE trees and local posterior probability for ASTRAL trees.

Details

The five trees represent independent phylogenomic analyses of the same set of taxa using different inference methods and data types:

70p_uce IQ-TREE maximum likelihood analysis of ultraconserved element (UCE) data. Recommended as the backbone tree.

70p_partition_entropy IQ-TREE maximum likelihood analysis of partitioned sequence data.

70p_ghost IQ-TREE GHOST heterotachous model on partitioned data.

70p_ASTRAL_uce ASTRAL coalescent analysis of UCE gene trees.

70p_ASTRAL_partition_entropy ASTRAL coalescent analysis of partitioned gene trees.

Source

Montanaro, G., Lopes, F., Gunter, N.L., et al. (2026). Phylogenomics resolves a 200-year-old puzzle: a revised tribal classification of Afro-Eurasian dung beetles (Coleoptera: Scarabaeinae). *bioRxiv*. doi:10.64898/2026.07.22.740134

Examples

```
# Core pipeline:
backbone <- sample_trees[["70p_uce"]]
others   <- sample_trees[names(sample_trees) != "70p_uce"]

# Optional: diagnose taxon overlap
check_taxa(backbone, others)

npm <- node_presence_matrix(backbone, others, support_col = c(1, 2))
tmp <- tempfile(fileext = ".pdf")
plot_phylorug(backbone, npm, file = tmp)
unlink(tmp)
```

---

translate_tips *Relabel tips across a list of trees using a lookup table*

Description

Translates tip labels of each tree in a list using a lookup table supplied as a data frame. This function is optional in the phylorug workflow, use it only if your tip labels are specimen codes, accession numbers, or any other identifiers that need converting to a different format, such as scientific species names.

If your tip labels are already in the correct format, skip this step and proceed directly to `node_presence_matrix()`. Only tips with a matching entry in `from_col` are replaced with the corresponding value in `to_col`. Unmatched tips are left unchanged. This function modifies only the `tip.label` field of each tree. Tree topology, branch lengths, and node labels are unaffected.

Usage

```
translate_tips(trees, data, from_col, to_col, verbose = TRUE)
```

Arguments

trees	A named list of "phylo" and/or "multiPhylo" objects, one element per analysis, as returned by read_trees(). A "multiPhylo" element is a pool of tied-optimal trees (POY/TNT/PAUP*) treated as a single analysis.
data	A data frame containing the label translation lookup table. Can be a standard data.frame or a tibble.
from_col	A character string naming the column in data that holds the current tip labels (for example "specimen_code"). Required.
to_col	A character string naming the column in data that holds the replacement labels (for example "scientific_name"). Required.
verbose	Logical. If TRUE (default), reports how many tip labels were translated and how many were left unchanged for each tree.

Value

A named list of "phylo" objects identical in structure to trees, with matching tip labels replaced according to data. Tree topology and edge lengths remain unchanged.

Examples

```
# Example trees and a lookup table ship inside phylorug, so we first read
# them into the environment, then tell translate_tips() which column holds
# the current labels and which holds the replacements:
dir  <- system.file("extdata", "beetles_50p", package = "phylorug")
file <- system.file("extdata", "beetles_50p", "biogeo.csv",
                    package = "phylorug")

trees <- read_trees(dir)
dict  <- utils::read.csv(file)

# Relabel tips from specimen codes to species names. The message reports how
# many tips were translated and how many were left unchanged in each tree:
translated <- translate_tips(trees, dict,
                             from_col = "specimen_code",
                             to_col   = "species_name")
```

Index

∗ **datasets**
sample_trees, 16

add_tree, 2
add_tree(), *7*, *12*
ape::plot.phylo(), *11*

check_taxa, 4
check_taxa(), *4*, *6*, *7*, *12–14*, *16*

node_presence_matrix, 6
node_presence_matrix(), *2–6*, *9*, *12*, *13*, *15–17*

plot_phylorug, 8
plot_phylorug(), *3*, *4*, *6–8*, *16*
prune_to_shared, 13
prune_to_shared(), *3*, *5*, *7*

read_trees, 14
read_trees(), *5*, *6*, *13*, *15*, *16*, *18*

sample_trees, 16

translate_tips, 17
translate_tips(), *5*