Package: rtrees (via r-universe)

June 11, 2026

Title Deriving Phylogenies from Synthesis Trees

Version 2.0.2

Description Provides tools to derive species-level phylogenies from large synthesis mega-trees for a wide range of taxonomic groups, including plants, birds, mammals, amphibians, reptiles, fish, bees, butterflies, and sharks. When a queried species is absent from the mega-tree, it is grafted onto the tree using one of two placement strategies: attachment at the basal node of the most closely related genus or family ('at_basal_node'), or random attachment below that basal node with probability proportional to branch length ('random_below_basal'). See Li (2023) <doi:10.1111/ecog.06643> for details. Multiple species from a genus not represented in the mega-tree are placed as a polytomy to preserve clade coherence. The package interfaces with the 'megatrees' data package, which bundles or downloads on demand curated mega-trees. Users can also provide their own mega-trees.

License GPL-3

Encoding UTF-8

LazyData true

Depends R (>= 3.5.0)

Imports ape, tidytree, dplyr, tibble, utils, castor, furrr, future, megatrees (>= 0.1.3), fastmatch, data.table, Rcpp

LinkingTo Rcpp

Suggests knitr, rmarkdown, testthat, piggyback, R.rsp, ggplot2

URL https://daijiang.github.io/rtrees/

VignetteBuilder knitr, R.rsp

Config/roxygen2/version 8.0.0

NeedsCompilation yes

Author Daijiang Li [aut, cre] (ORCID: <https://orcid.org/0000-0002-0925-3421>)

Maintainer Daijiang Li <daijianglee@gmail.com>

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

Date/Publication 2026-06-11 11:40:07 UTC

RemoteUrl https://github.com/cran/rtrees

RemoteRef HEAD

RemoteSha bcb28364b7380417caf6be7f3e62fbc6bda0d031

Contents

`add_root_info` *Add genus and family basal/root node information to a phylogeny*

Description

Based on the classification of tips, find where is the basal and root node for each genus and each family. Such information can be later used to graft new tips onto the phylogeny. This function can be used to process a user provided tree.

Usage

```
add_root_info(
  tree,
  classification,
  process_all_tips = TRUE,
  genus_list = NULL,
  family_list = NULL,
  show_warning = FALSE
)
```

Arguments

tree	A phylogeny with class "phylo".
classification	A data frame of 2 columns: genus, family. It should include all genus the tips of the tree belong to.
process_all_tips	Whether to find basal nodes for all tips? Default is TRUE.
genus_list	An optinoal subset list of genus to find root information.
family_list	An optinoal subset list of family to find root information. This should be for species that do not have co-genus in the tree.
show_warning	Whether to print warning information about non-monophyletic clades or not.

Value

A phylogeny with basal nodes information attached.

---

bind_tip *Bind a tip to a phylogeny*

Description

Graft a tip to a phylogeny at location specified.

Usage

```
bind_tip(
  tree = NULL,
  where,
  tip_label,
  frac = 0.5,
  new_node_above = FALSE,
  node_label = NULL,
  return_tree = TRUE,
  tree_tbl = NULL,
  node_heights = NULL,
  use_castor = TRUE,
  sequential = TRUE
)
```

Arguments

tree	A phylogeny, with class of "phylo".
where	Location where to insert the tip. It can be either tip label or node label, but must be characters. If the location does not have a name, assign it first.
tip_label	Name of the new tip inserted.

frac	The fraction of branch length, must be between 0 and 1. This only applies when location is a tip or new_node_above = TRUE. The distance from the new inserted node to the location (a node or a tip) is the branch length of the location * (1 - frac).
new_node_above	Whether to insert the new node above when the location is a node? Default is FALSE, which will attach the new tip to the location node.
node_label	Name of the new node created. This only applies when location is a tip or new_node_above = TRUE.
return_tree	Whether to return a phylogeny with class "phylo?" Default is TRUE. Otherwise, it will return a data frame.
tree_tbl	A tibble version of the tree, optional.
node_heights	A named numeric vector of node hieghts of the tree, generated by ape::branching.times(). It is also optional if tree is specified; but required if tree_tbl is specified.
use_castor	Whether to use package castor to get the phylogeny at a node; it is faster than tidytree::offspring to figure out what are the tip offsprings at a node.
sequential	Whether to add the tip with sequential node number in the edge matrix. For example, if we want to bind a tip to a clade and the node number of the tips of this clade is from 101 to 150. We can set the node id of the new tip to 151 and push all the remaining node id to 1 after their current values. This will require us to find out the node ids of all tips that are descents of the node where we want to bind the new tip to, and it can be time costly. Yet I am still not sure whether this is necessary. Normally, the node ids of the phylo class are sequential. Therefore, the default value here is TRUE. If set to FALSE, we can just assign the id of the new tip to Ntip + 1 to save time. In addition, we probably don't need to order the node column of the edge matrix every time.

Value

Either a phylogeny or a data frame, which can be then converted to a phylogeny later.

Examples

```
tr <- ape::read.tree(text = "((A:1,B:1):1,C:2);")
tr$node.label <- c("root", "N1")
bind_tip(tr, where = "N1", tip_label = "D")
tr_tbl <- tidytree::as_tibble(tr)
node_hts <- ape::branching.times(tr)
bind_tip(tree_tbl = tr_tbl, where = "N1",
         tip_label = "D", node_heights = node_hts)
```

---

bind_tip_df *Bind a tip to a phylogeny (data frame version)*

Description

Graft a tip to a phylogeny at location specified.

Usage

```
bind_tip_df(
  tree = NULL,
  where,
  tip_label,
  frac = 0.5,
  new_node_above = FALSE,
  node_label = NULL,
  return_tree = TRUE,
  tree_tbl = NULL,
  node_heights = NULL,
  use_castor = FALSE
)
```

Arguments

tree	A phylogeny, with class of "phylo".
where	Location where to insert the tip. It can be either tip label or node label, but must be characters. If the location does not have a name, assign it first.
tip_label	Name of the new tip inserted.
frac	The fraction of branch length, must be between 0 and 1. This only applies when location is a tip or new_node_above = TRUE. The distance from the new inserted node to the location (a node or a tip) is the branch length of the location * (1 - frac).
new_node_above	Whether to insert the new node above when the location is a node? Default is FALSE, which will attach the new tip to the location node.
node_label	Name of the new node created. This only applies when location is a tip or new_node_above = TRUE.
return_tree	Whether to return a phylogeny with class "phylo?" Default is TRUE. Otherwise, it will return a data frame.
tree_tbl	A tibble version of the tree, optional.
node_heights	A named numeric vector of node hieghts of the tree, generated by ape::branching.times(). It is also optional if tree is specified; but required if tree_tbl is specified.
use_castor	Whether to use package castor to get the phylogeny at a node; it is faster than tidytree::offspring to figure out what are the tip offsprings at a node.

Value

Either a phylogeny or a data frame, which can be then converted to a phylogeny later.

Examples

```
tr <- ape::read.tree(text = "((A:1,B:1):1,C:2);")
tr$node.label <- c("root", "N1")
bind_tip_df(tr, where = "N1", tip_label = "D")
tr_tbl <- tidytree::as_tibble(tr)
```

```
node_hts <- ape::branching.times(tr)
bind_tip_df(tree_tbl = tr_tbl, where = "N1",
            tip_label = "D", node_heights = node_hts)
```

`classifications` *Classifications of species*

Description

Genus and family information of different groups of taxon.

- Plant classification information. Its sources include: + based on `V.PhyloMaker::nodes.info.1` + based on The Plant List + taxonlookup + Plants of the World online
- Fish classification information was based on FishBase. There are 4,825 genus in this file. `https://fishtreeoflife.org/downloads/PFC_taxonomy.csv.xz`
- Bee classification information was from Bee Tree of Life. Note that we used 'Subfamily' in their nomenclature file as "family" here. If a genus' Subfamily is missing, we used its Family.
- Bird classification information was based on BirdLife, which resulted in 2,391 genus. `http://datazone.birdlife.org/species/taxonomy` However, based on the taxonomy file of the Jetz et al. 2012 phylogeny, there are additional 117 genus that are not in the file of BirdLife. Both are combined here, which leads to 2,508 genus.
- Mammal classification information was based on PHYLACINE, which has 1,400 genus. `https://github.com/MegaPast2Future/PHYLACINE_1.2/blob/master/Data/Taxonomy/Synonymy_table_valid_species_only.csv` Additional genus from Vertlife were added too. For the same genus from both PHYLACINE and Vertlife that have different family information, I used the family from Vertlife as I found that they are mostly more accurate.
- Amphibian classification information was from VertLife.
- Reptile classification information was largely from wikipedia.
- Shark and Ray classification information was largely from NCBI.
- Butterfly classification information was from Kawahara et al. 2023, using the tip labels of their phylogeny.

Usage

```
classifications
```

Format

A data frame with three columns: genus, family, and taxon (`plant`, `fish`, `bird`, `mammal`, `amphibian`, `reptile`, `shark_ray`, `bee`, `butterfly`).

get_graft_status *Extract grafting status information as a data frame*

Description

Extract grafting status information as a data frame

Usage

```
get_graft_status(tree)
```

Arguments

tree	A phylogeny generated by `get_tree(...)` with trailing stars in tip labels.

Value

A tibble with three columns: tip_label, species, and status.

get_one_tree *Derive a phylogeny from a mega-tree*

Description

For a list of species, generate a phylogeny from a provided mega-tree. If a species is not in the mega-tree, it will be grafted to the mega-tree with three scenarioes.

Usage

```
get_one_tree(
  sp_list,
  tree,
  taxon,
  scenario = c("at_basal_node", "random_below_basal"),
  show_grafted = FALSE,
  tree_by_user = FALSE,
  .progress = "text",
  dt = TRUE
)
```

Arguments

`sp_list` A character vector or a data frame with at least three columns: species, genus, family. Species column holds the species for which we want to have a phylogeny. It can also have two optional columns: close_sp and close_genus. We can specify the closest species/genus of the species based on expert knowledge. If specified, the new species will be grafted to that particular location.

It can also be a string vector if `taxon` is specified. Though it probably is a better idea to prepare your data frame with `sp_list_df()`. The string vector can also have the same format as that required by Phylomatic (i.e., family/genus/genus_sp).

`tree` A mega-tree with class `phylo` or a list of mega-trees with class `multiPhylo`. Optional if `taxon` is specified, in which case, a default mega-phylogeny (or a set of 100 randomly selected posterior phylogenies) will be used (see their own documentations from the `megatrees` package).

- For amphibian, the mega-trees are loaded via `megatrees::get_tree_amphibian_n100()`.
- For bee, the mega-tree is megatrees::tree_bee, with the bootstrap option loaded via `megatrees::get_tree_bee_n100()`.
- For butterfly, the mega-tee is megatrees::tree_butterfly.
- For bird, the mega-trees are loaded via `megatrees::get_tree_bird_n100()`.
- For fish, the mega-tree is megatrees::tree_fish_12k, with the all-taxon option loaded via `megatrees::get_tree_fish_32k_n50()`.
- For mammal, the default mega-trees are loaded via `megatrees::get_tree_mammal_n100_vertlife` with `megatrees::get_tree_mammal_n100_phylacine()` be the other option.
- For plant, the default mega-tree is megatrees::tree_plant_otl (Smith and Brown 2018); megatrees::tree_plant_Carruthers (Carruthers et al. 2026) is also available; 100 posterior trees are loaded via `megatrees::get_tree_plant_n100_Carruthers(`
- For reptile, the mega-trees are loaded via `megatrees::get_tree_reptile_n100()`.
- For shark, ray, and chimaeras, the mega-trees are loaded via `megatrees::get_tree_shark_ray_n10`

`taxon` The taxon of species in the `sp_list`. Currently, can be `amphibian`, `bird`, `fish`, `mammal`, `plant`, `reptile`, or `shark_ray`.

`scenario` How to insert a species into the mega-tree?

- In both scenarioes, if there is only 1 species in the genus or family, a new node will be inserted to the middle point of this only species' branch length and the new species will be attached to this new node.
- If `scenario = "at_basal_node"`, a species is attached to the basal node of the same genus or the same family if the mega-tree does not have any species of this genus.
- If `scenario = "random_below_basal"`, a species is attached to a randomly selected node that is at or below the basal node of the same genus of the same family if the mega-tree does not have any species in this genus. The probability of node been selected is proportional to its branch length. Because of the random sampling involved, you may want to run several times to get a collection of derived phylogenies.

show_grafted	Whether to indicate which species was grafted onto the mega-tree. If TRUE, a * will be appended to the species name on the tip if it was grafted within the same genus; ** will be appended if it was grafted within the same family.
tree_by_user	Is the mega-tree provided by user? Default is FALSE but it will be automatically set to TRUE when the class of tree is multiPhylo since we don't provide any such mega-trees here.
.progress	Form of progress bar, default to be text.
dt	Whether to use data.table version to bind tips bind_tip. The default is TRUE as it maybe slightly faster.

Value

A phylogeny for the species required, with class phylo.

---

get_tree *Get one or multiple trees from megatree(s)*

Description

For some taxa groups, there are multiple posterior megatrees. It is a common task to derive a phylogeny from each of these (or a random subset of) megatrees.

Usage

```
get_tree(
  sp_list,
  tree,
  taxon = NULL,
  scenario = c("at_basal_node", "random_below_basal"),
  show_grafted = FALSE,
  tree_by_user = FALSE,
  mc_cores = future::availableCores() - 2,
  .progress = "text",
  fish_tree = c("timetree", "all-taxon"),
  mammal_tree = c("vertlife", "phylacine"),
  bee_tree = c("maximum-likelihood", "bootstrap"),
 plant_tree = c("tree_plant_otl", "tree_plant_Carruthers", "tree_plant_n100_Carruthers"),
  dt = TRUE
)
```

Arguments

sp_list	A character vector or a data frame with at least three columns: species, genus, family. Species column holds the species for which we want to have a phylogeny. It can also have two optional columns: close_sp and close_genus. We can specify the closest species/genus of the species based on expert knowledge. If specified, the new species will be grafted to that particular location.

It can also be a string vector if `taxon` is specified. Though it probably is a better idea to prepare your data frame with `sp_list_df()`. The string vector can also have the same format as that required by Phylomatic (i.e., family/genus/genus_sp).

`tree` A mega-tree with class `phylo` or a list of mega-trees with class `multiPhylo`. Optional if `taxon` is specified, in which case, a default mega-phylogeny (or a set of 100 randomly selected posterior phylogenies) will be used (see their own documentations from the `megatrees` package).

- For amphibian, the mega-trees are loaded via `megatrees::get_tree_amphibian_n100()`.
- For bee, the mega-tree is megatrees::tree_bee, with the bootstrap option loaded via `megatrees::get_tree_bee_n100()`.
- For butterfly, the mega-tee is megatrees::tree_butterfly.
- For bird, the mega-trees are loaded via `megatrees::get_tree_bird_n100()`.
- For fish, the mega-tree is megatrees::tree_fish_12k, with the all-taxon option loaded via `megatrees::get_tree_fish_32k_n50()`.
- For mammal, the default mega-trees are loaded via `megatrees::get_tree_mammal_n100_vertlife` with `megatrees::get_tree_mammal_n100_phylacine()` be the other option.
- For plant, the default mega-tree is megatrees::tree_plant_otl (Smith and Brown 2018); megatrees::tree_plant_Carruthers (Carruthers et al. 2026) is also available; 100 posterior trees are loaded via `megatrees::get_tree_plant_n100_Carruthers(`
- For reptile, the mega-trees are loaded via `megatrees::get_tree_reptile_n100()`.
- For shark, ray, and chimaeras, the mega-trees are loaded via `megatrees::get_tree_shark_ray_n10`

`taxon` The taxon of species in the `sp_list`. Currently, can be `amphibian`, `bird`, `fish`, `mammal`, `plant`, `reptile`, or `shark_ray`.

`scenario` How to insert a species into the mega-tree?

- In both scenarioes, if there is only 1 species in the genus or family, a new node will be inserted to the middle point of this only species' branch length and the new species will be attached to this new node.
- If `scenario = "at_basal_node"`, a species is attached to the basal node of the same genus or the same family if the mega-tree does not have any species of this genus.
- If `scenario = "random_below_basal"`, a species is attached to a randomly selected node that is at or below the basal node of the same genus of the same family if the mega-tree does not have any species in this genus. The probability of node been selected is proportional to its branch length. Because of the random sampling involved, you may want to run several times to get a collection of derived phylogenies.

`show_grafted` Whether to indicate which species was grafted onto the mega-tree. If `TRUE`, a `*` will be appended to the species name on the tip if it was grafted within the same genus; `**` will be appended if it was grafted within the same family.

`tree_by_user` Is the mega-tree provided by user? Default is `FALSE` but it will be automatically set to `TRUE` when the class of `tree` is `multiPhylo` since we don't provide any such mega-trees here.

mc_cores	Number of cores to parallel processing when tree is a list of large number of trees. The default is the number of available cores minus 2.
.progress	Form of progress bar, default to be text.
fish_tree	Which fish tree do you want to use? If it is "timetree" (default), it will be the smaller time tree with 11638 species that all have sequence data; if it is "all-taxon", then it will be the 100 larger posterior phylogenies with 31516 soecues.
mammal_tree	Which set of mammal trees to use? If it is "vertlife" (default), then 100 randomly selected posterior phylogenies provided by Vertlife will be used; if it is "phylacine", then 100 randomly selected posterior phylogenies provided by PHYLACINE will be used.
bee_tree	Which bee tree to use? If it is "maximum-likelihood" (default), the a single maximum likelihood tree will be used. If it is "bootstrap", then a set of 100 randomly selected posterior phylogenies will be used. All trees are provided by the Bee Tree of Life.
plant_tree	Which plant tree to use? If "tree_plant_otl" (default), the Smith and Brown (2018) tree is used. If "tree_plant_Carruthers", the single best-scoring tree from Carruthers et al. (2026) is used. If "tree_plant_n100_Carruthers", 100 randomly selected posterior phylogenies from Carruthers et al. (2026) are downloaded and used (large file, ~135 MB, cached after first use).
dt	Whether to use data.table version to bind tips bind_tip. The default is TRUE as it maybe slightly faster.

Details

Derive a phylogeny from a mega-tree

For a list of species, generate a phylogeny or multiple phylogenies from a provided mega-tree or mega-trees. If a species is not in the mega-tree, it will be grafted to the mega-tree with two scenarios.

Value

A phylogeny for the species required, with class phylo; or a list of phylogenies with class multiPhylo depends on the input tree. Within each phylogeny, the grafted status of all species was saved as a data frame named as "graft_status".

Examples

```
test_sp <- c(
  "Serrasalmus_geryi", "Careproctus_reinhardti", "Gobiomorphus_coxii",
  "Periophthalmus_barbarus", "Prognichthys_glaphyrae", "Barathronus_bicolor",
  "Knipowitschia_croatica", "Rhamphochromis_lucius", "Neolissochilus_tweediei",
  "Haplochromis_nyanzae", "Astronesthes_micropogon", "Sanopus_reticulatus"
)
test_tree <- get_tree(
  sp_list = test_sp,
  taxon = "fish",
  show_grafted = TRUE
)
```

rm_stars *Remove trailing **

Description

Remove trailing *

Usage

```
rm_stars(tree)
```

Arguments

tree	A phylogeny generated by `get_tree(..., show_grafted = TRUE)` with trailing stars in tip labels.

Value

A phylogeny after removing trailing stars.

sp_list_df *Convert a vector of species names to a data frame*

Description

Convert a vector of species names to a data frame

Usage

```
sp_list_df(sp_list, taxon)
```

Arguments

sp_list	A string vector or a data frame with at least one column named "species".
taxon	The taxon group of this species list. If not specified, only species and genus will be returned.

Value

A data frame with columns: species, genus, and family (if `taxon` is specified).

Examples

```
sp_list_df(
  sp_list = c("Serrasalmus_geryi", "Careproctus_reinhardti", "Gobiomorphus_coxii"),
  taxon = "fish"
)
```

`taxa_supported` *Taxonomic groups supported*

Description

Supported taxonomic groups with mega-trees provided in the **megatrees** package.

Usage

```
taxa_supported
```

Format

An object of class `character` of length 9.

Index