Package ‘Rogue’

January 13, 2022

Title  Identify Rogue Taxa in Sets of Phylogenetic Trees
Version  2.1.0
License  GPL (>= 3)
Description  Rogue ("wildcard") taxa are leaves with uncertain phylogenetic position.
Their position may vary from tree to tree under inference methods that yield a tree set (e.g. bootstrapping, Bayesian tree searches, maximum parsimony).
The presence of rogue taxa in a tree set can potentially remove all information from a consensus tree. The information content of a consensus tree - a function of its resolution and branch support values - can often be increased by removing rogue taxa.

URL  https://github.com/ms609/Rogue/,
     https://github.com/aberer/RogueNaRok/,
     https://github.com/ms609/RogueNaRok/

BugReports  https://github.com/ms609/Rogue/issues/

Depends  R (>= 3.5.0)
Imports  ape (>= 5.0), cli (>= 3.0), fastmatch, grDevices, matrixStats, Rdpack (>= 0.7), Rfast, stats, TreeDist (>= 2.2.0), TreeTools (>= 1.6.0), utils,
Suggests  knitr, rmarkdown, spelling, testthat,
Config/Needs/github-actions  callr, pkgbuild, rcmdcheck,
Config/Needs/coverage  covr
Config/Needs/memcheck  devtools
Config/Needs/metadata  codemeta
Config/Needs/website  pkgdown
RdMacros  Rdpack
Description

RogueTaxa() finds wildcard leaves whose removal increases the resolution or branch support values of a consensus tree, using the relative bipartition, shared phylogenetic, or mutual clustering concepts of information.

Usage

RogueTaxa(
  trees,
  info = c("spic", "scic", "fspic", "fscic", "rbic"),
  return = c("taxa", "tree"),
  bestTree = NULL,
  computeSupport = TRUE,
  dropsetSize = 1,
  neverDrop = character(0),
  labelPenalty = 0,
  mreOptimization = FALSE,
  threshold = 50,
)
RogueTaxa

verbose = FALSE

QuickRogue(
trees,
info = "phylogenetic",
p = 0.5,
log = TRUE,
average = "median",
device = "mad",
neverDrop,
fullSeq = FALSE
)

C_RogueNaRok(
bootTrees = "",
runId = "tmp",
treeFile = ",
computeSupport = TRUE,
dropsetSize = 1,
excludeFile = "",
workDir = ",
labelPenalty = 0,
mreOptimization = FALSE,
threshold = 50
)

Arguments

trees List of trees to analyse.
info Concept of information to employ; see details.
return If taxa, returns the leaves identified as rogues; if tree, return a consensus tree omitting rogue taxa.
computeSupport Logical: If FALSE, then instead of trying to maximize the support in the consensus tree, RogueNaRok will try to maximize the number of bipartitions in the final tree by pruning taxa.
dropsetSize Integer specifying maximum size of dropset per iteration. If dropsetSize == n, then RogueNaRok will test in each iteration which tuple of n taxa increases the optimality criterion the most, pruning taxa accordingly. This improves the result, but run times will increase at least linearly.
neverDrop Tip labels that should not be dropped from the consensus.
labelPenalty A weight factor to penalize for dropset size when info = 'rbic'. The higher the value, the more conservative the algorithm is in pruning taxa. The default value of 0 gives the RBIC; 1 gives Pattengale's criterion.
threshold, mreOptimization A threshold or mode for the consensus tree that is optimized. Specify a value between 50 (majority rule consensus, the default) and 100 (strict consensus), or set
mreOptimization = TRUE for the extended majority rule consensus. Note that rogue taxa identified with respect to different thresholds can vary substantially.

verbose Logical specifying whether to display output from RogueNaRok. If FALSE, output will be included as an attribute of the return value.

p Proportion of trees that must contain a split before it is included in the consensus under consideration. 0.5, the default, corresponds to a majority rule tree; 1.0 will maximize the information content of the strict consensus.

log Logical specifying whether to log-transform distances when calculating leaf stability.

average Character specifying whether to use 'mean' or 'median' tip distances to calculate leaf stability.

deviation Character specifying whether to use 'sd' or 'mad' to calculate leaf stability.

fullSeq Logical specifying whether to list all taxa (TRUE), or only those that improve information content when all are dropped (FALSE).

bootTrees Path to a file containing a collection of bootstrap trees.

runId An identifier for this run, appended to output files.

treeFile, bestTree If a single best-known tree (such as an ML or MP tree) is provided, RogueNaRok optimizes the bootstrap support in this best-known tree (still drawn from the bootstrap trees); the threshold parameter is ignored.

excludeFile Taxa in this file (one taxon per line) will not be considered for pruning.

workDir Path to a working directory where output files are created.

Details

"Rogue" or (loosely) "wildcard" taxa (Nixon and Wheeler 1992) are leaves whose position in a tree is poorly constrained, typically because much of the phylogenetic data associated with the taxon is either missing or in conflict with other data (Kearney 2002).

These functions use heuristic methods to identify rogue taxa whose removal improves the information content of a consensus tree, by the definitions of information discussed below.

Value

RogueTaxa() returns a data.frame. Each row after the first, which describes the starting tree, describes a dropset operation. Columns describe:

- num: Sequential index of the drop operation
- taxNum: Numeric identifier of the dropped leaves
- taxon: Text identifier of dropped leaves
- rawImprovement: Improvement in score obtained by this operation
- IC: Information content of tree after dropping all leaves so far, by the measure indicated by info.

C_RogueNaRok() returns 0 if successful; -1 on error.
Functions

- QuickRogue: Shortcut to 'fast' heuristic, with option to return evaluation of all taxa using fullSeq = TRUE.

Information criteria

The splitwise phylogenetic information content measure produces the best results (Smith 2022). It uses the splitwise information content as a shortcut, which involves double counting of some information (which may or may not be desirable). The same holds for the mutual clustering information measure; this measure is less obviously suited to the detection of rogues. This measure interprets split frequency as a proxy for the probability that a split is true, which is a valid interpretation of a Bayesian posterior sample (Holder et al. 2008), a reasonable but imperfect interpretation of a bootstrap sample (Berry and Gascuel 1996), and a bad interpretation of a sample of most parsimonious trees.

The "relative bipartition information criterion" (RBIC) is the sum of all support values divided by the maximum possible support in a fully bifurcating tree with the initial set of taxa. The relative bipartition information content approach employs the 'RogueNaRok' implementation (Aberer et al. 2013), which can handle large trees relatively quickly. The RBIC is is not strictly a measure of information and can produce undesirable results (Wilkinson and Crotti 2017).

C_RogueNaRok() directly interfaces the 'RogueNaRok' C implementation, with no input checking; be aware that invalid input will cause undefined behaviour and is likely to crash R.

Author(s)

Martin R. Smith (martin.smith@durham.ac.uk), linking to RogueNaRok C library by Andre Aberer (<andre.aberer at googlemail.com>)

References


Examples

library("TreeTools", warn.conflicts = FALSE)

trees <- list(read.tree(text = "(a, (b, (c, (d, (e, (X1, X2))))));"),
               read.tree(text = "((a, (X1, X2)), (b, (c, (d, e))));"))
RogueTaxa(trees, dropsetSize = 2)

trees <- list(
    read.tree(text = "((a, y), (b, (c, (z, ((d, e), (f, (g, x)))))));",
    read.tree(text = "(a, (b, (c, (z, (((d, y), e), (f, (g, x))))));'",
    read.tree(text = "(a, (b, ((c, z), ((d, (e, y)), ((f, x), g))));'",
    read.tree(text = "(a, ((b, x), ((c, z), ((d, e), (f, (g, y))))));'")
)
cons <- consensus(trees, p = 0.5)
plot(cons)
LabelSplits(cons, SplitFrequency(cons, trees) / length(trees))
reduced <- RogueTaxa(trees, info = 'phylogenetic', ret = 'tree')
plot(reduced)
LabelSplits(reduced, SplitFrequency(reduced, trees) / length(trees))

QuickRogue(trees, fullSeq = TRUE)

bootTrees <- system.file('example/150.bs', package = 'Rogue')
tmpDir <- tempdir()
XX <- capture.output( # Don't print verbose run details to console
    C_RogueNaRok(bootTrees, workDir = tmpDir)
)

# Results have been written to our temporary directory
oldwd <- setwd(tmpDir)
head(read.table('RogueNaRok_droppedRogues.tmp', header = TRUE))

# Delete temporary files
file.remove('RogueNaRok_droppedRogues.tmp')
file.remove('RogueNaRok_info.tmp')

setwd(oldwd)
Description

TipInstability() calculates the instability of each leaf in a tree. Unstable leaves are likely to display roguish behaviour.

Usage

TipInstability(
  trees,
  log = TRUE,
  average = "mean",
  deviation = "sd",
  checkTips = TRUE
)

ColByStability(trees, log = TRUE, average = "mean", deviation = "sd")

Arguments

trees List of trees to analyse.
log Logical specifying whether to log-transform distances when calculating leaf stability.
average Character specifying whether to use 'mean' or 'median' tip distances to calculate leaf stability.
deviation Character specifying whether to use 'sd' or 'mad' to calculate leaf stability.
checkTips Logical specifying whether to check that tips are numbered consistently.

Details

Smith (2022) defines the instability of a pair of leaves as the median absolute divergence in the graph geodesic (the number of edges in the shortest path between the leaves) across all trees, normalized against the mean graph geodesic. The instability of a single leaf is the mean instability of all pairs that include that leaf; higher values characterise leaves whose position is more variable between trees.

Other concepts of leaf instability include

- The 'taxonomic instability index', as implemented in Mesquite: described by Thomson and Shaffer (2010) as \[ \sum_{(x,y), j \neq i} \frac{|D_{ijx} - D_{ijy}|}{(D_{ijx} - D_{ijy})^2}, \] where \( D_{ijx} \) is the patristic distance (i.e. length of edges) between leaves \( i \) and \( j \) in tree \( x \).
- the average stability of triplets (i.e. quartets including the root) that include the leaf (Thorley and Wilkinson 1999), implemented in "Phyutility" (Smith and Dunn 2008); and related to 'positional congruence' measures (Estabrook 1992; Pol and Escapa 2009).

Author(s)

Martin R. Smith (martin.smith@durham.ac.uk)
TipVolatility

References


See Also

Other tip instability functions: TipVolatility()

Examples

library("TreeTools", quietly = TRUE)
trees <- AddTipEverywhere(BalancedTree(8), 'Rogue')[3:6]
plot(consensus(trees), tip.col = ColByStability(trees))
instab <- TipInstability(trees, log = FALSE, ave = 'mean', dev = 'mad')
plot(ConsensusWithout(trees, names(instab[instab > 0.2])))

TipVolatility Detect rogue taxa using phylogenetic information distance

Description

Calculate the volatility of each tip: namely, the impact on the mean phylogenetic information distance (Smith 2020) between trees when that tip is removed. Effective when the number of trees is small.

Usage

TipVolatility(trees)
**TipVolatility**

**Arguments**

- `trees`: List of trees to analyse.

**Value**

`TipVolatility()` returns a named vector listing the volatility index calculated for each leaf.

**References**


**See Also**

Other tip instability functions: `TipInstability()`

**Examples**

```r
library("TreeTools", quietly = TRUE)
trees <- AddTipEverywhere(BalancedTree(8), "Rogue")
trees[] <- lapply(trees, AddTip, "Rogue", "Rogue"

sb <- TipVolatility(trees)
sbNorm <- 1 + (99 * (sb - min(sb)) / (max(sb - min(sb)))
col <- hcl.colors(128, "inferno")[sbNorm]
plot(consensus(trees), tip.color = col)
plot(ConsensusWithout(trees, names(sb[sb == max(sb)])))
```
Index

* tip instability functions
  TipInstability, 6
  TipVolatility, 8

C_RogueNaRok (RogueTaxa), 2
ColByStability (TipInstability), 6

QuickRogue (RogueTaxa), 2

RogueTaxa, 2

TipInstability, 6, 9
TipVolatility, 8, 8