Package 'STraTUS'

May 7, 2026

Title Enumeration and Uniform Sampling of Transmission Trees for a Known Phylogeny

Version 1.1.2

Author Matthew Hall [aut, cre], Caroline Colijn [ctb]

Maintainer Matthew Hall <matthew.hall@bdi.ox.ac.uk>

Description For a single, known pathogen phylogeny, provides functions for enumeration of the set of compatible epidemic transmission trees, and for uniform sampling from that set. Optional arguments allow for incomplete sampling with a known number of missing individuals, multiple sampling, and known infection time limits. Always assumed are a complete transmission bottleneck and no superinfection or reinfection. See Hall and Colijn (2019) <doi:10.1093/molbev/msz058> for methodology.

Depends R (>= 3.4)

Imports ape, phangorn, igraph, gmp, ggplot2, ggtree (>= 2.0.0), RcppAlgos, stats

License GPL

URL http://github.com/mdhall272/STraTUS/

Encoding UTF-8

RoxygenNote 7.1.0

LazyData TRUE

NeedsCompilation no

Repository CRAN

Date/Publication 2020-04-04 12:20:06 UTC

Contents

`build.edgelist` *For a sample, produce the transmission tree as a* `igraph` *object*

Description

For a sample, produce the transmission tree as a `igraph` object

Usage

```
build.edgelist(generator, sample)
```

Arguments

`generator`	A list of class `tt.generator` produced by `tt.generator`.
`sample`	A list of class `tt` produced by `sample.tt` or `sample.partial.tt`

Value

An `igraph` object

Examples

```
generator <- tt.generator(stratus.example.tree)
samples <- sample.tt(generator, 1)
build.edgelist(generator, samples[[1]])
```

`draw.fully.sampled` *For a sample with no unsampled hosts, draw the annotated phylogeny using* `ggtree`

Description

For a sample with no unsampled hosts, draw the annotated phylogeny using `ggtree`

Usage

```
draw.fully.sampled(generator, sample)
```

Arguments

`generator`	A list of class `tt.generator` produced by `tt.generator`.
`sample`	A list of class `tt` produced by `sample.tt` or `sample.partial.tt`

Value

A ggtree object

Examples

```
generator <- tt.generator(stratus.example.tree)
samples <- sample.tt(generator, 1)
draw.fully.sampled(generator, samples[[1]])
```

---

draw.incompletely.sampled

For a sample with or without unsampled hosts, draw the annotated phylogeny using ggtree

---

Description

For a sample with or without unsampled hosts, draw the annotated phylogeny using ggtree

Usage

```
draw.incompletely.sampled(generator, sample)
```

Arguments

generator	A list of class tt.generator produced by tt.generator.
sample	A list of class tt produced by sample.tt or sample.partial.tt

Value

A ggtree object

Examples

```
generator <- tt.generator(stratus.example.tree, max.unsampled = 2)
samples <- sample.tt(generator, 1, unsampled=2)
# Tree is annotated with the number of unsampled hosts along each branch
draw.incompletely.sampled(generator, samples[[1]])
# This still works if there are no unsampled hosts
samples <- sample.tt(generator, 1)
draw.incompletely.sampled(generator, samples[[1]])
```

grouping.map	*A vector assigning the tips of* stratus.example.tree *to groups (in the order they appear in* stratus.example.tree$tip.label*), as an example of multiple sampling.*

Description

A vector assigning the tips of stratus.example.tree to groups (in the order they appear in stratus.example.tree$tip.label), as an example of multiple sampling.

Format

A character vector of length 20

sample.partial.tt	*Resample the subtree rooted at any tree node, keeping the annotations for the rest of the tree fixed*

Description

Resample the subtree rooted at any tree node, keeping the annotations for the rest of the tree fixed

Usage

```
sample.partial.tt(
  generator,
  count = 1,
  unsampled = 0,
  starting.node = phangorn::getRoot(generator$tree),
  existing = NULL,
  check.integrity = TRUE,
  draw = count == 1,
  igraph = FALSE,
  verbose = FALSE
)
```

Arguments

generator	A list of class tt.generator produced by tt.generator.
count	How many transmission trees to sample.
unsampled	The number of unsampled hosts in the transmission chain. (The whole transmission chain, even if only part of the transmission tree is being resampled). A value >0 requires a generator list whose type is unsampled.

starting.node	The root of the subtree to resample. If this is the root of the whole tree, then existing is irrelevent (but generally sample.tt should be used for this purpose).
existing	An existing list of class tt, representing a transmission tree to be modified. Usually these are produced by a sample.tt or sample.partial.tt call.
check.integrity	Whether to check if existing is indeed a valid transmission tree.
draw	Use ggtree to draw a coloured phylogeny showing each transmission tree overload onto the phylogeny
igraph	Produce the transmission trees in igraph format.
verbose	Verbose output

Value

A list, each of whose elements is a list of class tt with one or more of the following elements:

- annotations Always present. A vector indicating which host (given by numbers corresponding to the ordering in generator$hosts) is assigned to each phylogeny node.
- edgelist Always present. A data.frame giving the edge list; the first column are parents and the second children.
- hidden Present if unsampled is greater than 0. The number of "hidden" unsampled hosts (with no associated nodes) along each branch.
- picture Present if draw was TRUE; a ggtree object.
- igraph Present if igraph was TRUE; an igraph object.

Examples

```
# draw one sample from the uniform distribution
generator <- tt.generator(stratus.example.tree)
samples <- sample.tt(generator, 1, draw = TRUE)
original.tt <- samples[[1]]
# sample anew, from node 31 downwards
revised.tt <- sample.partial.tt(generator, 1, starting.node = 31,
    existing = original.tt, draw = TRUE)
```

---

sample.tt *Sample one or more transmission trees uniformly*

Description

Sample one or more transmission trees uniformly

Usage

```
sample.tt(
  generator,
  count = 1,
  unsampled = 0,
  draw = count == 1,
  igraph = FALSE,
  verbose = FALSE
)
```

Arguments

generator	A list of class tt.generator produced by tt.generator.
count	How many transmission trees to sample.
unsampled	The number of unsampled hosts in the transmission chain.
draw	Use ggtree to draw a coloured phylogeny showing each transmission tree overlaid onto the phylogeny.
igraph	Produce the transmission trees in igraph format.
verbose	Verbose output

Value

A list, each of whose elements is a list of class tt with one or more of the following elements:

- annotations Always present. A vector indicating which host (given by numbers corresponding to the ordering in generator$hosts) is assigned to each phylogeny node.
- edgelist Always present. A data.frame giving the edge list; the first column are parents and the second children.
- hidden Present if unsampled is greater than 0. The number of "hidden" unsampled hosts (with no associated nodes) along each branch.
- picture Present if draw was TRUE; a ggtree object.
- igraph Present if igraph was TRUE; an igraph object.

Examples

```
# draw one sample from the uniform distribution
generator <- tt.generator(stratus.example.tree)
samples <- sample.tt(generator, 1, draw = TRUE)
samples[[1]]
# with unsampled.hosts
generator.us <- tt.generator(stratus.example.tree, max.unsampled = 2)
# note that you can ask for less unsampled hosts than the generator has (but not more)
samples.1us <- sample.tt(generator.us, 1, unsampled = 1,  draw = TRUE)
samples.1us[[1]]
# with multiply sampled hosts
generator.ms <- tt.generator(stratus.example.tree, tip.map = grouping.map)
samples.ms <- sample.tt(generator.ms, 1, draw = TRUE)
```

stratus.example.tree *An unexceptional phylogeny generated with* rtree *from* ape

Description

An unexceptional phylogeny generated with rtree from ape

Format

A phylogenetic tree (phylo format) with 20 tips and 19 internal nodes

tt.generator *Enumerate transmission trees for the given pathogen phylogeny, and provide a uniform sample generator*

Description

This function produces a list of class tt.generator which can be used to randomly sample transmission trees for the input phylogeny, and contains information on the number of compatible transmission trees.

Usage

```
tt.generator(
  tree,
  max.unsampled = 0,
  max.infection.to.sampling = Inf,
  max.sampling.to.noninfectious = Inf,
  minimum.heights = NULL,
  maximum.heights = NULL,
  tip.map = tree$tip.label,
  bigz = FALSE
)
```

Arguments

tree	A phylo object
max.unsampled	The maximum number of unsampled hosts in the transmission chain. The default is 0.
max.infection.to.sampling	The greatest time period (in tree branch length units) that can have elapsed between the infection of a host and a tip from that host appearing. The default is infinity, meaning that no such time limit exists.

`max.sampling.to.noninfectious`
: The greatest time period (in tree branch length units) that can have elapsed between a tip from a host appearing and that host becoming noninfectious. If this is 0, a host's infection ends at the time of its last tip. The default is infinity, meaning that no such time limit exists.

`minimum.heights`
: A vector of the same length as the set of sampled hosts (at present this is always the number of tips of the tree) dictating the minimum height at which nodes can be allocated to each host. The order is the same as the order of tips in `tree$tip.label`. If absent, no such restrictions will be placed. Each must be equal to or smaller than the height of the last tip from the corresponding host. This overrides the given value of `max.sampling.to.noninfectious`.

`maximum.heights`
: A vector of the same length as the set of sampled hosts (at present this is always the number of tips of the tree) dictating the maximum height at which nodes can be allocated to each host. The order is the same as the order of tips in `tree$tip.label`. If absent, no such restrictions will be placed. Each must be equal to or greater than the height of the last tip from the corresponding host. This overrides the given value of `max.infection.to.sampling`.

`tip.map`
: A vector of the same length as the tip set of the tree listing a string giving the host from which the corresponding sample was derived. If absent, each tip is assumed to come from a different host and the tip names are taken to be the host names.

`bigz`
: Use `bigz` from `gmp` for integers, recommended for large trees

Value

A list of class `tt.info` with the following fields:

- `tree` The input tree
- `tt.count`The total number of possible transmission trees.
- `hosts` The vector of host names. The order of the elements of this vector is used in the output of `sample.tt`.
- `height.limits`A matrix giving maximum and minimum node heights, in two columns. Rows are ordered by the order of hosts given in the `host` field.
- `bridge`A vector with the same length as the node set of the tree, dictating which nodes have their annotation forced by the tip annotations. Entries are host numbers for nodes whose annotation must be that host, and NA for nodes which can take multiple hosts.
- `node.calculations` A list with the same length as the number of nodes of the tree and whose entries are indexed in the same order. If `max.unsampled` is 0, each has the following fields (the terminology here comes from the Hall paper):
  - `p` The number of valid partitions of the subtree rooted at this node.
  - `pstar` The number of valid partitions of the unrooted tree obtained by attaching a single extra tip to the root node of the subtree rooted at this node. Alternatively, if any height constraints are given, a vector of the same length as the set of hosts, giving the number of partitions of the unrooted tree if the extra partition element is subject to the same minimum (but not maximum) height constraint as each host in turn.

- v A list indexed by the set of hosts, whose entries are the number of valid partitions of the subtree rooted at this node where the root node is in the partition element from each host.

Alternatively, if max.unsampled is greater than 0, the entries are:

- p A vector of length 1 + max.unsampled giving the number of valid partitions of the subtree rooted at this node if there are between 0 and max.unsampled (in order) partition elements containing no tips.
- pstar A vector of length 1 + max.unsampled giving the number of valid partitions of the tree obtained from the subtree rooted at this node by adding an extra tip connected to the root node, if there are between 0 and max.unsampled (in order) partition elements containing no tips.
- ps As with p, except this counts only partitions that have the root node in a sampled component (one containing at least one tip).
- pu As with p, except this counts only partitions that have the have the root node in an unsampled component (one containing no tip).
- v A list indexed by the set of hosts and "unsampled", whose entries are, for each host and an unsampled host, a vector of length 1 + max.unsampled counting the number of partitions that have the root node in that host's component if there are between 0 and max.unsampled partition elements containing no tips.

Examples

```
# make a generator for the example tree
generator <- tt.generator(stratus.example.tree)
# count the total number of transmission trees
generator$tt.count
# make a generator for the example tree with at most two unsampled hosts
generator.2us <- tt.generator(stratus.example.tree, max.unsampled = 2)
# make a generator for the example tree with no infection after sampling
generator.limits <- tt.generator(stratus.example.tree, max.sampling.to.noninfectious = 0)
# make a generator with multiple sampling defined by the vector grouping.map
generator.ms <- tt.generator(stratus.example.tree, tip.map = grouping.map)
```

Index