Package ‘nodeSub’

October 13, 2022

Type Package

Title Simulate DNA Alignments Using Node Substitutions

Version 1.2.5

Author Thijs Janzen

Maintainer Thijs Janzen <thijsjanzen@gmail.com>

Description Simulate DNA sequences for the node substitution model.

In the node substitution model, substitutions accumulate additionally
during a speciation event, providing a potential mechanistic explanation for
substitution rate variation. This package provides tools to simulate
such a process, simulate a reference process with only substitutions along
the branches, and provides tools to infer phylogenies from alignments. More
information can be found in Janzen (2021) <doi:10.1093/sysbio/syab085>.

URL https://github.com/thijsjanzen/nodeSub

BugReports https://github.com/thijsjanzen/nodeSub

License GPL-3

Encoding UTF-8

RoxygenNote 7.2.0

VignetteBuilder knitr

LinkingTo Rcpp

Depends Rcpp, ape

Imports apTreeshape, phangorn, tibble, DDD, Rmpfr, assertthat,
phapply, phylobase, geiger, babette, beautier, beastier,
mauricer, tracerer, rappdirs, testit, stringr

Suggests testthat, TreeSim, dplyr, knitr, rmarkdown, ggplot2,
magrittr, tidyR, nLTT, RPANDA

NeedsCompilation yes

Repository CRAN

Date/Publication 2022-08-19 12:30:05 UTC
nodeSub-package

Package providing functions to simulate sequences under different DNA evolution models

Description

Simulate DNA sequences for the node substitution model. In the node substitution model, substitutions accumulate additionally during a speciation event, providing a potential mechanistic explanation for substitution rate variation. This package provides tools to simulate such a process, simulate a reference process with only substitutions along the branches, and provides tools to infer phylogenies from alignments. More information can be found in Janzen (2021) <doi:10.1093/sysbio/syab085>.

Version History:

Version 1.2.3 - Removed summary statistic tests for CRAN
Version 1.2.2 - Changed codedov links in README
Version 1.2.1 - Expanded dependency on RPANDA
Version 1.2 - Release on CRAN

Author(s)

Thijs Janzen Maintainer: Thijs Janzen <t.janzen@rug.nl>
calc_expected_hidden_nodes

References

Thijs Janzen, Folmer Bokma, Rampal S Etienne, Nucleotide Substitutions during Speciation may Explain Substitution Rate Variation, Systematic Biology, 2021; syab085

calc_expected_hidden_nodes

*Calculate the number of expected hidden nodes in a phylogenetic tree*

Description

Calculate the number of expected hidden nodes using equation 1 in Manceau et al. 2020

Usage

```
calc_expected_hidden_nodes(phy, lambda = NULL, mu = NULL)
```

Arguments

- **phy**: phylogenetic tree
- **lambda**: birth rate
- **mu**: death rate

Value

expected number of hidden nodes

References


calc_fraction

*Calculate the expected fraction of substitutions at the nodes, relative to the fraction at the branches*

Description

calculates the relative contribution of substitutions at the nodes

Usage

```
calc_fraction(phy = NULL, node_time = 0, model = "unlinked")
```
calc_sum_stats

Arguments

phy       phylogenetic tree (optional)
node_time time spent at the node
model     node substitution model

Value

expected fraction

calc_required_node_time

Calculate the required node time to obtain a desired fraction of substitutions at the node

Description

calculates the required node time to obtain a desired fraction of substitutions at the node

Usage

calc_required_node_time(phy = NULL, s = 0.5, model = "unlinked")

Arguments

phy       phylogenetic tree
s         desired fraction
model     node substitution model, either "linked" or "unlinked".

Value

expected fraction

calc_sum_stats

calculate summary statistics of a phylogenetic tree, compared with a reference tree. The following statistics are calculated: the beta statistic, gamma statistic, crown age, mean branch length, number of tips, the nLTT statistic and the laplacian difference, given by RPANDA's JSDtree. Because JSDtree can sometimes cause issues, some additional checks are performed to ensure that is possible to run this function.
**count_hidden**

**Description**

calculate summary statistics of a phylogenetic tree, compared with a reference tree. The following
statistics are calculated: the beta statistic, gamma statistic, crown age, mean branch length, number
of tips, the nLTT statistic and the laplacian difference, given by RPANDA’s JSDtree. Because
JSDtree can sometimes cause issues, some additional checks are performed to ensure that is possible
to run this function.

**Usage**

calc_sum_stats(trees, true_tree, verbose = FALSE)

**Arguments**

trees          a phyloList object containing multiple trees
true_tree      a phylo object containing the reference tree, preferably without extinct lineages.
               If extinct lineages are found, these are dropped.
verbose        verbose output if true (e.g. progressbars)

**Value**

list with two tibbles 1) containing the summary statistics of all trees and 2) containing the difference
with the true tree

**count_hidden**

Function to calculate the number of hidden speciation events, e.g. speciation events that have lead to an extinct species. Thus, these hidden speciation events can only be detected in complete trees (as opposed to reconstructed trees).

**Description**

Function to calculate the number of hidden speciation events, e.g. speciation events that have lead to an extinct species. Thus, these hidden speciation events can only be detected in complete trees (as opposed to reconstructed trees).

**Usage**

count_hidden(tree)

**Arguments**

tree           phylo object

**Value**

number of hidden speciation events
create_balanced_tree  
create a balanced tree out of branching times

Description

create a balanced tree out of branching times

Usage

create_balanced_tree(brts)

Arguments

brts  
vector of branching times

Value

phylo phylo object

---

create_equal_alignment  

function create an alignment with identical information content

Description

function create an alignment with identical information content

Usage

create_equal_alignment(  
  input_tree,  
  sub_rate,  
  alignment_result,  
  sim_function = NULL,  
  verbose = FALSE,  
  node_time = NULL,  
  input_alignment_type = "nodesub"  
)

**create_equal_alignment_explicit**

Function create an alignment with identical information content, using the explicit method to simulate substitutions.

**Arguments**

- **input_tree**: phylogeny for which to generate alignment
- **sub_rate**: substitution rate used in the original phylogeny
- **alignment_result**: result of sim_normal, sim_linked or sim_unlinked
- **sim_function**: function that accepts a tree, sequence length, rootsequence and substitution rate (in that order). Default is sim_normal
- **verbose**: provide intermediate output
- **node_time**: node time
- **input_alignment_type**: was the input alignment simulated with a node substitution model or a normal substitution model? Used to calculate the twin mutation rate. Options are "nodesub" and "normal".

**Value**

A list with four properties:
1. alignment: the alignment itself
2. adjusted_rate: the substitution rate used to obtain identical information content
3. total_accumulated_substitutions: the total number of substitutions accumulated.
4. total_node_substitutions: total number of substitutions accumulated on the nodes
5. total_branch_substitutions: total number of substitutions accumulated on the branches.

**Usage**

```r
create_equal_alignment_explicit(
    input_tree, sub_rate, alignment_result, verbose = FALSE
)
```
create_unbalanced_tree

create an unbalanced tree out of branching times

Description

create an unbalanced tree out of branching times

Usage

create_unbalanced_tree(brts)

Arguments

brts vector of branching times

Value

phylo phylo object
estimate_marginal_models

estimate the marginal likelihood of the relaxed and strict clock model for a provided alignment

Description

estimate_marginal_models estimates the marginal likelihood of both the strict and the relaxed clock model, given the JC69 substitution model, using the NS package in BEAST, made available via the babette R package. The NS package performs nested sampling, and uses an MCMC approach to estimate the marginal likelihood. Sampling is performed until convergence of the MCMC chain.

Usage

```r
estimate_marginal_models(
  fasta_filename,
  use_yule_prior = FALSE,
  rng_seed = 42,
  sub_rate = 1,
  verbose = FALSE
)
```

Arguments

- `fasta_filename`: file name of fasta file holding alignment for which the marginal likelihood is to be estimated
- `use_yule_prior`: by default, a birth-death prior is used as tree prior, but if `use_yule_prior` is set to TRUE, a pure-birth prior will be used.
- `rng_seed`: seed of pseudo-random number generator
- `sub_rate`: substitution rate
- `verbose`: boolean indicating if verbose intermediate output is to be generated

Value

data frame with marginal likelihoods and relative weights per clock model.

get_p_matrix

`calculate p matrix`

Description

calculates the p matrix
infer_phylogeny

Usage

get_p_matrix(branch_length, eig = phangorn::edQt(), rate = 1)

Arguments

branch_length  branch length
eig           eigen object
rate           rate

Value

p matrix

infer_phylogeny(infer the time calibrated phylogeny associated with the provided alignment. This function uses the R package babette to infer the phylogeny using BEAST2.

Description

infer the time calibrated phylogeny associated with the provided alignment. This function uses the R package babette to infer the phylogeny using BEAST2.

Usage

infer_phylogeny(
  alignment,
  treatment_name,
  tree_prior = beautier::create_bd_tree_prior(),
  clock_prior = beautier::create_strict_clock_model(),
  mcmc_seed = NULL,
  chain_length = 1e+07,
  sample_interval = 5000,
  burnin = 0.1,
  working_dir = NULL,
  sub_rate = 1
)

Arguments

alignment     Phydat object containing the focal alignment
treatment_name string to be appended to BEAST files
tree_prior    tree prior used, default = birth-death prior
clock_prior   clock prior used, default = strict clock
mcmc_seed     seed of the mcmc chain, default is the system time
**reduce_tree**

- **chain_length**: length of the mcmc chain, default is 1e7.
- **sample_interval**: interval of sampling, default is 5000
- **burnin**: burnin of posterior distribution
- **working_dir**: beast2 working dir
- **sub_rate**: substitution rate used to generate the original alignment (if available), default is 1

**Value**

list with all trees, and the consensus tree

---

**Function to remove speciation events occurring after an extinction event. Extinct species are pruned randomly, such that only a single extinct species per branching event (if any extinct species) remains.**

**Description**

Function to remove speciation events occurring after an extinction event. Extinct species are pruned randomly, such that only a single extinct species per branching event (if any extinct species) remains.

**Usage**

```r
reduce_tree(tree)
```

**Arguments**

- **tree**: phylo object

**Value**

pruned tree
**sim_linked**

simulate a sequence assuming conditional substitutions on the node.

**Description**

simulate a sequence assuming conditional substitutions on the node.

**Usage**

```
sim_linked(
    phy,
    Q = rep(1, 6),
    rate = 0.1,
    node_mut_rate_double = 1e-09,
    l = 1000,
    bf = rep(0.25, 4),
    rootseq = NULL,
    node_time = 0.01
)
```

**Arguments**

- **phy**: tree for which to simulate sequences
- **Q**: substitution matrix along the branches, default = JC
- **rate**: mutation rate, default = 1
- **node_mut_rate_double**: mutation rate on the node, default = 1e-9
- **l**: number of base pairs to simulate
- **bf**: base frequencies, default = c(0.25, 0.25, 0.25, 0.25)
- **rootseq**: sequence at the root, simulated by default
- **node_time**: time spent at the node

**Value**

list with four items

1. alignment Phydat object with the resulting alignment
2. rootseq the rootsequence used
3. total_branch_substitutions total number of substitutions accumulated on the branches
4. total_node_substitutions total number of substitutions accumulated at the nodes
Simulate sequences for a given evolutionary tree, using a standard model of sequence evolution along the branches. Code for this function was heavily inspired by the function simSeq from the phangorn package.

**Usage**

```r
sim_normal(x, l = 1000, Q = NULL, bf = NULL, rootseq = NULL, rate = 1)
```

**Arguments**

- `x`: a phylogenetic tree `tree`, i.e. an object of class `phylo`
- `l`: length of the sequence to simulate.
- `Q`: the rate matrix.
- `bf`: base frequencies.
- `rootseq`: a vector of length `l` containing the root sequence, other root sequence is randomly generated.
- `rate`: mutation rate

**Value**

list with four items

1. alignment Phydat object with the resulting alignment
2. rootseq the rootsequence used
3. total_branch_substitutions total number of substitutions accumulated on the branches
4. total_node_substitutions total number of substitutions accumulated at the nodes

**Author(s)**

Klaus Schliep <klaus.schliep@gmail.com>
**sim_normal_explicit**

*simulate a sequence assuming substitutions are only accumulated along the branches, using the explicit simulation method (e.g. reverse substitutions are modeled explicitly)*

**Description**

simulate a sequence assuming substitutions are only accumulated along the branches, using the explicit simulation method (e.g. reverse substitutions are modeled explicitly)

**Usage**

```r
sim_normal_explicit(x, l = 1000, Q = NULL, bf = NULL, rootseq = NULL, rate = 1)
```

**Arguments**

- `x`: a phylogenetic tree, i.e. an object of class phylo or and object of class pml.
- `l`: length of the sequence to simulate.
- `Q`: the rate matrix.
- `bf`: base frequencies.
- `rootseq`: a vector of length l containing the root sequence, other root sequence is randomly generated.
- `rate`: mutation rate or scaler for the edge length, a numerical value greater than zero.

**Value**

list with four items

1. alignment Phydat object with the resulting alignment
2. rootseq the rootsequence used
3. total_branch_substitutions total number of substitutions accumulated on the branches
4. total_node_substitutions total number of substitutions accumulated at the nodes

**sim_unlinked**

*Simulate a sequence assuming node substitutions are not shared amongst offspring, given two substitution matrices: one for substitutions occurring on the nodes, and one for substitutions occurring along the branches.*

**Description**

Simulate a sequence assuming node substitutions are not shared amongst offspring, given two substitution matrices: one for substitutions occurring on the nodes, and one for substitutions occurring along the branches.
**Usage**

```r
sim_unlinked_explicit(  
  phy,  
  Q1 = rep(1, 6),  
  Q2 = rep(1, 6),  
  rate1 = 0.1,  
  rate2 = 0.1,  
  l = 1000,  
  bf = rep(0.25, 4),  
  rootseq = NULL,  
  node_time = 0.001  
)
```

**Arguments**

- **phy**: tree for which to simulate sequences
- **Q1**: substitution matrix along the branches, default = JC
- **Q2**: substitution matrix on the nodes, default = JC
- **rate1**: mutation rate along the branch, default = 0.1
- **rate2**: mutation rate on the node, default = 0.1
- **l**: number of base pairs to simulate
- **bf**: base frequencies, default = c(0.25, 0.25, 0.25, 0.25)
- **rootseq**: sequence at the root, simulated by default
- **node_time**: amount of time spent at the nodes

**Value**

list with four items

1. alignment Phydat object with the resulting alignment
2. rootseq the rootsequence used
3. total_branch_substitutions total number of substitutions accumulated on the branches
4. total_node_substitutions total number of substitutions accumulated at the nodes

---

**sim_unlinked_explicit**  
*Simulate a sequence assuming node substitutions are not shared amongst offspring, using the explicit simulation method (e.g. reverse substitutions are modeled explicitly)*

**Description**

Simulate a sequence assuming node substitutions are not shared amongst offspring, using the explicit simulation method (e.g. reverse substitutions are modeled explicitly)
Usage

```r
sim_unlinked_explicit(
  phy,
  Q1 = rep(1, 6),
  Q2 = rep(1, 6),
  rate1 = 0.1,
  rate2 = 0.1,
  l = 1000,
  bf = rep(0.25, 4),
  rootseq = NULL,
  node_time = 0.001
)
```

Arguments

- **phy**: phylogenetic tree for which to simulate sequences
- **Q1**: substitution matrix along the branches, default = JC
- **Q2**: substitution matrix on the nodes, default = JC
- **rate1**: mutation rate along the branch, default = 0.1
- **rate2**: mutation rate on the node, default = 0.1
- **l**: number of base pairs to simulate
- **bf**: base frequencies, default = c(0.25, 0.25, 0.25, 0.25)
- **rootseq**: sequence at the root, simulated by default
- **node_time**: amount of time spent at the nodes

Value

- list with four items
  1. alignment Phydat object with the resulting alignment
  2. rootseq the rootsequence used
  3. total_branch_substitutions total number of substitutions accumulated on the branches
  4. total_node_substitutions total number of substitutions accumulated at the nodes

slow_matrix

- **this function calculates the p matrix within R** this is slower than the C++ implementation in get_p_matrix but provides a way to debug and verify

Description

this function calculates the p matrix within R this is slower than the C++ implementation in get_p_matrix but provides a way to debug and verify
**slow_matrix**

**Usage**

```
slow_matrix(eig, branch_length, rate)
```

**Arguments**

- **eig**: eigen object
- **branch_length**: branch length
- **rate**: substitution rate

**Value**

- p matrix
Index

calc_expected_hidden_nodes, 3
calc_fraction, 3
calc_required_node_time, 4
calc_sum_stats, 4
count_hidden, 5
create_balanced_tree, 6
create_equal_alignment, 6
create_equal_alignment_explicit, 7
create_unbalanced_tree, 8
estimate_marginal_models, 9
get_p_matrix, 9
infer_phylogeny, 10
nodeSub (nodeSub-package), 2
nodeSub-package, 2
reduce_tree, 11
sim_linked, 12
sim_normal, 13
sim_normal_explicit, 14
sim_unlinked, 14
sim_unlinked_explicit, 15
slow_matrix, 16