Package 'babette'

February 1, 2020

Title Control 'BEAST2'

Version 2.1.2

Maintainer Richèl J.C. Bilderbeek <richel@richelbilderbeek.nl>

Description 'BEAST2' (<https://www.beast2.org>) is a widely used
Bayesian phylogenetic tool, that uses DNA/RNA/protein data
and many model priors to create a posterior of jointly estimated
phylogenies and parameters.
'BEAST2' is commonly accompanied by 'BEAUti 2', 'Tracer' and 'DensiTree'.
'babette' provides for an alternative workflow of using
all these tools separately. This allows doing complex Bayesian
phylogenetics easily and reproducibly from 'R'.

License GPL-3

LazyData true

RoxygenNote 7.0.2

VignetteBuilder knitr

Depends R (>= 3.5.0), beautier (>= 2.3), beastier (>= 2.1), mauricer,
tracerer

Imports phangorn, remotes, stringr, testit, xml2

Suggests ape, ggplot2, hunspell, knitr, lintr, nLTT, rappdirs,
rmarkdown, spelling, testthat (>= 2.1.0)

Language en-US

Encoding UTF-8

SystemRequirements BEAST2 (https://www.beast2.org/)

NeedsCompilation no

Author Richèl J.C. Bilderbeek [aut, cre]
(<https://orcid.org/0000-0003-1107-7049>),
Joëlle Barido-Sottani [rev] (Joëlle reviewed the package for rOpenSci,
see https://github.com/richelbilderbeek/onboarding/issues/209),
David Winter [rev] (David reviewed the package for rOpenSci, see
https://github.com/richelbilderbeek/onboarding/issues/209),
Giovanni Laudanno [ctb]

Repository CRAN

Date/Publication 2020-02-01 20:00:02 UTC

R topics documented:

`babette` *babette: A package for Bayesian phylogenetics.*

Description

'babette' provides for an alternative workflow of using the popular phylogenetics tool 'BEAST2', including it peripheral tools. From an alignment and inference model, a posterior of jointly estimated phylogenies and parameter estimates is generated.

See Also

Use bbt_self_test to do verify babette is installed correctly.

These are packages associated with 'babette':

- 'beautier' creates 'BEAST2' input files.
- 'beastier' runs 'BEAST2'.
- 'mauricer' does 'BEAST2' package management.
- 'tracerer' parses 'BEAST2' output files.

Examples

```
library(testthat)

if (is_beast2_installed()) {

  inference_model <- create_test_inference_model()
  beast2_options <- create_beast2_options()

  out <- bbt_run_from_model(
    fasta_filename = get_babette_path("anthus_aco.fas"),
    inference_model = inference_model,
    beast2_options = beast2_options
  )

  expect_true("estimates" %in% names(out))
  expect_true("anthus_aco_trees" %in% names(out))
  expect_true("operators" %in% names(out))
  expect_true("output" %in% names(out))
  expect_true(is_phylo(out$anthus_aco_trees[[1]]))

  #' The number of expected trees. The tree at state zero is also logged
  n_trees_expected <- 1 + (inference_model$mcmc$chain_length /
    inference_model$mcmc$treelog$log_every
  )
  expect_equal(length(out$anthus_aco_trees), n_trees_expected)

  expect_true("Sample" %in% names(out$estimates))
  expect_true("posterior" %in% names(out$estimates))
  expect_true("likelihood" %in% names(out$estimates))
  expect_true("prior" %in% names(out$estimates))
  expect_true("treeLikelihood" %in% names(out$estimates))
  expect_true("TreeHeight" %in% names(out$estimates))
  expect_true("YuleModel" %in% names(out$estimates))
  expect_true("birthRate" %in% names(out$estimates))

  expect_true("operator" %in% names(out$operators))
  expect_true("p" %in% names(out$operators))
  expect_true("accept" %in% names(out$operators))
  expect_true("reject" %in% names(out$operators))
  expect_true("acceptFC" %in% names(out$operators))
  expect_true("rejectFC" %in% names(out$operators))
  expect_true("rejectIv" %in% names(out$operators))
  expect_true("rejectOp" %in% names(out$operators))

  # Clean up temporary files created by babette
  bbt_delete_temp_files(
    inference_model = inference_model,
    beast2_options = beast2_options
  )
}
```

bbt_delete_temp_files *Delete all the temporary files created by bbt_run_from_model*

Description

Delete all the temporary files created by bbt_run_from_model

Usage

```
bbt_delete_temp_files(inference_model, beast2_options)
```

Arguments

inference_model	a Bayesian phylogenetic inference model, as returned by create_inference_model
beast2_options	'BEAST2' options, as can be created by create_beast2_options

Examples

```
library(testthat)

if (is_beast2_installed()) {
  # Do a minimal run
  inference_model <- create_test_inference_model()
  beast2_options <- create_beast2_options()
  expect_silent(
    bbt_run_from_model(
      fasta_filename = get_fasta_filename(),
      inference_model = inference_model,
      beast2_options = beast2_options
    )
  )
  # Temporary files are created by 'bbt_run'
  expect_true(file.exists(inference_model$mcmc$tracelog$filename))
  expect_true(file.exists(inference_model$mcmc$treelog$filename))
  expect_true(file.exists(inference_model$mcmc$screenlog$filename))
  expect_true(file.exists(beast2_options$input_filename))
  expect_true(file.exists(beast2_options$output_state_filename))

  bbt_delete_temp_files(
    inference_model = inference_model,
    beast2_options = beast2_options
  )

  # Temporary files are deleted
  expect_false(file.exists(inference_model$mcmc$tracelog$filename))
  expect_false(file.exists(inference_model$mcmc$treelog$filename))
  expect_false(file.exists(inference_model$mcmc$screenlog$filename))
  expect_false(file.exists(beast2_options$input_filename))
```

```
  expect_false(file.exists(beast2_options$output_state_filename))
}
```

---

bbt_run *Do a full run: create a 'BEAST2' configuration file (like 'BEAUti 2'), run 'BEAST2', parse results (like 'Tracer')*

Description

Prefer using bbt_run_from_model, as it has a cleaner interface.

Usage

```
bbt_run(
  fasta_filename,
  tipdates_filename = NA,
  site_model = beautier::create_jc69_site_model(),
  clock_model = beautier::create_strict_clock_model(),
  tree_prior = beautier::create_yule_tree_prior(),
  mrca_prior = NA,
  mcmc = beautier::create_mcmc(),
  beast2_input_filename = beastier::create_temp_input_filename(),
  rng_seed = 1,
  beast2_output_state_filename = beastier::create_temp_state_filename(),
  beast2_path = beastier::get_default_beast2_path(),
  overwrite = TRUE,
  verbose = FALSE
)
```

Arguments

fasta_filename	a FASTA filename
tipdates_filename	name of the file containing tip dates
site_model	one site model, see create_site_models
clock_model	one clock model, see create_clock_model
tree_prior	one tree priors, as created by create_tree_prior
mrca_prior	one Most Recent Common Ancestor prior, as returned by create_mrca_prior
mcmc	the MCMC options, see create_mcmc
beast2_input_filename	path of the 'BEAST2' configuration file. By default, this file is put in a temporary folder with a random filename, as the user needs not read it: it is used as input of 'BEAST2'. Specifying a beast2_input_filename allows to store that file in a more permanently stored location.

rng_seed	the random number generator seed. Must be either NA or a positive non-zero value. An RNG seed of NA results in 'BEAST2' picking a random seed.
beast2_output_state_filename	name of the final state file created by 'BEAST2', containing the operator acceptances. By default, this file is put a temporary folder with a random filename, as the user needs not read it: its content is parsed and returned by this function. Specifying a beast2_output_state_filename allows to store that file in a more permanently stored location.
beast2_path	name of either a 'BEAST2' binary file (usually simply beast) or a 'BEAST2' jar file (usually has a .jar extension). Use get_default_beast2_bin_path to get the default BEAST binary file's path Use get_default_beast2_jar_path to get the default BEAST jar file's path
overwrite	will 'BEAST2' overwrite files? Like 'BEAST2', this is set to TRUE by default. If TRUE, 'BEAST2' will overwrite the beast2_options$output_state_filename if its present. If FALSE, 'BEAST2' will not overwrite the beast2_options$output_state_filename if its present and babette will give an error message. Note that if overwrite is set to FALSE when a tracelog (see create_tracelog), screenlog (see create_screenlog) or treelog (see create_treelog) file already exists, 'BEAST2' (and thus babette) will freeze.
verbose	set to TRUE for more output

Value

a list with the following elements:

- estimates: a data frame with 'BEAST2' parameter estimates
- [alignment_id]_trees: a multiPhylo containing the phylogenies in the 'BEAST2' posterior. [alignment_id] is the ID of the alignment. For example, when running bbt_run with anthus_aco.fas, this element will have name anthus_aco_trees
- operators: a data frame with the 'BEAST2' MCMC operator acceptances
- output: a numeric vector with the output sent to standard output and error streams
- ns: (optional) the results of a marginal likelihood estimation, will exist only when create_ns_mcmc was used for the MCMC. This structure will contain the following elements:
  - marg_log_lik the marginal log likelihood estimate
  - marg_log_lik_sd the standard deviation around the estimate
  - estimates the parameter estimates created during the marginal likelihood estimation
  - trees the trees created during the marginal likelihood estimation

Author(s)

Richèl J.C. Bilderbeek

See Also

Use remove_burn_ins to remove the burn-ins from the posterior's estimates (posterior$estimates)

Examples

```
library(testthat)

if (is_beast2_installed()) {

  mcmc <- create_test_mcmc()
  mcmc$tracelog$filename <- tempfile()
  mcmc$treelog$filename <- tempfile()
  mcmc$screenlog$filename <- tempfile()

  out <- bbt_run(
    fasta_filename = get_babette_path("anthus_aco.fas"),
    beast2_input_filename = tempfile(),
    beast2_output_state_filename = tempfile(),
    mcmc = mcmc
  )

  expect_true("estimates" %in% names(out))
  expect_true("anthus_aco_trees" %in% names(out))
  expect_true("operators" %in% names(out))
  expect_true("output" %in% names(out))
  expect_true(is_phylo(out$anthus_aco_trees[[1]]))

  #' The number of expected trees. The tree at state zero is also logged
  n_trees_expected <- 1 + (mcmc$chain_length /
    mcmc$treelog$log_every
  )
  expect_equal(length(out$anthus_aco_trees), n_trees_expected)

  expect_true("Sample" %in% names(out$estimates))
  expect_true("posterior" %in% names(out$estimates))
  expect_true("likelihood" %in% names(out$estimates))
  expect_true("prior" %in% names(out$estimates))
  expect_true("treeLikelihood" %in% names(out$estimates))
  expect_true("TreeHeight" %in% names(out$estimates))
  expect_true("YuleModel" %in% names(out$estimates))
  expect_true("birthRate" %in% names(out$estimates))

  expect_true("operator" %in% names(out$operators))
  expect_true("p" %in% names(out$operators))
  expect_true("accept" %in% names(out$operators))
  expect_true("reject" %in% names(out$operators))
  expect_true("acceptFC" %in% names(out$operators))
  expect_true("rejectFC" %in% names(out$operators))
  expect_true("rejectIv" %in% names(out$operators))
  expect_true("rejectOp" %in% names(out$operators))
}
```

`bbt_run_from_model` *Do a full run: create a 'BEAST2' configuration file (like 'BEAUti 2'), run 'BEAST2', parse results (like 'Tracer')*

Description

Do a full run: create a 'BEAST2' configuration file (like 'BEAUti 2'), run 'BEAST2', parse results (like 'Tracer')

Usage

```
bbt_run_from_model(
  fasta_filename,
  inference_model = beautier::create_inference_model(),
  beast2_options = beastier::create_beast2_options()
)
```

Arguments

`fasta_filename`	a FASTA filename
`inference_model`	a Bayesian phylogenetic inference model, as returned by create_inference_model
`beast2_options`	'BEAST2' options, as can be created by create_beast2_options

Value

a list with the following elements:

- `estimates`: a data frame with 'BEAST2' parameter estimates
- `[alignment_id]_trees`: a `multiPhylo` containing the phylogenies in the 'BEAST2' posterior. `[alignment_id]` is the ID of the alignment. For example, when running `bbt_run_from_model` with `anthus_aco.fas`, this element will have name `anthus_aco_trees`
- `operators`: a data frame with the 'BEAST2' MCMC operator acceptances
- `output`: a numeric vector with the output sent to standard output and error streams
- `ns`: (optional) the results of a marginal likelihood estimation, will exist only when `create_ns_mcmc` was used for `mcmc`. This structure will contain the following elements:
  - `marg_log_lik` the marginal log likelihood estimate
  - `marg_log_lik_sd` the standard deviation around the estimate
  - `estimates` the parameter estimates created during the marginal likelihood estimation
  - `trees` the trees created during the marginal likelihood estimation

Author(s)

Richèl J.C. Bilderbeek

See Also

Use `remove_burn_ins` to remove the burn-ins from the posterior's estimates (`posterior$estimates`)

Examples

```
library(testthat)

if (is_beast2_installed()) {

  inference_model <- create_test_inference_model()

  out <- bbt_run_from_model(
    fasta_filename = get_babette_path("anthus_aco.fas"),
    inference_model = inference_model
  )

  expect_true("estimates" %in% names(out))
  expect_true("anthus_aco_trees" %in% names(out))
  expect_true("operators" %in% names(out))
  expect_true("output" %in% names(out))
  expect_true(is_phylo(out$anthus_aco_trees[[1]]))

  #' The number of expected trees. The tree at state zero is also logged
  n_trees_expected <- 1 + (inference_model$mcmc$chain_length /
    inference_model$mcmc$treelog$log_every
  )
  expect_equal(length(out$anthus_aco_trees), n_trees_expected)

  expect_true("Sample" %in% names(out$estimates))
  expect_true("posterior" %in% names(out$estimates))
  expect_true("likelihood" %in% names(out$estimates))
  expect_true("prior" %in% names(out$estimates))
  expect_true("treeLikelihood" %in% names(out$estimates))
  expect_true("TreeHeight" %in% names(out$estimates))
  expect_true("YuleModel" %in% names(out$estimates))
  expect_true("birthRate" %in% names(out$estimates))

  expect_true("operator" %in% names(out$operators))
  expect_true("p" %in% names(out$operators))
  expect_true("accept" %in% names(out$operators))
  expect_true("reject" %in% names(out$operators))
  expect_true("acceptFC" %in% names(out$operators))
  expect_true("rejectFC" %in% names(out$operators))
  expect_true("rejectIv" %in% names(out$operators))
  expect_true("rejectOp" %in% names(out$operators))
}
```

`bbt_self_test` *Do a self test to verify babette that works correctly.*

Description

Do a self test to verify babette that works correctly.

Usage

```
bbt_self_test()
```

Author(s)

Richèl J.C. Bilderbeek

Examples

```
library(testthat)

if (is_beast2_installed()) {
  expect_silent(bbt_self_test())
} else {
  expect_error(bbt_self_test())
}
```

---

check_beast2_pkgs *Checks if* bbt_run *has the 'BEAST2' packages needed to process its arguments. Will stop if not.*

Description

For example, to use a Nested Sampling MCMC, the 'BEAST2' 'NS' package needs to be installed.

Usage

```
check_beast2_pkgs(mcmc, beast2_path = get_default_beast2_bin_path())
```

Arguments

mcmc	the MCMC options, see create_mcmc
beast2_path	name of either a 'BEAST2' binary file (usually simply beast) or a 'BEAST2' jar file (usually has a .jar extension). Use get_default_beast2_bin_path to get the default BEAST binary file's path Use get_default_beast2_jar_path to get the default BEAST jar file's path

Examples

```
library(testthat)

# This test uninstalls the NS 'BEAST2' package.
# Only do that on CI services, else a user without internet
# suddenly finds the NS 'BEAST2' package installed and unable
# to reinstall it
if (is_beast2_installed() && is_on_ci()) {

  # Check to need to install NS later
```

```
  was_ns_installed <- is_beast2_ns_pkg_installed()

  if (is_beast2_ns_pkg_installed()) {
    uninstall_beast2_pkg("NS")
  }

  # Without the NS 'BEAST2' package installed,
  # a Nested Sampling MCMC cannot be created.
  expect_false(is_beast2_ns_pkg_installed())
  expect_error(
    check_beast2_pkgs(
      mcmc = create_ns_mcmc()
    ),
    "Must install 'NS' to use 'create_ns_mcmc'."
  )

  install_beast2_pkg("NS")

  expect_silent(
    check_beast2_pkgs(
      mcmc = create_ns_mcmc()
    )
  )

  if (!was_ns_installed) {
    uninstall_beast2_pkg("BS")
  }
}
```

`create_test_bbt_run_output` *Get an example output of* `bbt_run` *or* `bbt_run_from_model`.

Description

This output is used in testing.

Usage

```
create_test_bbt_run_output()
```

Value

the same results as `bbt_run` or `bbt_run_from_model`

Author(s)

Richèl J.C. Bilderbeek

Examples

```
out <- create_test_bbt_run_output()

library(testthat)
expect_true("estimates" %in% names(out))
expect_true("anthus_aco_trees" %in% names(out))
expect_true("operators" %in% names(out))
expect_true("output" %in% names(out))
expect_true(is_phylo(out$anthus_aco_trees[[1]]))
expect_true(length(out$anthus_aco_trees) == 2)

expect_true("Sample" %in% names(out$estimates))
expect_true("posterior" %in% names(out$estimates))
expect_true("likelihood" %in% names(out$estimates))
expect_true("prior" %in% names(out$estimates))
expect_true("treeLikelihood" %in% names(out$estimates))
expect_true("TreeHeight" %in% names(out$estimates))
expect_true("YuleModel" %in% names(out$estimates))
expect_true("birthRate" %in% names(out$estimates))

expect_true("operator" %in% names(out$operators))
expect_true("p" %in% names(out$operators))
expect_true("accept" %in% names(out$operators))
expect_true("reject" %in% names(out$operators))
expect_true("acceptFC" %in% names(out$operators))
expect_true("rejectFC" %in% names(out$operators))
expect_true("rejectIv" %in% names(out$operators))
expect_true("rejectOp" %in% names(out$operators))
```

---

create_test_ns_output *Create testing output similar to when running a 'BEAST2' run with nested sampling*

Description

Create testing output similar to when running a 'BEAST2' run with nested sampling

Usage

```
create_test_ns_output()
```

Author(s)

Richèl J.C. Bilderbeek

See Also

Use parse_beast2_output_to_ns to parse this output to a Nested Sampling result. See create_ns_mcmc to see how to do a marginal likelihood estimation using Nested Sampling.

Examples

```
library(testthat)

output <- create_test_ns_output()
expect_true(is.character(output))
expect_true(length(output) > 100)
```

`default_params_doc` *This function does nothing. It is intended to inherit is parameters' documentation.*

Description

This function does nothing. It is intended to inherit is parameters' documentation.

Usage

```
default_params_doc(
  beast2_input_filename,
  beast2_options,
  beast2_output_log_filename,
  beast2_output_state_filename,
  beast2_output_trees_filenames,
  beast2_path,
  beast2_working_dir,
  cleanup,
  clock_model,
  clock_models,
  fasta_filename,
  fasta_filenames,
  inference_model,
  mcmc,
  mrca_prior,
  mrca_priors,
  overwrite,
  rng_seed,
  site_model,
  site_models,
  tipdates_filename,
  tree_prior,
  tree_priors,
  verbose
)
```

Arguments

beast2_input_filename	path of the 'BEAST2' configuration file. By default, this file is put in a temporary folder with a random filename, as the user needs not read it: it is used as input of 'BEAST2'. Specifying a beast2_input_filename allows to store that file in a more permanently stored location.
beast2_options	'BEAST2' options, as can be created by create_beast2_options
beast2_output_log_filename	name of the log file created by 'BEAST2', containing the parameter estimates in time. By default, this file is put a temporary folder with a random filename, as the user needs not read it: its content is parsed and returned by this function. Specifying a beast2_output_log_filename allows to store that file in a more permanently stored location.
beast2_output_state_filename	name of the final state file created by 'BEAST2', containing the operator acceptances. By default, this file is put a temporary folder with a random filename, as the user needs not read it: its content is parsed and returned by this function. Specifying a beast2_output_state_filename allows to store that file in a more permanently stored location.
beast2_output_trees_filenames	name of the one or more trees files created by 'BEAST2', one per alignment. By default, these files are put a temporary folder with a random filename, as the user needs not read it: their content is parsed and returned by this function. Specifying beast2_output_trees_filenames allows to store these one or more files in a more permanently stored location.
beast2_path	name of either a 'BEAST2' binary file (usually simply beast) or a 'BEAST2' jar file (usually has a .jar extension). Use get_default_beast2_bin_path to get the default BEAST binary file's path Use get_default_beast2_jar_path to get the default BEAST jar file's path
beast2_working_dir	the folder 'BEAST2' will work in. This is an (empty) temporary folder by default. This allows to call 'BEAST2' in multiple parallel processes, as each process can have its own working directory
cleanup	set to FALSE to keep all temporary files
clock_model	one clock model, see create_clock_model
clock_models	one or more clock models, see create_clock_models
fasta_filename	a FASTA filename
fasta_filenames	one or more FASTA filename, each with one alignment
inference_model	a Bayesian phylogenetic inference model, as returned by create_inference_model
mcmc	the MCMC options, see create_mcmc
mrca_prior	one Most Recent Common Ancestor prior, as returned by create_mrca_prior
mrca_priors	a list of one or more Most Recent Common Ancestor priors, as returned by create_mrca_prior

overwrite	will 'BEAST2' overwrite files? Like 'BEAST2', this is set to TRUE by default. If TRUE, 'BEAST2' will overwrite the beast2_options$output_state_filename if its present. If FALSE, 'BEAST2' will not overwrite the beast2_options$output_state_filename if its present and babette will give an error message. Note that if overwrite is set to FALSE when a tracelog (see create_tracelog), screenlog (see create_screenlog) or treelog (see create_treelog) file already exists, 'BEAST2' (and thus babette) will freeze.
rng_seed	the random number generator seed. Must be either NA or a positive non-zero value. An RNG seed of NA results in 'BEAST2' picking a random seed.
site_model	one site model, see create_site_models
site_models	one or more site models, see create_site_models
tipdates_filename	name of the file containing tip dates
tree_prior	one tree priors, as created by create_tree_prior
tree_priors	one or more tree priors, see create_tree_priors
verbose	set to TRUE for more output

Note

This is an internal function, so it should be marked with @noRd. This is not done, as this will disallow all functions to find the documentation parameters

Author(s)

Richèl J.C. Bilderbeek

---

get_alignment_ids_from_xml

Get the alignment IDs from one or more 'BEAST2' XML input files.

Description

Get the alignment IDs from one or more 'BEAST2' XML input files.

Usage

```
get_alignment_ids_from_xml(xml_filename)
```

Arguments

xml_filename	name of a 'BEAST2' XML input file.

Value

a character vector with one or more alignment IDs.

Author(s)

Richèl J.C. Bilderbeek

Examples

```
library(testthat)

alignment_id <- get_alignment_ids_from_xml(
  get_babette_path("2_4.xml")
)
expect_equal(alignment_id, "test_output_0")

alignment_ids <- get_alignment_ids_from_xml(
  get_babette_path("anthus_2_4.xml")
)
expect_equal(alignment_ids, c("Anthus_nd2", "Anthus_aco"))
```

---

get_babette_path *Get the full path of a file in the* inst/extdata *folder*

Description

Get the full path of a file in the inst/extdata folder

Usage

```
get_babette_path(filename)
```

Arguments

filename	the file's name, without the path

Value

the full path of the filename, if and only if the file is present. Will stop otherwise.

Author(s)

Richèl J.C. Bilderbeek

See Also

for more files, use get_babette_paths

Examples

```
library(testthat)

expect_true(file.exists(get_babette_path("anthus_aco.fas")))
expect_true(file.exists(get_babette_path("anthus_nd2.fas")))

expect_error(get_babette_path("abs.ent"))
```

---

get_babette_paths *Get the full paths of files in the* inst/extdata *folder*

Description

Get the full paths of files in the inst/extdata folder

Usage

```
get_babette_paths(filenames)
```

Arguments

filenames	the files' names, without the path

Value

the filenames' full paths, if and only if all files are present. Will stop otherwise.

Author(s)

Richèl J.C. Bilderbeek

See Also

for one file, use get_babette_path

Examples

```
library(testthat)

filenames <- c("anthus_aco.fas", "anthus_nd2.fas")
full_paths <- get_babette_paths(filenames)
expect_equal(length(full_paths), 2)
expect_true(all(file.exists(full_paths)))
```

parse_beast2_output *Process the 'BEAST2' output dependent on 'BEAST2' package specifics*

Description

Process the 'BEAST2' output dependent on 'BEAST2' package specifics

Usage

```
parse_beast2_output(out, inference_model)
```

Arguments

`out`	a list with the complete babette output, with elements: • `output` textual output of a 'BEAST2' run
`inference_model`	a Bayesian phylogenetic inference model, as returned by create_inference_model

Value

complete babette output with added attributes, which depends on the 'BEAST2' package.

- `marg_log_lik` the marginal log likelihood estimate
- `marg_log_lik_sd` the standard deviation around the estimate
- `estimates` the parameter estimates created during the marginal likelihood estimation
- `trees` the trees created during the marginal likelihood estimation

Author(s)

Richèl J.C. Bilderbeek

parse_beast2_output_to_ns *Put the info of a Nested Sampling run in a structure*

Description

Put the info of a Nested Sampling run in a structure

Usage

```
parse_beast2_output_to_ns(output)
```

Arguments

output	screen output

Value

a list with the following elements:

- marg_log_lik the marginal log likelihood estimate
- marg_log_lik_sd the standard deviation around the estimate

Author(s)

Richèl J.C. Bilderbeek

See Also

use create_test_ns_output to obtain a test screen output.

Examples

```
library(testthat)

ns <- parse_beast2_output_to_ns(
  output = create_test_ns_output()
)
expect_equal(ns$marg_log_lik, -141, tolerance = 0.2)
expect_equal(ns$marg_log_lik_sd, 1.60, tolerance = 0.5)
expect_equal(ns$ess, 5.49, tolerance = 0.2)
```

---

plot_densitree *Draw multiple trees on top of one another.*

Description

Draw multiple trees on top of one another.

Usage

```
plot_densitree(phylos, ...)
```

Arguments

phylos	one or more phylogenies, must be of class multiPhylo
...	options to be passed to phangorn's densiTree function

Value

nothing. Will produce a plot.

Author(s)

Richèl J.C. Bilderbeek

Examples

```
if (is_beast2_installed()) {
  inference_model <- create_test_inference_model()
  beast2_options <- create_beast2_options()

   out <- bbt_run_from_model(
    get_babette_path("anthus_aco.fas"),
    inference_model = inference_model,
    beast2_options = beast2_options
  )

  plot_densitree(out$anthus_aco_trees)

  # Clean up temporary files created by babette
  bbt_delete_temp_files(
    inference_model = inference_model,
    beast2_options = beast2_options
  )
}
```

---

prepare_file_creation *Prepare for 'BEAST2' creating files*

Description

The inference model and 'BEAST2' options contain paths that may point to sub-sub-sub folders. Create those folders and test if these folders can be written to

Usage

```
prepare_file_creation(inference_model, beast2_options)
```

Arguments

inference_model	a Bayesian phylogenetic inference model, as returned by create_inference_model
beast2_options	'BEAST2' options, as can be created by create_beast2_options

Examples

```
library(testthat)

# For a test inference model, the files can be prepared
inference_model <- create_test_inference_model()
beast2_options <- create_beast2_options()
expect_silent(prepare_file_creation(inference_model, beast2_options))
```

update_babette *Update all babette dependencies, by installing their latest versions*

Description

Update all babette dependencies, by installing their latest versions

Usage

```
update_babette(upgrade = "default")
```

Arguments

upgrade	One of "default", "ask", "always", or "never". "default" respects the value of the R_REMOTES_UPGRADE environment variable if set, and falls back to "ask" if unset. "ask" prompts the user for which out of date packages to upgrade. For non-interactive sessions "ask" is equivalent to "always". TRUE and FALSE are also accepted and correspond to "always" and "never" respectively.

Author(s)

Giovanni Laudanno, Richèl J.C. Bilderbeek

Examples

```
library(testthat)

if (is_on_travis()) {

  # Updates the babette dependencies without asking
  update_babette(upgrade = "always")

  # Updating again should produce no output, as there is no
  expect_silent(update_babette(upgrade = "always"))
}
```

Index