

# Package: tracerer (via r-universe)

November 27, 2024

**Type** Package

**Title** Tracer from R

**Version** 2.2.3

**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. 'Tracer' (<<https://github.com/beast-dev/tracer/>>) is a GUI tool to parse and analyze the files generated by 'BEAST2'. This package provides a way to parse and analyze 'BEAST2' input files without active user input, but using R function calls instead.

**License** GPL-3

**Imports** jsonlite, Rcpp, testit

**Suggests** ape, ggplot2, hunspell, knitr, markdown, phangorn, rappdirs, rbenchmark, reshape2, rmarkdown, spelling, stringr, testthat (>= 2.1.0)

**VignetteBuilder** knitr

**RoxygenNote** 7.2.3

**URL** <https://docs.ropensci.org/tracerer/> (website)  
<https://github.com/ropensci/tracerer/>

**BugReports** <https://github.com/ropensci/tracerer/issues>

**LinkingTo** Rcpp

**Language** en-US

**Encoding** UTF-8

**Repository** <https://ropensci.r-universe.dev>

**RemoteUrl** <https://github.com/ropensci/tracerer>

**RemoteRef** master

**RemoteSha** 82c8d826a019147b5e99412cf33e7401b94f6125

## Contents

calc_act . . . . .	2
calc_act_cpp . . . . .	3
calc_act_r . . . . .	4
calc_ess . . . . .	5
calc_esses . . . . .	6
calc_geom_mean . . . . .	6
calc_hpd_interval . . . . .	7
calc_mode . . . . .	8
calc_stderr_mean . . . . .	9
calc_std_error_of_mean_cpp . . . . .	9
calc_summary_stats . . . . .	10
calc_summary_stats_trace . . . . .	11
calc_summary_stats_traces . . . . .	12
check_trace . . . . .	13
count_trees_in_file . . . . .	14
cs_std_dev . . . . .	14
default_params_doc . . . . .	15
extract_operators_lines . . . . .	16
get_tracerer_path . . . . .	16
get_tracerer_paths . . . . .	17
get_tracerer_tempfilename . . . . .	18
is_posterior . . . . .	19
is_trees_file . . . . .	19
is_trees_posterior . . . . .	20
parse_beast_log . . . . .	21
parse_beast_output_files . . . . .	21
parse_beast_posterior . . . . .	22
parse_beast_state_operators . . . . .	24
parse_beast_tracelog_file . . . . .	24
parse_beast_trees . . . . .	25
remove_burn_in . . . . .	26
remove_burn_ins . . . . .	27
save_beast_estimates . . . . .	27
save_beast_trees . . . . .	28
<b>Index</b>	<b>29</b>

---

calc\_act

*Calculate the auto-correlation time, alternative implementation*

---

### Description

Calculate the auto-correlation time, alternative implementation

**Usage**

```
calc_act(trace, sample_interval)
```

**Arguments**

```
trace          the values
sample_interval the interval in timesteps between samples
```

**Value**

the auto\_correlation time

**Author(s)**

The original Java version of the algorithm was from Remco Bouckaert, ported to R and adapted by Richèl J.C. Bilderbeek

**See Also**

Java code can be found here: <https://github.com/CompEvol/beast2/blob/9f040ed0357c4b946ea276a481a4c654ad4fsrc/beast/core/util/ESS.java#L161> # nolint URLs can be long

**Examples**

```
trace <- sin(seq(from = 0.0, to = 2.0 * pi, length.out = 100))
# 38.18202
calc_act(trace = trace, sample_interval = 1)
```

---

calc_act_cpp	<i>Calculate the auto correlation time from</i> <a href="https://github.com/beast-dev/beast-mcmc/blob/800817772033c13061f026226e41128d21fd14f3/src/dr/inference/trace/TraceCorrelation.java#L159">https://github.com/beast-dev/beast-mcmc/blob/800817772033c13061f026226e41128d21fd14f3/src/dr/inference/trace/TraceCorrelation.java#L159</a> # nolint
--------------	--

---

**Description**

Calculate the auto correlation time from <https://github.com/beast-dev/beast-mcmc/blob/800817772033c13061f026226e41128d21fd14f3/src/dr/inference/trace/TraceCorrelation.java#L159> # nolint

**Usage**

```
calc_act_cpp(sample, sample_interval)
```

**Arguments**

sample                    sample  
sample\_interval  
                          sample interval

**Value**

the auto correlation time

**Author(s)**

Richèl J.C. Bilderbeek

---

calc_act_r	<i>Calculate the auto-correlation time using only R. Consider using <a href="#">calc_act</a> instead, as it is orders of magnitude faster</i>
------------	---

---

**Description**

Calculate the auto-correlation time using only R. Consider using [calc\\_act](#) instead, as it is orders of magnitude faster

**Usage**

```
calc_act_r(trace, sample_interval)
```

**Arguments**

trace                    the values  
sample\_interval  
                          the interval in timesteps between samples

**Value**

the auto correlation time

**Author(s)**

The original Java version of the algorithm was from Remco Bouckaert, ported to R and adapted by Richèl J.C. Bilderbeek

**See Also**

Java code can be found here: <https://github.com/CompEvol/beast2/blob/9f040ed0357c4b946ea276a481a4c654ad4fsrc/beast/core/util/ESS.java#L161> # nolint URLs can be long

**Examples**

```
trace <- sin(seq(from = 0.0, to = 2.0 * pi, length.out = 100))
calc_act_r(trace = trace, sample_interval = 1) # 38.18202
```

---

calc_ess	<i>Calculates the Effective Sample Size</i>
----------	---

---

**Description**

Calculates the Effective Sample Size

**Usage**

```
calc_ess(trace, sample_interval)
```

**Arguments**

trace                   the values without burn-in  
sample\_interval        the interval in timesteps between samples

**Value**

the effective sample size

**Author(s)**

The original Java version of the algorithm was from Remco Bouckaert, ported to R and adapted by Richèl J.C. Bilderbeek

**See Also**

Java code can be found here: <https://github.com/CompEvol/beast2/blob/9f040ed0357c4b946ea276a481a4c654ad4fsrc/beast/core/util/ESS.java#L161> # nolint URLs can be long

**Examples**

```
filename <- get_tracerer_path("beast2_example_output.log")
estimates <- parse_beast_tracelog_file(filename)
calc_ess(estimates$posterior, sample_interval = 1000)
```

---

calc_esses	<i>Calculates the Effective Sample Sizes from a parsed BEAST2 log file</i>
------------	--

---

**Description**

Calculates the Effective Sample Sizes from a parsed BEAST2 log file

**Usage**

```
calc_esses(traces, sample_interval)
```

**Arguments**

traces            a dataframe with traces with removed burn-in  
sample\_interval   the interval in timesteps between samples

**Value**

the effective sample sizes

**Author(s)**

Richèl J.C. Bilderbeek

**Examples**

```
# Parse an example log file
estimates <- parse_beast_tracelog_file(
  get_tracerer_path("beast2_example_output.log")
)

# Calculate the effective sample sizes of all parameter estimates
calc_esses(estimates, sample_interval = 1000)
```

---

calc_geom_mean	<i>Calculate the geometric mean</i>
----------------	-------------------------------------

---

**Description**

Calculate the geometric mean

**Usage**

```
calc_geom_mean(values)
```

**Arguments**

values            a numeric vector of values

**Value**

returns the geometric mean if all values are at least zero, else returns NA

**Author(s)**

Richèl J.C. Bilderbeek

---

calc\_hpd\_interval        *Calculate the Highest Probability Density of an MCMC trace that has its burn-in removed*

---

**Description**

Calculate the Highest Probability Density of an MCMC trace that has its burn-in removed

**Usage**

```
calc_hpd_interval(trace, proportion = 0.95)
```

**Arguments**

trace            a numeric vector of parameter estimates obtained from an MCMC run. Must have its burn-in removed

proportion       the proportion of numbers within the interval. For example, use 0.95 for a 95 percentage interval

**Value**

a numeric vector, with at index 1 the lower boundary of the interval, and at index 2 the upper boundary of the interval

**Author(s)**

The original Java version of the algorithm was from J. Heled, ported to R and adapted by Richèl J.C. Bilderbeek

**See Also**

The function [remove\\_burn\\_in](#) removes a burn-in. The Java code that inspired this function can be found here: <https://github.com/beast-dev/beast-mcmc/blob/98705c59db65e4f406a420bbade949aeecfe05d0/src/dr/stats/DiscreteStatistics.java#L317> # nolint URLs can be long

### Examples

```
estimates <- parse_beast_tracelog_file(
  get_tracerer_path("beast2_example_output.log")
)
tree_height_trace <- remove_burn_in(
  estimates$TreeHeight,
  burn_in_fraction = 0.1
)

# Values will be 0.453 and 1.816
calc_hpd_interval(tree_height_trace, proportion = 0.95)
```

---

calc_mode	<i>Calculate the mode of values If the distribution is bi or multimodal or uniform, NA is returned</i>
-----------	--

---

### Description

Calculate the mode of values If the distribution is bi or multimodal or uniform, NA is returned

### Usage

```
calc_mode(values)
```

### Arguments

values            numeric vector to calculate the mode of

### Value

the mode of the trace

### Author(s)

Richèl J.C. Bilderbeek

### Examples

```
# In a unimodal distribution, find the value that occurs most
calc_mode(c(1, 2, 2))
calc_mode(c(1, 1, 2))

# For a uniform distribution, NA is returned
tracerer:::calc_mode(c(1, 2))
```



---

calc\_stderr\_mean      *Calculate the standard error of the mean*

---

**Description**

Calculate the standard error of the mean

**Usage**

```
calc_stderr_mean(trace)
```

**Arguments**

trace                  the values

**Value**

the standard error of the mean

**Author(s)**

The original Java version of the algorithm was from Remco Bouckaert, ported to R and adapted by Richèl J.C. Bilderbeek

**See Also**

Java code can be found here: <https://github.com/beast-dev/beast-mcmc/blob/800817772033c13061f026226e41128src/dr/inference/trace/TraceCorrelation.java#L159> # nolint URLs can be long

**Examples**

```
trace <- sin(seq(from = 0.0, to = 2.0 * pi, length.out = 100))
calc_stderr_mean(trace) # 0.4347425
```

---

calc\_std\_error\_of\_mean\_cpp  
*Calculates the standard error of the mean*

---

**Description**

Calculates the standard error of the mean

**Usage**

```
calc_std_error_of_mean_cpp(sample)
```

**Arguments**

sample            numeric vector of values

**Value**

the standard error of the mean

**Author(s)**

Richèl J.C. Bilderbeek

---

calc\_summary\_stats    *Calculates the Effective Sample Sizes of one estimated variable's trace.*

---

**Description**

Calculates the Effective Sample Sizes of one estimated variable's trace.

**Usage**

```
calc_summary_stats(traces, sample_interval)
```

**Arguments**

traces            one or more traces, supplies as either, (1) a numeric vector or, (2) a data frame of numeric values.

sample\_interval    the interval (the number of state transitions between samples) of the MCMC run that produced the trace. Using a different sample\_interval than the actually used sampling interval will result in bogus return values.

**Value**

the summary statistics of the traces. If one numeric vector is supplied, a list is returned with the elements listed below. If the traces are supplied as a data frame, a data frame is returned with the elements listed below as column names.

The elements are:

- mean: mean
- stderr\_mean: standard error of the mean
- stdev: standard deviation
- variance: variance
- mode: mode
- geom\_mean: geometric mean

- hpd\_interval\_low: lower bound of 95% highest posterior density
- hpd\_interval\_high: upper bound of 95% highest posterior density
- act: auto correlation time
- ess: effective sample size

**Note**

This function assumes the burn-in is removed. Use [remove\\_burn\\_in](#) (on a vector) or [remove\\_burn\\_ins](#) (on a data frame) to remove the burn-in.

**Author(s)**

Richèl J.C. Bilderbeek

**See Also**

Use [calc\\_summary\\_stats\\_trace](#) to calculate the summary statistics of one trace (stored as a numeric vector). Use [calc\\_summary\\_stats\\_traces](#) to calculate the summary statistics of more traces (stored as a data frame).

**Examples**

```
estimates_all <- parse_beast_tracelog_file(
  get_tracerer_path("beast2_example_output.log")
)
estimates <- remove_burn_ins(estimates_all, burn_in_fraction = 0.1)

# From a single variable's trace
calc_summary_stats(
  estimates$posterior,
  sample_interval = 1000
)

# From all variables' traces
calc_summary_stats(
  estimates,
  sample_interval = 1000
)
```

---

calc\_summary\_stats\_trace

*Calculates the Effective Sample Sizes of one estimated variable's trace.*

---

**Description**

Calculates the Effective Sample Sizes of one estimated variable's trace.

**Usage**

```
calc_summary_stats_trace(trace, sample_interval)
```

**Arguments**

`trace` a numeric vector of values. Assumes the burn-in is removed.  
`sample_interval` the interval in timesteps between samples

**Value**

the effective sample sizes

**Author(s)**

Richèl J.C. Bilderbeek

**See Also**

Use [remove\\_burn\\_in](#) to remove the burn-in of a trace

**Examples**

```
estimates_all <- parse_beast_tracelog_file(  
  get_tracerer_path("beast2_example_output.log")  
)  
estimates <- remove_burn_ins(estimates_all, burn_in_fraction = 0.1)  
  
calc_summary_stats_trace(  
  estimates$posterior,  
  sample_interval = 1000  
)
```

---

calc\_summary\_stats\_traces

*Calculates the Effective Sample Sizes of the traces of multiple estimated variables.*

---

**Description**

Calculates the Effective Sample Sizes of the traces of multiple estimated variables.

**Usage**

```
calc_summary_stats_traces(traces, sample_interval)
```

**Arguments**

traces                    a data frame with traces of estimated parameters. Assumes the burn-ins are removed.

sample\_interval            the interval in timesteps between samples

**Value**

the effective sample sizes

**Author(s)**

Richèl J.C. Bilderbeek

**See Also**

Use [remove\\_burn\\_ins](#) to remove the burn-ins of all traces

**Examples**

```
estimates_all <- parse_beast_tracelog_file(
  get_tracerer_path("beast2_example_output.log")
)
estimates <- remove_burn_ins(estimates_all, burn_in_fraction = 0.1)

calc_summary_stats_traces(
  estimates,
  sample_interval = 1000
)
```

---

check\_trace

*Check if the trace is a valid. Will [stop](#) if not*

---

**Description**

Check if the trace is a valid. Will [stop](#) if not

**Usage**

```
check_trace(trace)
```

**Arguments**

trace                    the values

**Author(s)**

Richèl J.C. Bilderbeek

**Examples**

```
check_trace(seq(1, 2))
```

---

```
count_trees_in_file    Count the number of trees in a .trees file
```

---

**Description**

Count the number of trees in a .trees file

**Usage**

```
count_trees_in_file(trees_filename)
```

**Arguments**

trees\_filename name of a BEAST2 posterior .trees file, as can be read using [parse\\_beast\\_trees](#)

**Value**

the number of trees

**Author(s)**

Richèl J.C. Bilderbeek

**See Also**

if the .trees file is invalid, use [is\\_trees\\_file](#) with verbose = TRUE for the reason

---

```
cs_std_dev            Calculate the corrected sample standard deviation.
```

---

**Description**

Calculate the corrected sample standard deviation.

**Usage**

```
cs_std_dev(values)
```

**Arguments**

values numeric values

**Value**

the corrected sample standard deviation

**Author(s)**

Richèl J.C. Bilderbeek

---

default_params_doc	<i>Documentation of general function arguments. This function does nothing. It is intended to inherit function argument documentation.</i>
--------------------	--

---

**Description**

Documentation of general function arguments. This function does nothing. It is intended to inherit function argument documentation.

**Usage**

```
default_params_doc(
  log_filename,
  sample_interval,
  state_filename,
  trace,
  tracelog_filename,
  trees_filename,
  trees_filenames,
  verbose
)
```

**Arguments**

log_filename	deprecated name of the BEAST2 tracelog .log output file. Use tracelog_filename instead
sample_interval	the interval in timesteps between samples
state_filename	name of the BEAST2 state .xml.state output file
trace	the values
tracelog_filename	name of the BEAST2 tracelog .log output file, as can be read using <a href="#">parse_beast_tracelog_file</a>
trees_filename	name of a BEAST2 posterior .trees file, as can be read using <a href="#">parse_beast_trees</a>
trees_filenames	the names of one or more a BEAST2 posterior .trees file. Each .trees file can be read using <a href="#">parse_beast_trees</a>
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

---

extract\_operators\_lines

*Extract the JSON lines out of a .xml.state with the unparsed BEAST2 MCMC operator acceptances file with the operators*

---

**Description**

Extract the JSON lines out of a .xml.state with the unparsed BEAST2 MCMC operator acceptances file with the operators

**Usage**

```
extract_operators_lines(filename)
```

**Arguments**

filename            name of the BEAST2 .xml.state output file

**Value**

the JSON lines of a .xml.state file with the unparsed BEAST2 MCMC operator acceptances

**Author(s)**

Richèl J.C. Bilderbeek

---

get\_tracerer\_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_tracerer_path(filename)
```



**Arguments**

filename            the file's name, without the path

**Value**

the full path to the filename

**Author(s)**

Richèl J.C. Bilderbeek

**See Also**

for more files, use [get\\_tracerer\\_paths](#)

**Examples**

```
get_tracerer_path("beast2_example_output.log")
get_tracerer_path("beast2_example_output.trees")
get_tracerer_path("beast2_example_output.xml")
get_tracerer_path("beast2_example_output.xml.state")
```

---

`get_tracerer_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_tracerer_paths(filenamees)
```

**Arguments**

filenamees            the files' names, without the path

**Value**

the filenamees' full paths

**Author(s)**

Richèl J.C. Bilderbeek

**See Also**

for one file, use [get\\_tracerer\\_path](#)

## Examples

```
get_tracerer_paths(  
  c(  
    "beast2_example_output.log",  
    "beast2_example_output.trees",  
    "beast2_example_output.xml",  
    "beast2_example_output.xml.state"  
  )  
)
```

---

get\_tracerer\_tempfilename

*Get a temporary filename*

---

## Description

Get a temporary filename, similar to [tempfile](#), except that it always writes to a temporary folder named [tracerer](#).

## Usage

```
get_tracerer_tempfilename(pattern = "file", fileext = "")
```

## Arguments

pattern	a non-empty character vector giving the initial part of the name.
fileext	a non-empty character vector giving the file extension

## Value

name for a temporary file

## Note

this function is added to make sure no temporary cache files are left undeleted

---

is_posterior	<i>Determines if the input is a BEAST2 posterior</i>
--------------	--

---

**Description**

Determines if the input is a BEAST2 posterior

**Usage**

```
is_posterior(x)
```

**Arguments**

x                    the input

**Value**

TRUE if the input contains all information of a BEAST2 posterior. Returns FALSE otherwise.

**Author(s)**

Richèl J.C. Bilderbeek

**Examples**

```
trees_filename <- get_tracerer_path("beast2_example_output.trees")
tracelog_filename <- get_tracerer_path("beast2_example_output.log")
posterior <- parse_beast_posterior(
  trees_filename = trees_filename,
  tracelog_filename = tracelog_filename
)
is_posterior(posterior)
```

---

is_trees_file	<i>Measure if a file a valid BEAST2 .trees file</i>
---------------	---

---

**Description**

Measure if a file a valid BEAST2 .trees file

**Usage**

```
is_trees_file(trees_filename, verbose = FALSE)
```

**Arguments**

trees\_filename    name of a BEAST2 posterior .trees file, as can be read using [parse\\_beast\\_trees](#)  
 verbose            set to TRUE for more output

**Value**

TRUE if trees\_filename is a valid .trees file

**Author(s)**

Richèl J.C. Bilderbeek

**See Also**

Most of the work is done by [read.nexus](#)

**Examples**

```
# TRUE
is_trees_file(get_tracerer_path("beast2_example_output.trees"))
is_trees_file(get_tracerer_path("unplottable_anthus_aco.trees"))
is_trees_file(get_tracerer_path("anthus_2_4_a.trees"))
is_trees_file(get_tracerer_path("anthus_2_4_b.trees"))
# FALSE
is_trees_file(get_tracerer_path("mcbette_issue_8.trees"))
```

---

is_trees_posterior	<i>Determines if the input is a BEAST2 posterior, as parsed by parse_beast_trees</i>
--------------------	--

---

**Description**

Determines if the input is a BEAST2 posterior, as parsed by parse\_beast\_trees

**Usage**

```
is_trees_posterior(x)
```

**Arguments**

x                    the input

**Value**

TRUE or FALSE

**Author(s)**

Richèl J.C. Bilderbeek

---

parse_beast_log	<i>Deprecated function to parse a BEAST2 .log output file. Use <a href="#">parse_beast_tracelog_file</a> instead</i>
-----------------	--

---

**Description**

Deprecated function to parse a BEAST2 .log output file. Use [parse\\_beast\\_tracelog\\_file](#) instead

**Usage**

```
parse_beast_log(tracelog_filename, filename = "deprecated")
```

**Arguments**

tracelog_filename	name of the BEAST2 tracelog .log output file, as can be read using <a href="#">parse_beast_tracelog_file</a>
filename	deprecated name of the BEAST2 .log output file

**Value**

data frame with the parameter estimates

**Author(s)**

Richèl J.C. Bilderbeek

**Examples**

```
# Deprecated
parse_beast_log(
  tracelog_filename = get_tracerer_path("beast2_example_output.log")
)
# Use the function 'parse_beast_tracelog_file' instead
parse_beast_tracelog_file(
  tracelog_filename = get_tracerer_path("beast2_example_output.log")
)
```

---

parse_beast_output_files
--------------------------

---

*Parse all BEAST2 output files*

---

**Description**

Parse all BEAST2 output files

**Usage**

```
parse_beast_output_files(log_filename, trees_filenames, state_filename)
```

**Arguments**

log\_filename    deprecated name of the BEAST2 tracelog .log output file. Use tracelog\_filename instead

trees\_filenames    the names of one or more a BEAST2 posterior .trees file. Each .trees file can be read using [parse\\_beast\\_trees](#)

state\_filename    name of the BEAST2 state .xml .state output file

**Value**

a list with the following elements:

itemestimates: parameter estimates item [alignment\_id]\_trees: the phylogenies in the BEAST2 posterior. [alignment\_id] is the ID of the alignment.

itemoperators: the BEAST2 MCMC operator acceptances

**Author(s)**

Richèl J.C. Bilderbeek

**See Also**

Use [remove\\_burn\\_ins](#) to remove the burn-in from out\$estimates

**Examples**

```
trees_filenames <- get_tracerer_path("beast2_example_output.trees")
log_filename <- get_tracerer_path("beast2_example_output.log")
state_filename <- get_tracerer_path("beast2_example_output.xml.state")
parse_beast_output_files(
  log_filename = log_filename,
  trees_filenames = trees_filenames,
  state_filename = state_filename
)
```

---

parse\_beast\_posterior *Parses BEAST2 output files to a posterior*

---

**Description**

Parses BEAST2 output files to a posterior

**Usage**

```
parse_beast_posterior(  
  trees_filenames,  
  tracelog_filename,  
  log_filename = "deprecated"  
)
```

**Arguments**

`trees_filenames` the names of one or more a BEAST2 posterior .trees file. Each .trees file can be read using [parse\\_beast\\_trees](#)

`tracelog_filename` name of the BEAST2 tracelog .log output file, as can be read using [parse\\_beast\\_tracelog\\_file](#)

`log_filename` deprecated name of the BEAST2 tracelog .log output file. Use `tracelog_filename` instead

**Value**

a list with the following elements:

`itemestimates`: parameter estimates item `[alignment_id]_trees`: the phylogenies in the BEAST2 posterior. `[alignment_id]` is the ID of the alignment.

**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**

```
trees_filenames <- get_tracerer_path("beast2_example_output.trees")  
tracelog_filename <- get_tracerer_path("beast2_example_output.log")  
posterior <- parse_beast_posterior(  
  trees_filenames = trees_filenames,  
  tracelog_filename = tracelog_filename  
)
```

---

parse\_beast\_state\_operators

*Parses a BEAST2 state .xml.state output file to get only the operators acceptances*

---

### Description

Parses a BEAST2 state .xml.state output file to get only the operators acceptances

### Usage

```
parse_beast_state_operators(  
  state_filename = get_tracerer_path("beast2_example_output.xml.state"),  
  filename = "deprecated"  
)
```

### Arguments

state\_filename name of the BEAST2 state .xml.state output file  
filename deprecated name of the BEAST2 .xml.state output file, use state\_filename instead

### Value

data frame with all the operators' success rates

### Author(s)

Richèl J.C. Bilderbeek

### Examples

```
parse_beast_state_operators(  
  state_filename = get_tracerer_path("beast2_example_output.xml.state")  
)
```

---

parse\_beast\_tracelog\_file

*Parses a BEAST2 tracelog .log output file*

---

### Description

Parses a BEAST2 tracelog .log output file



**Usage**

```
parse_beast_tracelog_file(tracelog_filename)
```

**Arguments**

```
tracelog_filename  
                    name of the BEAST2 tracelog .log output file, as can be read using parse\_beast\_tracelog\_file
```

**Value**

data frame with the parameter estimates

**Author(s)**

Richèl J.C. Bilderbeek

**See Also**

Use [remove\\_burn\\_ins](#) to remove the burn-in from the returned parameter estimates. Use [save\\_beast\\_estimates](#) to save the estimates to a .log file.

**Examples**

```
parse_beast_tracelog_file(  
  tracelog_filename = get_tracerer_path("beast2_example_output.log")  
)
```

---

parse_beast_trees	<i>Parses a BEAST2 .trees output file</i>
-------------------	---

---

**Description**

Parses a BEAST2 .trees output file

**Usage**

```
parse_beast_trees(filename)
```

**Arguments**

```
filename           name of the BEAST2 .trees output file
```

**Value**

the phylogenies in the posterior

**Author(s)**

Richèl J.C. Bilderbeek

**See Also**

Use [save\\_beast\\_trees](#) to save the phylogenies to a .trees file. Use [is\\_trees\\_file](#) with verbose = TRUE to find out why a file is invalid

**Examples**

```
trees_filename <- get_tracerer_path("beast2_example_output.trees")
parse_beast_trees(trees_filename)
```

---

remove_burn_in	<i>Removed the burn-in from a trace</i>
----------------	---

---

**Description**

Removed the burn-in from a trace

**Usage**

```
remove_burn_in(trace, burn_in_fraction)
```

**Arguments**

trace                   the values  
burn\_in\_fraction       the fraction that needs to be removed, must be [0,1>

**Value**

the values with the burn-in removed

**Author(s)**

Richèl J.C. Bilderbeek

**Examples**

```
# Create a trace from one to and including ten
v <- seq(1, 10)

# Remove the first ten percent of its values,
# in this case removes the first value, which is one
w <- remove_burn_in(trace = v, burn_in_fraction = 0.1)
```

---

remove_burn_ins	<i>Removed the burn-ins from a data frame</i>
-----------------	---

---

**Description**

Removed the burn-ins from a data frame

**Usage**

```
remove_burn_ins(traces, burn_in_fraction = 0.1)
```

**Arguments**

traces	a data frame with traces
burn_in_fraction	the fraction that needs to be removed, must be $[0, 1[$ . Its default value of 10 as of Tracer

**Value**

the data frame with the burn-in removed

**Author(s)**

Richèl J.C. Bilderbeek

---

save_beast_estimates	<i>Save the BEAST2 estimates as a BEAST2 .log file. There will be some differences: a BEAST2 .log file also saves the model as comments and formats the numbers in a way non-standard to R</i>
----------------------	--

---

**Description**

Save the BEAST2 estimates as a BEAST2 .log file. There will be some differences: a BEAST2 .log file also saves the model as comments and formats the numbers in a way non-standard to R

**Usage**

```
save_beast_estimates(estimates, filename)
```

**Arguments**

estimates	a data frame of BEAST2 parameter estimates
filename	name of the .log file to save to

**Value**

nothing

**Author(s)**

Richèl J.C. Bilderbeek

**See Also**

Use [parse\\_beast\\_log](#) to read a BEAST2 .log file

---

save_beast_trees	<i>Save the BEAST2 trees as a BEAST2 .log file. There will be some differences: a BEAST2 .log file also saves the model as comments and formats the numbers in a way non-standard to R</i>
------------------	--

---

**Description**

Save the BEAST2 trees as a BEAST2 .log file. There will be some differences: a BEAST2 .log file also saves the model as comments and formats the numbers in a way non-standard to R

**Usage**

```
save_beast_trees(trees, filename)
```

**Arguments**

trees	BEAST2 posterior trees, of type <code>ape::multiPhylo</code>
filename	name of the .trees file to save to

**Value**

nothing

**Author(s)**

Richèl J.C. Bilderbeek

**See Also**

Use [parse\\_beast\\_log](#) to read a BEAST2 .log file

# Index

[calc\\_act](#), [2](#), [4](#)  
[calc\\_act\\_cpp](#), [3](#)  
[calc\\_act\\_r](#), [4](#)  
[calc\\_ess](#), [5](#)  
[calc\\_esses](#), [6](#)  
[calc\\_geom\\_mean](#), [6](#)  
[calc\\_hpd\\_interval](#), [7](#)  
[calc\\_mode](#), [8](#)  
[calc\\_std\\_error\\_of\\_mean\\_cpp](#), [9](#)  
[calc\\_stderr\\_mean](#), [9](#)  
[calc\\_summary\\_stats](#), [10](#)  
[calc\\_summary\\_stats\\_trace](#), [11](#), [11](#)  
[calc\\_summary\\_stats\\_traces](#), [11](#), [12](#)  
[check\\_trace](#), [13](#)  
[count\\_trees\\_in\\_file](#), [14](#)  
[cs\\_std\\_dev](#), [14](#)

[default\\_params\\_doc](#), [15](#)

[extract\\_operators\\_lines](#), [16](#)

[get\\_tracerer\\_path](#), [16](#), [17](#)  
[get\\_tracerer\\_paths](#), [17](#), [17](#)  
[get\\_tracerer\\_tempfilename](#), [18](#)

[is\\_posterior](#), [19](#)  
[is\\_trees\\_file](#), [14](#), [19](#), [26](#)  
[is\\_trees\\_posterior](#), [20](#)

[parse\\_beast\\_log](#), [21](#), [28](#)  
[parse\\_beast\\_output\\_files](#), [21](#)  
[parse\\_beast\\_posterior](#), [22](#)  
[parse\\_beast\\_state\\_operators](#), [24](#)  
[parse\\_beast\\_tracelog\\_file](#), [15](#), [21](#), [23](#), [24](#),  
[25](#)  
[parse\\_beast\\_trees](#), [14](#), [15](#), [19](#), [22](#), [23](#), [25](#)

[read.nexus](#), [20](#)  
[remove\\_burn\\_in](#), [7](#), [11](#), [12](#), [26](#)  
[remove\\_burn\\_ins](#), [11](#), [13](#), [22](#), [23](#), [25](#), [27](#)

[save\\_beast\\_estimates](#), [25](#), [27](#)  
[save\\_beast\\_trees](#), [26](#), [28](#)  
[stop](#), [13](#)

[tempfile](#), [18](#)  
[tracerer](#), [18](#)