

# Package ‘o2ools’

July 22, 2025

**Title** Tools for 'outbreaker2'

**Version** 0.0.1

**Description** Streamlines the post-processing, summarization, and visualization of 'outbreaker2' output via a suite of helper functions. Facilitates tidy manipulation of posterior samples, integration with case metadata, generation of diagnostic plots and summary statistics.

**License** MIT + file LICENSE

**Encoding** UTF-8

**RoxygenNote** 7.3.2

**Suggests** knitr, rmarkdown, testthat (>= 3.0.0), outbreaker2, incidence2, mixtree, epiccontacts, dplyr, tidyr, furr, ggplot2, igraph, tidygraph, ggraph

**Config/testthat/edition** 3

**Depends** R (>= 3.5)

**LazyData** true

**VignetteBuilder** knitr

**URL** <https://github.com/CyGei/o2ools>, <https://cygei.github.io/o2ools/>

**BugReports** <https://github.com/CyGei/o2ools/issues>

**NeedsCompilation** no

**Author** Cyril Geismar [aut, cre]

**Maintainer** Cyril Geismar <c.geismar21@imperial.ac.uk>

**Repository** CRAN

**Date/Publication** 2025-06-06 13:00:06 UTC

## Contents

|                            |   |
|----------------------------|---|
| augment_linelist . . . . . | 2 |
| filter_chain . . . . .     | 3 |
| get_accuracy . . . . .     | 4 |
| get_consensus . . . . .    | 4 |

|                                    |           |
|------------------------------------|-----------|
| get_entropy . . . . .              | 5         |
| get_Ri . . . . .                   | 6         |
| get_si . . . . .                   | 7         |
| get_trees . . . . .                | 8         |
| identify . . . . .                 | 8         |
| linelist . . . . .                 | 9         |
| out . . . . .                      | 10        |
| sample.outbreaker_chains . . . . . | 10        |
| ttable . . . . .                   | 11        |
| <b>Index</b>                       | <b>12</b> |

---

|                  |                                                             |
|------------------|-------------------------------------------------------------|
| augment_linelist | <i>Append summaries of outbreaker results to a linelist</i> |
|------------------|-------------------------------------------------------------|

---

**Description**

For each case in linelist, appends summary statistics of selected parameters from an outbreaker\_chains object (e.g. infection times, number of generations).

**Usage**

```
augment_linelist(  
  out,  
  linelist,  
  params = c("t_inf", "kappa"),  
  summary_fns = list(mean = function(x) mean(x, na.rm = TRUE), q25 = function(x)  
    quantile(x, 0.25, na.rm = TRUE), q75 = function(x) quantile(x, 0.75, na.rm = TRUE))  
)
```

**Arguments**

- out                    An outbreaker\_chains object containing posterior samples.
- linelist              A data.frame with an id column matching the IDs in out.
- params                Character vector of parameter prefixes to summarise (e.g. "t\_inf", "kappa").
- summary\_fns          A **named** list of summary functions. Each function takes a numeric vector and returns a single value. Example: list( mean = function(x) mean(x, na.rm = TRUE), q25 = function(x) quantile(x, 0.25, na.rm = TRUE), q75 = function(x) quantile(x, 0.75, na.rm = TRUE) )

**Value**

The input linelist, with new columns named <param>\_<fn> (e.g. t\_inf\_mean, kappa\_q25).

**Examples**

```
augmented_linelist <- augment_linelist(
  out, linelist,
  params = c("t_inf", "kappa"),
  summary_fns = list(
    median = function(x) median(x, na.rm = TRUE),
    q25 = function(x) quantile(x, 0.25, na.rm = TRUE),
    q75 = function(x) quantile(x, 0.75, na.rm = TRUE)
  )
)
```

filter\_chain

*Filter chain by parameter threshold***Description**

Mask target columns whenever a parameter column fails a threshold test.

**Usage**

```
filter_chain(out, param, thresh, comparator = "<=", target = "alpha")
```

**Arguments**

|            |                                                         |
|------------|---------------------------------------------------------|
| out        | A data frame of class <code>outbreaker_chains</code> .  |
| param      | Name of the parameter prefix (e.g. "kappa").            |
| thresh     | Numeric threshold.                                      |
| comparator | A string comparator: one of "<=", ">=", "<", ">", "==". |
| target     | Name of the target prefix to mask (e.g. "alpha").       |

**Value**

An `outbreaker_chains` data frame with `target_*` entries set to NA wherever `param_*` comparator thresh is FALSE.

**Examples**

```
# Mask alpha_i whenever kappa_i > 1
filter_chain(out, param = "kappa", thresh = 1, comparator = "<=", target = "alpha")
```

---

|              |                                                          |
|--------------|----------------------------------------------------------|
| get_accuracy | <i>Calculate the accuracy of outbreak reconstruction</i> |
|--------------|----------------------------------------------------------|

---

**Description**

Accuracy is defined as the proportion of correctly assigned ancestries across the posterior sample.

**Usage**

```
get_accuracy(out, true_tree)
```

**Arguments**

|           |                                                                                  |
|-----------|----------------------------------------------------------------------------------|
| out       | An object of class outbreaker_chains.                                            |
| true_tree | A data frame with the true transmission tree, including 'from' and 'to' columns. |

**Value**

A numeric vector of accuracy values for each posterior tree.

**Examples**

```
true_tree <- data.frame(from = as.character(outbreaker2::fake_outbreak$ances), to = linelist$id)
get_accuracy(out, true_tree)
```

---

|               |                                            |
|---------------|--------------------------------------------|
| get_consensus | <i>Get the consensus transmission tree</i> |
|---------------|--------------------------------------------|

---

**Description**

Computes the most frequent ancestor for each case across the posterior sample.

**Usage**

```
get_consensus(out)
```

**Arguments**

|     |                                      |
|-----|--------------------------------------|
| out | An object of class outbreaker_chains |
|-----|--------------------------------------|

**Value**

A data frame showing the most frequent ancestor for each case.

**Examples**

```
get_consensus(out)
```

get\_entropy

*Compute the entropy of transmission trees***Description**

Computes the mean entropy of transmission trees from outbreaker2, quantifying uncertainty in inferred infectors. By default, entropy is normalised between 0 (complete certainty) and 1 (maximum uncertainty).

**Usage**

```
get_entropy(out, normalise = TRUE)
```

**Arguments**

|           |                                                                                                           |
|-----------|-----------------------------------------------------------------------------------------------------------|
| out       | A data frame of class outbreaker_chains containing posterior samples of transmission ancestries (alpha).  |
| normalise | Logical. If TRUE (default), entropy is normalised between 0 and 1. If FALSE, returns raw Shannon entropy. |

**Details**

Entropy quantifies uncertainty in inferred infectors across posterior samples using the Shannon entropy formula:

$$H(X) = - \sum p_i \log(p_i)$$

where  $p_i$  is the proportion of times each infector is inferred. If `normalise = TRUE`, entropy is scaled by its maximum possible value,  $\log(K)$ , where  $K$  is the number of distinct inferred infectors:

$$H^*(X) = \frac{H(X)}{\log(K)}$$

This normalisation ensures values range from 0 to 1:

- **0:** Complete certainty — the same infector is inferred across all samples.
- **1:** Maximum uncertainty — all infectors are equally likely.

**Value**

A numeric value representing the mean entropy of transmission trees across posterior samples.

## Examples

```
# High entropy
out <- data.frame(alpha_1 = sample(c("2", "3"), 100, replace = TRUE),
                  alpha_2 = sample(c("1", "3"), 100, replace = TRUE))
class(out) <- c("outbreaker_chains", class(out))
get_entropy(out)

# Low entropy
out <- data.frame(alpha_1 = sample(c("2", "3"), 100, replace = TRUE, prob = c(0.95, 0.05)),
                  alpha_2 = sample(c("1", "3"), 100, replace = TRUE, prob = c(0.95, 0.05)))
class(out) <- c("outbreaker_chains", class(out))
get_entropy(out)
```

---

get\_Ri

---

*Compute case reproduction numbers (Ri) from outbreaker2 chains*


---

## Description

This function computes the number of secondary infections caused by each individual from outbreaker2 MCMC chains. For each MCMC iteration, it counts how many times each individual appears as an infector (alpha parameter).

## Usage

```
get_Ri(out)
```

## Arguments

out                    An object of class outbreaker\_chains

## Value

A data frame where:

- Each row represents an MCMC iteration
- Each column represents an individual (named by their identifier)
- Values represent the reproduction number (Ri) for that individual in that iteration

## Examples

```
out_id <- identify(out, ids = linelist$name)
Ri <- get_Ri(out_id)
str(Ri)
```

get\_si

*Compute the serial interval (si) from transmission trees***Description**

The serial interval is the time between the onset of symptoms in an infector-infectee pair. This function computes the serial interval statistics from a list of transmission trees.

**Usage**

```
get_si(
  trees,
  date_suffix = "date",
  stats = list(mean = mean, lwr = function(x) quantile(x, 0.025, na.rm = TRUE), upr =
    function(x) quantile(x, 0.975, na.rm = TRUE))
)
```

**Arguments**

- |             |                                                                                                                                                                                                                                                                                                                                                                |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| trees       | A list of data frames, generated by <a href="#">get_trees</a> . It should contain information about the dates of onset.                                                                                                                                                                                                                                        |
| date_suffix | A string indicating the suffix for date of onset columns. Default is "date", which means the columns should be named from_date and to_date.                                                                                                                                                                                                                    |
| stats       | <p>A list of functions to compute statistics. Default is:</p> <ul style="list-style-type: none"> <li>• mean: the mean serial interval.</li> <li>• lwr: the 2.5th percentile (lower quantile).</li> <li>• upr: the 97.5th percentile (upper quantile).</li> </ul> <p>Each function should take a numeric vector as input and return a single numeric value.</p> |

**Value**

A data frame with serial interval statistic

**See Also**

[get\\_trees](#) for generating a list of transmission trees.

**Examples**

```
trees <- get_trees(out, date = linelist$onset)
si_stats <- get_si(trees)
str(si_stats)
```

---

|           |                                             |
|-----------|---------------------------------------------|
| get_trees | <i>Extract posterior transmission trees</i> |
|-----------|---------------------------------------------|

---

### Description

Generates a list of data frames representing posterior transmission trees from an `outbreaker_chains` object. Each tree contains 'from' and 'to' columns, and may optionally include `kappa`, `t_inf`, and user-supplied columns.

### Usage

```
get_trees(out, kappa = FALSE, t_inf = FALSE, ...)
```

### Arguments

|                    |                                                                                                                                |
|--------------------|--------------------------------------------------------------------------------------------------------------------------------|
| <code>out</code>   | A data frame of class <code>outbreaker_chains</code> .                                                                         |
| <code>kappa</code> | Logical. If TRUE, includes kappa values in the output. Default is FALSE.                                                       |
| <code>t_inf</code> | Logical. If TRUE, includes infection times ( <code>t_inf</code> ) in the output. Default is FALSE.                             |
| <code>...</code>   | Additional vectors to include as columns in the output. Must be given in the same order as used in <code>outbreaker()</code> . |

### Value

A list of data frames, one per posterior sample. Each data frame has at least 'from' and 'to' columns.

### Examples

```
get_trees(out, id = linelist$id,
          name = linelist$name,
          group = linelist$group,
          onset = linelist$onset)
```

---

|          |                                                                       |
|----------|-----------------------------------------------------------------------|
| identify | <i>Replace integers in outbreaker2 output with unique identifiers</i> |
|----------|-----------------------------------------------------------------------|

---

### Description

Replace integers in `outbreaker2` output with unique identifiers

### Usage

```
identify(out, ids)
```



**Arguments**

**out** A data frame of class `outbreaker_chains`.  
**ids** A vector of IDs from the original linelist (see `outbreaker2::outbreaker_data()`).

**Value**

A data frame of class `outbreaker_chains` with integers replaced by the corresponding IDs.

**Examples**

```
identify(out, id = linelist$name)
```

---

|          |                                             |
|----------|---------------------------------------------|
| linelist | <i>Simulated linelist with group labels</i> |
|----------|---------------------------------------------|

---

**Description**

A simulated linelist derived from `fake_outbreak`, where cases are assigned to the patient or hcw group. First names are randomly generated using the **randomNames** package.

**Usage**

```
linelist
```

**Format**

A data frame with 30 rows and 5 columns:

**id** Case ID  
**name** Simulated first name  
**group** Group label: "patient" or "hcw"  
**onset** Date of symptom onset  
**sample** Date of sample collection

**See Also**

[fake\\_outbreak](#)

**Examples**

```
head(linelist)
```

---

|     |                                |
|-----|--------------------------------|
| out | <i>outbreaker2 toy dataset</i> |
|-----|--------------------------------|

---

**Description**

The outbreaker2 result generated from the example in the [outbreaker2 vignette](#). This dataset was produced by running `outbreaker()` on the `fake_outbreak` data.

**Usage**

out

**Format**

An `outbreaker_chains` object.

**Source**

<https://www.repidemicsconsortium.org/outbreaker2/articles/introduction.html>

---

|                          |                                                     |
|--------------------------|-----------------------------------------------------|
| sample.outbreaker_chains | <i>Sample rows from an outbreaker_chains object</i> |
|--------------------------|-----------------------------------------------------|

---

**Description**

This function samples rows from an object of class `outbreaker_chains`.

**Usage**

```
sample.outbreaker_chains(out, ...)
```

**Arguments**

|     |                                                                                                                  |
|-----|------------------------------------------------------------------------------------------------------------------|
| out | A data frame of class <code>outbreaker_chains</code> .                                                           |
| ... | Additional arguments to be passed to <code>sample()</code> , such as <code>size</code> or <code>replace</code> . |

**Value**

An object of class `outbreaker_chains`, with sampled rows.

---

|        |                                                   |
|--------|---------------------------------------------------|
| ttable | <i>Compute the transmission contingency table</i> |
|--------|---------------------------------------------------|

---

**Description**

Generates a contingency table based on 'from' (infectior) and 'to' (infectee) vectors.

**Usage**

```
ttable(from, to, levels = NULL, ...)
```

**Arguments**

|        |                                                                                                |
|--------|------------------------------------------------------------------------------------------------|
| from   | A vector of infectors.                                                                         |
| to     | A vector of infectees.                                                                         |
| levels | Optional. A vector of factor levels. Defaults to the unique, sorted values of 'from' and 'to'. |
| ...    | Additional arguments passed to the table function.                                             |

**Value**

A contingency table of infectors (rows) and infectees (columns).

**Examples**

```
from <- c("A", "A", NA, "C", "C", "C")
to <- c("A", "B", "B", "C", "C", "C")
ttable(from, to)
```

# Index

## \* **datasets**

linelist, [9](#)

out, [10](#)

augment\_linelist, [2](#)

fake\_outbreak, [9](#)

filter\_chain, [3](#)

get\_accuracy, [4](#)

get\_consensus, [4](#)

get\_entropy, [5](#)

get\_Ri, [6](#)

get\_si, [7](#)

get\_trees, [7](#), [8](#)

identify, [8](#)

linelist, [9](#)

out, [10](#)

sample.outbreaker\_chains, [10](#)

ttable, [11](#)