Package 'ringbp'

June 15, 2026

Type Package

Title Simulate and Evaluate Targeted Interventions in Infectious Disease Outbreaks

Version 1.0.0

Description Branching process simulation model of infectious disease transmission with flexible parameterisation of epidemiology and targeted interventions, including isolation, contact tracing and quarantine, to reduce transmission, together with functions to evaluate outbreak control. Introduced in Hellewell et al. (2020) <doi:10.1016/S2214-109X(20)30074-7>.

License MIT + file LICENSE

URL https://epiforecasts.io/ringbp/, https://github.com/epiforecasts/ringbp

BugReports https://github.com/epiforecasts/ringbp/issues

Depends R (>= 4.4.0)

Imports checkmate, data.table, sn

Suggests future, future.apply, knitr, rmarkdown, roxyglobals (>= 1.0.0), spelling, testthat (>= 3.0.0), tinyplot

VignetteBuilder knitr

Config/roxyglobals/filename globals.R

Config/roxyglobals/unique FALSE

Config/testthat/edition 3

Encoding UTF-8

Language en-GB

RoxygenNote 7.3.3

NeedsCompilation no

Author Joel Hellewell [aut] (ORCID: <https://orcid.org/0000-0003-2683-0849>), Sam Abbott [aut] (ORCID: <https://orcid.org/0000-0001-8057-8037>), Amy Gimma [aut],

Tim Lucas [aut],
Sebastian Funk [aut],
Adam Kucharski [aut],
Hugo Gruson [ctb] (ORCID: <https://orcid.org/0000-0002-4094-1476>),
Carl A. B. Pearson [aut, rev] (ORCID:
<https://orcid.org/0000-0003-0701-7860>),
Joshua W. Lambert [aut, cre] (ORCID:
<https://orcid.org/0000-0001-5218-3046>)

Maintainer Joshua W. Lambert <joshua.lambert@lshtm.ac.uk>

Repository CRAN

Date/Publication 2026-06-15 13:30:02 UTC

Contents

delay_opts 2
event_prob_opts 4
extinction 5
incubation_to_generation_time 7
intervention_opts 8
offspring_opts 9
outbreak_model 10
outbreak_setup 12
outbreak_step 13
scenario_sim 15
sim_opts 17

Index **18**

`delay_opts` *Create a list of delay distributions to run the* **ringbp** *model*

Description

Create a list of delay distributions to run the **ringbp** model

Usage

```
delay_opts(
  incubation_period,
  onset_to_isolation,
  latent_period = 0,
  onset_to_self_isolation = function(n) rep(Inf, n)
)
```

Arguments

`incubation_period`
: a `function`: a random number generating `function` that samples from incubation period distribution, the `function` accepts a single `integer` argument specifying the number of times to sample the incubation period (i.e. length of the `function` output).

`onset_to_isolation`
: a `function`: a random number generating `function` that accepts a single `integer` argument specifying the length of the `function` output.

`latent_period`
: a non-negative `numeric` scalar: the minimum time between an individual being exposed and becoming infectious. It is a population-wide parameter, with no variability between individuals. It sets the minimum generation time in the model. Default is 0 (i.e. an individual becomes immediately infectious after being infected).

  If `latent_period` is positive then generation times are sampled conditional on `gt >= latent_period` (i.e. left-truncated at `latent_period`). This may reduce the realised proportion of presymptomatic transmission, depending on the `incubation_period` distribution and `presymptomatic_transmission` (in `event_prob_opts()`).

`onset_to_self_isolation`
: a `function`: a random number generating `function` that samples from the onset-to-self-isolation distribution, the `function` accepts a single `integer` argument specifying the length of the `function` output.

  By default `onset_to_self_isolation` is a function that generates `Inf` (i.e. individuals never self-isolate). A different onset-to-self-isolation `function` only needs to be specified if a non-zero value is specified to `symptomatic_self_isolate` in `event_prob_opts()` (which by default is 0). If `onset_to_self_isolation` is specified but `symptomatic_self_isolate` is zero, a warning will be thrown and the `onset_to_self_isolation` will be ignored; if `symptomatic_self_isolate` is non-zero and `onset_to_self_isolation` is an `Inf` generating function, `scenario_sim()` will error.

Value

A `list` with class `<ringbp_delay_opts>`.

Examples

```
delay_opts(
  incubation_period = \(n) rweibull(n = n, shape = 2.32, scale = 6.49),
  onset_to_isolation = \(n) rweibull(n = n, shape = 1.65, scale = 4.28)
)
```

`event_prob_opts` *Create a list of event probabilities to run the* **ringbp** *model*

Description

Create a list of event probabilities to run the **ringbp** model

Usage

```
event_prob_opts(
  asymptomatic,
  presymptomatic_transmission,
  symptomatic_traced,
  symptomatic_self_isolate = 0
)
```

Arguments

`asymptomatic` a `numeric` scalar probability (between 0 and 1 inclusive): proportion of cases that are completely asymptomatic (subclinical)

`presymptomatic_transmission`
a `numeric` scalar probability (between 0 and 1 inclusive): proportion of transmission that occurs before symptom onset.

`symptomatic_traced`
a `numeric` scalar probability (between 0 and 1 inclusive), or a `function` of time returning probabilities in `[0, 1]`: proportion of infectious contacts ascertained by contact tracing.

A scalar is treated as a constant contact-tracing probability over the whole simulation. A `function` accepts a `numeric` vector of times (the contact's exposure time in days since the exposure of the initial cases on day 0) and returns a `numeric` vector of probabilities of the same length, allowing the contact-tracing probability to vary with time. For example, `\(t) ifelse(t < 30, 0, 0.5)` represents a contact-tracing programme that activates on day 30 and ascertains 50% of contacts.

Only contacts whose infector is symptomatic are eligible for tracing (see `outbreak_step()` for how isolation times are assigned).

`symptomatic_self_isolate`
a `numeric` scalar probability (between 0 and 1 inclusive): proportion of cases that self-isolate when they become symptomatic. These individuals do not get tested and do not require a positive test result to enter isolation. Default is 0 (i.e. no infectious individuals self-isolate).

If `symptomatic_self_isolate` is non-zero a random number generating `function` needs to be specified in the `onset_to_self_isolation` argument in the `delay_opts()` function, otherwise the `scenario_sim()` will error.

Value

A list with class <ringbp_event_prob_opts>.

Examples

```
event_prob_opts(
  asymptomatic = 0.1,
  presymptomatic_transmission = 0.5,
  symptomatic_traced = 0.2
)

# time-varying contact tracing ascertainment: programme activates
# on day 30 and ascertains 50%
event_prob_opts(
  asymptomatic = 0.1,
  presymptomatic_transmission = 0.5,
  symptomatic_traced = \(t) ifelse(t < 30, 0, 0.5)
)
```

---

extinction *Outbreak extinction functions*

Description

extinct_prob(): Calculate proportion of runs that have controlled outbreak

detect_extinct(): Calculate whether outbreaks went extinct or not

Usage

```
extinct_prob(scenario, extinction_week = NULL)

detect_extinct(scenario, extinction_week = NULL)
```

Arguments

scenario	a data.table: weekly cases output by scenario_sim()
extinction_week	By default NULL but also accepts a positive integer scalar or integer vector to test if the outbreak has gone extinct (i.e. no new cases) by this week (zero indexed): • By default (NULL) the extinction status is stored in the output of scenario_sim() which is supplied to the scenario argument. If extinction_week is not specified or specified as NULL then the pre-computed extinction status from the outbreak simulation will be used. This is defined as all infectious cases have had the opportunity to transmit but no new cases are generated. • A single integer to test if extinction has occurred by this week. For example, extinction_week = 5 tests whether the outbreak is extinct by week 5 (inclusive) until the end of the outbreak.

- An `integer` vector of length two can be supplied to provide the lower and upper bounds (inclusive) of the week range to test for whether extinction occurred by this window. For example `extinction_week = c(5, 10)` will test whether the outbreak went extinct by week 5 and there we no new cases between weeks 5 and 10 (inclusive).
- An `integer` vector of length *n* can be supplied to provide the weeks to test for whether extinction occurred by this window. For example `extinction_week = 12:16` will test that there are no new infections between 12 and 16 weeks after the initial cases (inclusive). These integer sequences will most likely be contiguous but the function does allow non-contiguous integer sequences.

If extinction occurs before the `extinction_week` window then the outbreak extinction is considered extinct, however, if the extinction occurs within the `extinction_week` window it is not considered extinct. Therefore, using a single `integer` for `extinction_week` and thinking of this as "*has the outbreak gone extinct by week X*".

Details

The data passed to `scenario` has to be produced by `scenario_sim()`. It cannot be produced by `outbreak_model()` as it requires the `sim` column, which is only appended in `scenario_sim()`.

Warning: the output from `scenario_sim()` contains an `cap_cases` attribute which is used by `extinct_prob()` and `detect_extinct()`, therefore if you modify the output of `scenario_sim()` before passing to `extinct_prob()` be careful not to drop the attribute (e.g. from subsetting the `data.table`).

Value

`extinct_prob()`: a single `numeric` with the probability of extinction

`detect_extinct()`: a `data.table`, with two columns `sim` and `extinct`, for a binary classification of whether the outbreak went extinct in each simulation replicate. `1` is an outbreak that went extinct, `0` if not.

Examples

```
res <- scenario_sim(
  n = 10,
  initial_cases = 1,
  offspring = offspring_opts(
    community = \(n) rnbinom(n = n, mu = 2.5, size = 0.16),
    isolated = \(n) rnbinom(n = n, mu = 0.5, size = 1)
  ),
  delays = delay_opts(
    incubation_period = \(n) rweibull(n = n, shape = 2.32, scale = 6.49),
    onset_to_isolation = \(n) rweibull(n = n, shape = 1.65, scale = 4.28)
  ),
  event_probs = event_prob_opts(
    asymptomatic = 0,
    presymptomatic_transmission = 0.5,
    symptomatic_traced = 0.2
```

```
  ),
  interventions = intervention_opts(quarantine = FALSE),
  sim = sim_opts(cap_max_days = 350, cap_cases = 4500)
)

# calculate probability of extinction
extinct_prob(res)

# determine if each outbreak simulation replicate has gone extinct
detect_extinct(res)

# calculate extinction in the last 2 weeks of the simulated outbreak
# (i.e. the penultimate and last week of the outbreak)
extinct_prob(res, extinction_week = max(res$week) - 1)

# calculate extinction as no new cases between weeks 12 and 16 of the outbreak
extinct_prob(res, extinction_week = 12:16)
```

`incubation_to_generation_time` *Convert symptom onset times to generation times*

Description

Samples generation times from a skew-normal distribution based on relative symptom onset times (`symptom_onset_time` - `exposure_time`), ensuring all generation times are at least `latent_period`. The location parameter of the skew-normal distribution is set to the relative symptom onset times.

Usage

```
incubation_to_generation_time(
  symptom_onset_time,
  exposure_time = rep(0, length(symptom_onset_time)),
  alpha,
  latent_period = 0
)
```

Arguments

`symptom_onset_time`	a positive `numeric` vector: symptom onset time(s) of the infector(s) in the case data. The symptom onset times are generated by sampling from the incubation period.
`exposure_time`	a non-negative `numeric` vector: time of exposure of the infector(s) in the case data. Used to convert symptom onset in absolute time to relative time for each infectee. Default is for all exposure times to be 0.
`alpha`	a `numeric` scalar: skew parameter of the skew-normal distribution. Used to model the relationship between incubation period and generation time.

latent_period a non-negative numeric scalar: the minimum time between an individual being exposed and becoming infectious. It is a population-wide parameter, with no variability between individuals. It sets the minimum generation time in the model. Default is 0 (i.e. an individual becomes immediately infectious after being infected).

If latent_period is positive then generation times are sampled conditional on gt >= latent_period (i.e. left-truncated at latent_period). This may reduce the realised proportion of presymptomatic transmission, depending on the incubation_period distribution and presymptomatic_transmission (in event_prob_opts()).

Value

a numeric vector of generation times of equal length to the vector input to symptom_onset_time: the i-th element of the vector contains a sample from the generation time distribution of an individual with incubation period given by the i-th element of the symptom_onset_time vector. The lower bound of the output generation time vector is set by the latent_period, to prevent transmission before becoming infectious.

Examples

```
incubation_to_generation_time(
  symptom_onset_time = c(1, 2, 3, 4, 1),
  alpha = 2
)
```

---

intervention_opts *Create a list of intervention settings to run the* **ringbp** *model*

Description

Create a list of intervention settings to run the **ringbp** model

Usage

```
intervention_opts(quarantine = FALSE, test_sensitivity = 1)
```

Arguments

quarantine a logical scalar: whether quarantine is in effect. If TRUE, traced contacts are isolated when their infector is isolated, regardless of their own symptom status (so they may be isolated before symptom onset, and asymptomatic traced contacts are isolated too). If FALSE, only symptomatic traced contacts are isolated, no earlier than their own symptom onset. Defaults to FALSE

test_sensitivity a numeric scalar probability (between 0 and 1 inclusive), or a function of time returning probabilities in [0, 1]: the test sensitivity (i.e. probability that a true positive tests positive).

A scalar is treated as a constant test sensitivity over the whole simulation. A `function` accepts a `numeric` vector of times (symptom onset time in days since the exposure of the initial cases on day 0) and returns a `numeric` vector of probabilities of the same length, allowing the test sensitivity to vary with time. For example, `\(t) ifelse(t < 30, 0, 0.8)` represents a testing programme that activates on day 30 with sensitivity 0.8.

Only symptomatic individuals that do not self-isolate are tested; a false-negative result means the case is not isolated via the testing pathway (see `outbreak_step()` for how isolation times are assigned). Default is 1, which assumes all tested individuals get a positive test result.

Value

A `list` with class `<ringbp_intervention_opts>`.

Examples

```
# quarantine is not active (default)
intervention_opts(quarantine = FALSE)

# quarantine is active
intervention_opts(quarantine = TRUE)

# 20% of tests return a false-negative
intervention_opts(test_sensitivity = 0.8)

# time-varying test sensitivity, in the first 30 days of the outbreak
# sensitivity is 0.5, then after 30 days, sensitivity improves to 0.8
intervention_opts(
  test_sensitivity = \(t) ifelse(t > 30, yes = 0.8, no = 0.5)
)
```

---

offspring_opts *Create a list of offspring distributions to run the* **ringbp** *model*

Description

Create a list of offspring distributions to run the **ringbp** model

Usage

```
offspring_opts(community, isolated, asymptomatic = community)
```

Arguments

`community`	a `function`: a random number generating `function` that samples from the community (non-isolated) offspring distribution, the `function` accepts a single `integer` argument specifying the number of times to sample the offspring distribution (i.e. the length of the `function` output)

isolated	a function: a random number generating function that samples from the offspring distribution of isolated cases, the function accepts a single integer argument specifying the number of times to sample the offspring distribution (i.e. the length of the function output). This distribution is used for the transmission of any case once it has been isolated, whether symptomatic or asymptomatic
asymptomatic	a function: a random number generating function that samples from the subclinical non-isolated cases offspring distribution, the function accepts a single integer argument specifying the number of times to sample the offspring distribution (i.e. the length of the function output). Will be specified as the same as the community offspring distribution if left unspecified

Details

If asymptomatic is not provided it will be specified as the same as community meaning transmission of subclinical cases to be equal to clinical cases unless specified otherwise.

Value

A list with class <ringbp_offspring_opts>.

Examples

```
# Negative binomial offspring distributions with:
# Community R0 of 2.5 and dispersion of 0.16
# Isolated R0 of 0.5 and dispersion of 1
# Asymptomatic R0 of 2.5 and dispersion of 0.16
offspring_opts(
  community = \(n) rnbinom(n = n, mu = 2.5, size = 0.16),
  isolated = \(n) rnbinom(n = n, mu = 0.5, size = 1),
  asymptomatic = \(n) rnbinom(n = n, mu = 2.5, size = 0.16)
)
```

---

outbreak_model *Run a single instance of the branching process model*

Description

Run a single instance of the branching process model

Usage

```
outbreak_model(
  initial_cases,
  offspring,
  delays,
  event_probs,
  interventions,
  sim
)
```

Arguments

initial_cases	a non-negative integer scalar: number of initial or starting cases which are all assumed to be missed by contact tracing (i.e. tracing ascertainment = 0).
offspring	a list with class <ringbp_offspring_opts>: the offspring distribution functions for the **ringbp** model, returned by offspring_opts(). Contains three elements: community, isolated, and asymptomatic
delays	a list with class <ringbp_delay_opts>: the delay distribution functions for the **ringbp** model, returned by delay_opts(). Contains 4 elements: incubation_period, onset_to_isolation, latent_period and onset_to_self_isolation
event_probs	a list with class <ringbp_event_prob_opts>: the event probabilities for the **ringbp** model, returned by event_prob_opts(). Contains 5 elements: asymptomatic, presymptomatic_transmission, alpha, symptomatic_traced and symptomatic_self_isolate
interventions	a list with class <ringbp_intervention_opts>: the intervention settings for the **ringbp** model, returned by intervention_opts(). Contains 2 elements: quarantine and test_sensitivity
sim	a list with class <ringbp_sim_opts>: the simulation control options for the **ringbp** model, returned by sim_opts()

Value

data.table of cases by week, cumulative cases, and the effective reproduction number of the outbreak. data.table columns are:

- $week: numeric
- $weekly_cases: numeric
- $cumulative: numeric
- $effective_r0: numeric
- $cases_per_gen: list

Examples

```
set.seed(1)
offspring <- offspring_opts(
  community = \(n) rnbinom(n = n, mu = 2.5, size = 0.16),
  isolated = \(n) rnbinom(n = n, mu = 0.5, size = 1),
  asymptomatic = \(n) rnbinom(n = n, mu = 2.5, size = 0.16)
)
delays <- delay_opts(
  incubation_period = \(n) rweibull(n = n, shape = 2.32, scale = 6.49),
  onset_to_isolation = \(n) rweibull(n = n, shape = 1.65, scale = 4.28)
)
event_probs <- event_prob_opts(
  asymptomatic = 0,
  presymptomatic_transmission = 0.5,
  symptomatic_traced = 0.2
)
interventions <- intervention_opts(quarantine = FALSE)
out <- outbreak_model(
```

```
  initial_cases = 1,
  offspring = offspring,
  delays = delays,
  event_probs = event_probs,
  interventions = interventions,
  sim = sim_opts()
)
out
```

`outbreak_setup` *Set up initial cases for branching process*

Description

Set up initial cases for branching process

Usage

```
outbreak_setup(initial_cases, delays, event_probs, interventions)
```

Arguments

`initial_cases`	a non-negative `integer` scalar: number of initial or starting cases which are all assumed to be missed by contact tracing (i.e. tracing ascertainment = 0).
`delays`	a `list` with class `<ringbp_delay_opts>`: the delay distribution `functions` for the **ringbp** model, returned by `delay_opts()`. Contains 4 elements: `incubation_period`, `onset_to_isolation`, `latent_period` and `onset_to_self_isolation`
`event_probs`	a `list` with class `<ringbp_event_prob_opts>`: the event probabilities for the **ringbp** model, returned by `event_prob_opts()`. Contains 5 elements: `asymptomatic`, `presymptomatic_transmission`, `alpha`, `symptomatic_traced` and `symptomatic_self_isolate`
`interventions`	a `list` with class `<ringbp_intervention_opts>`: the intervention settings for the **ringbp** model, returned by `intervention_opts()`. Contains 2 elements: `quarantine` and `test_sensitivity`

Value

`data.table` of cases in outbreak so far. `data.table` columns are:

- `$exposure`: `numeric`
- `$asymptomatic`: `logical`
- `$caseid`: `integer`
- `$infector`: `numeric`
- `$traced`: `logical`
- `$onset`: `numeric`
- `$new_cases`: `integer`
- `$self_isolate`: `logical`
- `$isolated_time`: `numeric`
- `$sampled`: `logical`

Examples

```
delays <- delay_opts(
  incubation_period = \(n) rweibull(n = n, shape = 2.32, scale = 6.49),
  onset_to_isolation = \(n) rweibull(n = n, shape = 1.65, scale = 4.28)
)
event_probs <- event_prob_opts(
  asymptomatic = 0,
  presymptomatic_transmission = 0.15,
  symptomatic_traced = 0
)
interventions <- intervention_opts()

# generate initial cases
case_data <- outbreak_setup(
  initial_cases = 5,
  delays = delays,
  event_probs = event_probs,
  interventions = interventions
)
case_data
```

`outbreak_step` *Move forward one generation in the branching process*

Description

Move forward one generation in the branching process

Usage

```
outbreak_step(case_data, offspring, delays, event_probs, interventions)
```

Arguments

`case_data`	a `data.table`: cases in outbreak so far; initially generated by `outbreak_setup()`
`offspring`	a `list` with class `<ringbp_offspring_opts>`: the offspring distribution `functions` for the **ringbp** model, returned by `offspring_opts()`. Contains three elements: `community`, `isolated`, and `asymptomatic`
`delays`	a `list` with class `<ringbp_delay_opts>`: the delay distribution `functions` for the **ringbp** model, returned by `delay_opts()`. Contains 4 elements: `incubation_period`, `onset_to_isolation`, `latent_period` and `onset_to_self_isolation`
`event_probs`	a `list` with class `<ringbp_event_prob_opts>`: the event probabilities for the **ringbp** model, returned by `event_prob_opts()`. Contains 5 elements: `asymptomatic`, `presymptomatic_transmission`, `alpha`, `symptomatic_traced` and `symptomatic_self_isolate`
`interventions`	a `list` with class `<ringbp_intervention_opts>`: the intervention settings for the **ringbp** model, returned by `intervention_opts()`. Contains 2 elements: `quarantine` and `test_sensitivity`

Details

Each new case is assigned an isolation time (`isolated_time`) as the earliest of up to three pathways; a case to which no pathway applies is never isolated (`isolated_time` is `Inf`):

- **Self-isolation**: a symptomatic case self-isolates with probability `symptomatic_self_isolate` (from `event_prob_opts()`), entering isolation an `onset_to_self_isolation` delay (from `delay_opts()`) after symptom onset. Self-isolating cases are not tested.
- **Testing**: a symptomatic case that does not self-isolate is tested and returns a positive result with probability `test_sensitivity` (from `intervention_opts()`), entering isolation an `onset_to_isolation` delay after symptom onset. A false-negative result does not isolate the case via this pathway.
- **Tracing**: a case whose infector is symptomatic is traced with probability `symptomatic_traced` (from `event_prob_opts()`). When `quarantine` is active (from `intervention_opts()`) tracing is exposure-based: a traced case is isolated when its infector is isolated, regardless of its own symptom status. Without `quarantine`, only a traced symptomatic case is isolated, and no earlier than its own symptom onset.

Self-isolation and testing are symptom-based, so asymptomatic cases are isolated only via the tracing pathway, and only when `quarantine` is active.

Value

A `list` with three elements:

1. `$cases`: a `data.table` with case data
2. `$effective_r0`: a `numeric` with the effective reproduction number
3. `$cases_in_gen`: a `numeric` with the number of new cases in that generation

Examples

```
offspring <- offspring_opts(
  community = \(n) rnbinom(n = n, mu = 2.5, size = 0.16),
  isolated = \(n) rnbinom(n = n, mu = 0, size = 1),
  asymptomatic = \(n) rnbinom(n = n, mu = 1.25, size = 0.16)
)
delays <- delay_opts(
  incubation_period = \(n) rweibull(n = n, shape = 2.32, scale = 6.49),
  onset_to_isolation = \(n) rweibull(n = n, shape = 1.65, scale = 4.28)
)
event_probs <- event_prob_opts(
  asymptomatic = 0,
  presymptomatic_transmission = 0.15,
  symptomatic_traced = 0
)
interventions <- intervention_opts(quarantine = FALSE)

# generate initial cases
case_data <- outbreak_setup(
  initial_cases = 5,
  delays = delays,
```

```
  event_probs = event_probs,
  interventions = interventions
)
case_data
# generate next generation of cases
out <- outbreak_step(
  case_data = case_data,
  offspring = offspring,
  delays = delays,
  event_probs = event_probs,
  interventions = interventions
)
case_data <- out[[1]]
case_data
```

`scenario_sim` *Run a specified number of simulations with identical parameters*

Description

Run a specified number of simulations with identical parameters

Usage

```
scenario_sim(
  n,
  initial_cases,
  offspring,
  delays,
  event_probs,
  interventions,
  sim
)
```

Arguments

`n`	a positive `integer` scalar: number of simulations to run
`initial_cases`	a non-negative `integer` scalar: number of initial or starting cases which are all assumed to be missed by contact tracing (i.e. tracing ascertainment = 0).
`offspring`	a `list` with class `<ringbp_offspring_opts>`: the offspring distribution `functions` for the **ringbp** model, returned by `offspring_opts()`. Contains three elements: `community`, `isolated`, and `asymptomatic`
`delays`	a `list` with class `<ringbp_delay_opts>`: the delay distribution `functions` for the **ringbp** model, returned by `delay_opts()`. Contains 4 elements: `incubation_period`, `onset_to_isolation`, `latent_period` and `onset_to_self_isolation`
`event_probs`	a `list` with class `<ringbp_event_prob_opts>`: the event probabilities for the **ringbp** model, returned by `event_prob_opts()`. Contains 5 elements: `asymptomatic`, `presymptomatic_transmission`, `alpha`, `symptomatic_traced` and `symptomatic_self_isolate`

interventions	a list with class <ringbp_intervention_opts>: the intervention settings for the **ringbp** model, returned by intervention_opts(). Contains 2 elements: quarantine and test_sensitivity
sim	a list with class <ringbp_sim_opts>: the simulation control options for the **ringbp** model, returned by sim_opts()

Value

A data.table object returning the results for multiple simulations using the same set of parameters. The table has columns

- week: The week in the simulation.
- weekly_cases: The number of new cases that week.
- cumulative: The cumulative cases.
- effective_r0: The effective reproduction rate for the whole simulation
- cases_per_gen: A list column with the cases per generation. This is repeated each row.
- sim: Index column for which simulation.

Examples

```
offspring <- offspring_opts(
  community = \(n) rnbinom(n = n, mu = 2.5, size = 0.16),
  isolated = \(n) rnbinom(n = n, mu = 0, size = 1),
  asymptomatic = \(n) rnbinom(n = n, mu = 2.5, size = 0.16)
)
delays <- delay_opts(
  incubation_period = \(n) rweibull(n = n, shape = 2.32, scale = 6.49),
  onset_to_isolation = \(n) rweibull(n = n, shape = 2.5, scale = 5)
)
event_probs <- event_prob_opts(
  asymptomatic = 0,
  presymptomatic_transmission = 0.3,
  symptomatic_traced = 0
)
interventions <- intervention_opts(quarantine = TRUE)
sim <- sim_opts(
  cap_max_days = 365,
  cap_cases = 2000
)
res <- scenario_sim(
  n = 5,
  initial_cases = 5,
  offspring = offspring,
  delays = delays,
  event_probs = event_probs,
  interventions = interventions,
  sim = sim
)
res
```

`sim_opts` *Create a list of simulation control options for the* **ringbp** *model*

Description

Create a list of simulation control options for the **ringbp** model

Usage

```
sim_opts(cap_max_days = 350, cap_cases = 5000)
```

Arguments

`cap_max_days`	a positive `integer` scalar: stops the outbreak simulation once all new infections occur on more than `cap_max_days` days after the start of the outbreak, counted from the exposure of the initial cases on day 0
`cap_cases`	a positive `integer` scalar: number of cumulative cases at which the branching process (simulation) was terminated

Value

A `list` with class `<ringbp_sim_opts>`.

Examples

```
# default simulation control options
sim_opts()

# specifying custom simulation control options
sim_opts(
  cap_max_days = 140,
  cap_cases = 1000
)
```

Index

delay_opts, 2
delay_opts(), *4*, *11–15*
detect_extinct *(*extinction*)*, 5
detect_extinct(), *6*

event_prob_opts, 4
event_prob_opts(), *3*, *8*, *11–15*
extinct_prob *(*extinction*)*, 5
extinct_prob(), *6*
extinction, 5

incubation_to_generation_time, 7
intervention_opts, 8
intervention_opts(), *11–14*, *16*

offspring_opts, 9
offspring_opts(), *11*, *13*, *15*
outbreak_model, 10
outbreak_model(), *6*
outbreak_setup, 12
outbreak_setup(), *13*
outbreak_step, 13
outbreak_step(), *4*, *9*

scenario_sim, 15
scenario_sim(), *3–6*
sim_opts, 17
sim_opts(), *11*, *16*