Package: simulist (via r-universe)

October 17, 2024

Title Simulate Disease Outbreak Line List and Contacts Data

Version 0.3.0.9000

Description Tools to simulate realistic raw case data for an epidemic in the form of line lists and contacts using a branching process. Simulated outbreaks are parameterised with epidemiological parameters and can have age structured populations, age-stratified hospitalisation and death risk and time-varying case fatality risk.

License MIT + file LICENSE

URL https://github.com/epiverse-trace/simulist, https://epiverse-trace.github.io/simulist/

BugReports https://github.com/epiverse-trace/simulist/issues

Depends R (>= 3.6.0)

Imports checkmate, epiparameter, randomNames, rlang, stats

Suggests dplyr, epicontacts (>= 1.1.3), ggplot2, incidence2 (>= 2.1.0), knitr, rmarkdown, spelling, testthat (>= 3.0.0), tidyr

VignetteBuilder knitr

Remotes epiverse-trace/epiparameter

Config/Needs/website epiverse-trace/epiversetheme

Config/testthat/edition 3

Encoding UTF-8

Language en-GB

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Repository https://epiverse-trace.r-universe.dev

RemoteUrl https://github.com/epiverse-trace/simulist

RemoteRef HEAD

RemoteSha d6c8ca7fcffd19cb048a7463da642016b7ede73e

Contents

`create_config` *Create a list of configuration settings for some details of* `sim_linelist()`

Description

Create a list of configuration settings for some details of `sim_linelist()`

Usage

```
create_config(...)
```

Arguments

`...`	<dynamic-dots> Named elements to replace default settings. Only if names match exactly are elements replaced, otherwise the function errors.

Details

The `config` argument in `sim_linelist()` controls the small details around time windows around infections (time of first contact and last contact with infector), and the distribution of the Cycle threshold (Ct) value from a Real-time PCR or quantitative PCR (qPCR) for confirmed cases, the network effect in the simulation, and if there is a time-varying death risk.

Accepted arguments and their defaults are:

- `last_contact_distribution = "pois"`
- `last_contact_distribution_params = c(lambda = 3)`
- `first_contact_distribution = "pois"`
- `first_contact_distribution_params = c(lambda = 3)`
- `ct_distribution = "norm"`
- `ct_distribution_params = c(mean = 25, sd = 2)`
- `network = "adjusted"`
- `time_varying_death_risk = NULL`

These parameters do not warrant their own arguments in `sim_linelist()` as they rarely need to be changed from their default setting. Therefore it is not worth increasing the number of `sim_linelist()` arguments to accommodate these and the `config` argument keeps the function signature simpler and more readable.

The accepted distributions are:

- last_contact_distribution = c("pois", "geom")
- first_contact_distribution = c("pois", "geom")
- ct_distribution = c("norm", "lnorm")

The network option controls whether to sample contacts from a adjusted or unadjusted contact distribution. Adjusted (default) sampling uses $q(n) \sim (n+1)p(n+1)$ where $p(n)$ is the probability density function of a distribution, e.g., Poisson or Negative binomial. Unadjusted (network = "unadjusted") instead samples contacts directly from a probability distribution $p(n)$.

Value

A list of settings for sim_linelist()

Examples

```
# example with default configuration
create_config()

# example with customised Ct distribution
create_config(
  ct_distribution = "lnorm",
  ct_distribution_params = c(meanlog = 2, sdlog = 1)
)
```

---

sim_contacts *Simulate contacts for an infectious disease outbreak*

Description

Simulate contacts for an infectious disease outbreak

Usage

```
sim_contacts(
  contact_distribution = function(x) stats::dpois(x = x, lambda = 2),
  infectious_period = function(x) stats::rlnorm(n = x, meanlog = 2, sdlog = 0.5),
  prob_infection = 0.5,
  outbreak_start_date = as.Date("2023-01-01"),
  anonymise = FALSE,
  outbreak_size = c(10, 10000),
  population_age = c(1, 90),
 contact_tracing_status_probs = c(under_followup = 0.7, lost_to_followup = 0.2, unknown
    = 0.1),
  config = create_config()
)
```

Arguments

`contact_distribution`

A `function` or an `<epiparameter>` object to generate the number of contacts per infection.

The function can be defined or anonymous. The function must have a single argument in the form of an `integer` vector with elements representing the number of contacts, and return a `numeric` vector where each element corresponds to the probability of observing the number of contacts in the vector passed to the function. The index of the `numeric` vector returned is offset by one to the corresponding probability of observing the number of contacts, i.e. the first element of the output vector is the probability of observing zero contacts, the second element is the probability of observing one contact, etc.

An `<epiparameter>` can be provided. This will be converted into a probability mass function internally.

The default is an anonymous function with a Poisson probability mass function (`dpois()`) with a mean (λ) of 2 contacts per infection.

`infectious_period`

A `function` or an `<epiparameter>` object for the infectious period. This defines the duration from becoming infectious to no longer infectious. In the simulation, individuals are assumed to become infectious immediately after being infected (the latency period is assumed to be zero). The time intervals between an infected individual and their contacts are assumed to be uniformly distributed within the infectious period. Infectious periods must be strictly positive.

The function can be defined or anonymous. The function must return a vector of randomly generated real numbers representing sampled infectious periods. The function must have a single argument, the number of random infectious periods to generate.

An `<epiparameter>` can be provided. This will be converted into random number generator internally.

The default is an anonymous function with a lognormal distribution random number generator (`rlnorm()`) with `meanlog = 2` and `sdlog = 0.5`.

`prob_infection` A single `numeric` for the probability of a secondary contact being infected by an infected primary contact.

`outbreak_start_date`

A `date` for the start of the outbreak.

`anonymise` A `logical` boolean for whether case names should be anonymised. Default is `FALSE`.

`outbreak_size` A `numeric` vector of length 2 defining the minimum and the maximum number of infected individuals for the simulated outbreak. Default is `c(10, 1e4)`, so the minimum outbreak size is 10 infected individuals, and the maximum outbreak size is 10,000 infected individuals. Either number can be changed to increase or decrease the maximum or minimum outbreak size to allow simulating larger or smaller outbreaks. If the minimum outbreak size cannot be reached after running the simulation for many iterations (internally) then the function errors, whereas if the maximum outbreak size is exceeded the function returns the data early and a warning stating how many cases and contacts are returned.

`population_age` Either a `numeric` vector with two elements or a `<data.frame>` with age structure in the population. Use a `numeric` vector to specific the age range of the population, the first element is the lower bound for the age range, and and the second is the upper bound for the age range (both inclusive, i.e. [lower, upper]). The `<data.frame>` with age groups and the proportion of the population in that group. See details and examples for more information.

`contact_tracing_status_probs` A named `numeric` vector with the probability of each contact tracing status. The names of the vector must be `"under_followup"`, `"lost_to_followup"`, `"unknown"`. Values of each contact tracing status must sum to one.

`config` A list of settings to adjust the randomly sampled delays and Ct values. See `create_config()` for more information.

Value

A contacts `<data.frame>`

Author(s)

Joshua W. Lambert, Carmen Tamayo

Examples

```
# quickly simulate contact tracing data using the function defaults
contacts <- sim_contacts()
head(contacts)

# to simulate more realistic contact tracing data load epiparameters from
# {epiparameter}
library(epiparameter)
contact_distribution <- epiparameter(
  disease = "COVID-19",
  epi_name = "contact distribution",
  prob_distribution = create_prob_distribution(
    prob_distribution = "pois",
    prob_distribution_params = c(mean = 2)
  )
)

infectious_period <- epiparameter(
  disease = "COVID-19",
  epi_name = "infectious period",
  prob_distribution = create_prob_distribution(
    prob_distribution = "gamma",
    prob_distribution_params = c(shape = 1, scale = 1)
  )
)

contacts <- sim_contacts(
  contact_distribution = contact_distribution,
  infectious_period = infectious_period,
```

```
  prob_infection = 0.5
)
```

`sim_linelist` *Simulate a line list*

Description

The line list is simulated using a branching process and parameterised with epidemiological parameters.

Usage

```
sim_linelist(
  contact_distribution = function(x) stats::dpois(x = x, lambda = 2),
  infectious_period = function(x) stats::rlnorm(n = x, meanlog = 2, sdlog = 0.5),
  prob_infection = 0.5,
  onset_to_hosp = function(x) stats::rlnorm(n = x, meanlog = 1.5, sdlog = 0.5),
  onset_to_death = function(x) stats::rlnorm(n = x, meanlog = 2.5, sdlog = 0.5),
  onset_to_recovery = NULL,
  hosp_risk = 0.2,
  hosp_death_risk = 0.5,
  non_hosp_death_risk = 0.05,
  outbreak_start_date = as.Date("2023-01-01"),
  anonymise = FALSE,
  outbreak_size = c(10, 10000),
  population_age = c(1, 90),
  case_type_probs = c(suspected = 0.2, probable = 0.3, confirmed = 0.5),
  config = create_config()
)
```

Arguments

`contact_distribution`

A `function` or an `<epiparameter>` object to generate the number of contacts per infection.

The function can be defined or anonymous. The function must have a single argument in the form of an `integer` vector with elements representing the number of contacts, and return a `numeric` vector where each element corresponds to the probability of observing the number of contacts in the vector passed to the function. The index of the `numeric` vector returned is offset by one to the corresponding probability of observing the number of contacts, i.e. the first element of the output vector is the probability of observing zero contacts, the second element is the probability of observing one contact, etc.

An `<epiparameter>` can be provided. This will be converted into a probability mass function internally.

The default is an anonymous function with a Poisson probability mass function (`dpois()`) with a mean (λ) of 2 contacts per infection.

infectious_period
: A function or an <epiparameter> object for the infectious period. This defines the duration from becoming infectious to no longer infectious. In the simulation, individuals are assumed to become infectious immediately after being infected (the latency period is assumed to be zero). The time intervals between an infected individual and their contacts are assumed to be uniformly distributed within the infectious period. Infectious periods must be strictly positive.

  The function can be defined or anonymous. The function must return a vector of randomly generated real numbers representing sampled infectious periods. The function must have a single argument, the number of random infectious periods to generate.

  An <epiparameter> can be provided. This will be converted into random number generator internally.

  The default is an anonymous function with a lognormal distribution random number generator (rlnorm()) with meanlog = 2 and sdlog = 0.5.

prob_infection
: A single numeric for the probability of a secondary contact being infected by an infected primary contact.

onset_to_hosp
: A function or an <epiparameter> object for the onset-to-hospitalisation delay distribution. onset_to_hosp can also be set to NULL to not simulate hospitalisation (admission) dates.

  The function can be defined or anonymous. The function must return a vector of numerics for the length of the onset-to-hospitalisation delay. The function must have a single argument.

  An <epiparameter> can be provided. This will be converted into a random number generator internally.

  The default is an anonymous function with a lognormal distribution random number generator (rlnorm()) with meanlog = 1.5 and sdlog = 0.5.

  If onset_to_hosp is set to NULL then hosp_risk and hosp_death_risk will be automatically set to NULL if not manually specified.

onset_to_death
: A function or an <epiparameter> object for the onset-to-death delay distribution. onset_to_death can also be set to NULL to not simulate dates for individuals that died.

  The function can be defined or anonymous. The function must return a vector of numerics for the length of the onset-to-death delay. The function must have a single argument.

  An <epiparameter> can be provided. This will be converted into a random number generator internally.

  The default is an anonymous function with a lognormal distribution random number generator (rlnorm()) with meanlog = 2.5 and sdlog = 0.5.

  If onset_to_death is set to NULL then non_hosp_death_risk and hosp_death_risk will be automatically set to NULL if not manually specified.

onset_to_recovery
: A function or an <epiparameter> object for the onset-to-recovery delay distribution. onset_to_recovery can also be NULL to not simulate dates for individuals that recovered.

The function can be defined or anonymous. The function must return a vector of numerics for the length of the onset-to-recovery delay. The function must have a single argument.

An <epiparameter> can be provided. This will be converted into a random number generator internally.

The default is NULL so by default cases that recover get an NA in the $date_outcome line list column.

hosp_risk
: Either a single numeric for the hospitalisation risk of everyone in the population, or a <data.frame> with age specific hospitalisation risks. Default is 20% hospitalisation (0.2) for the entire population. If the onset_to_hosp argument is set to NULL this argument will automatically be set to NULL if not specified or can be manually set to NULL. See details and examples for more information.

hosp_death_risk
: Either a single numeric for the death risk for hospitalised individuals across the population, or a <data.frame> with age specific hospitalised death risks Default is 50% death risk in hospitals (0.5) for the entire population. If the onset_to_death argument is set to NULL this argument will automatically be set to NULL if not specified or can be manually set to NULL. See details and examples for more information. The hosp_death_risk can vary through time if specified in the time_varying_death_risk element of config, see vignette("time-varying-cfr", package = "simulist") for more information.

non_hosp_death_risk
: Either a single numeric for the death risk for outside of hospitals across the population, or a <data.frame> with age specific death risks outside of hospitals. Default is 5% death risk outside of hospitals (0.05) for the entire population. If the onset_to_death argument is set to NULL this argument will automatically be set to NULL if not specified or can be manually set to NULL. See details and examples for more information. The non_hosp_death_risk can vary through time if specified in the time_varying_death_risk element of config, see vignette("time-varying-cfr", package = "simulist") for more information.

outbreak_start_date
: A date for the start of the outbreak.

anonymise
: A logical boolean for whether case names should be anonymised. Default is FALSE.

outbreak_size
: A numeric vector of length 2 defining the minimum and the maximum number of infected individuals for the simulated outbreak. Default is c(10, 1e4), so the minimum outbreak size is 10 infected individuals, and the maximum outbreak size is 10,000 infected individuals. Either number can be changed to increase or decrease the maximum or minimum outbreak size to allow simulating larger or smaller outbreaks. If the minimum outbreak size cannot be reached after running the simulation for many iterations (internally) then the function errors, whereas if the maximum outbreak size is exceeded the function returns the data early and a warning stating how many cases and contacts are returned.

population_age
: Either a numeric vector with two elements or a <data.frame> with age structure in the population. Use a numeric vector to specific the age range of the population, the first element is the lower bound for the age range, and and the

second is the upper bound for the age range (both inclusive, i.e. [lower, upper]). The <data.frame> with age groups and the proportion of the population in that group. See details and examples for more information.

`case_type_probs`
: A named `numeric` vector with the probability of each case type. The names of the vector must be `"suspected"`, `"probable"`, `"confirmed"`. Values of each case type must sum to one.

`config`
: A list of settings to adjust the randomly sampled delays and Ct values. See `create_config()` for more information.

Details

For age-stratified hospitalised and death risks a <data.frame> will need to be passed to the `hosp_risk` and/or `hosp_death_risk` arguments. This <data.frame> should have two columns:

- `age_limit`: a column with one `numeric` per cell for the lower bound (minimum) age of the age group (inclusive).
- `risk`: a column with one `numeric` per cell for the proportion (or probability) of hospitalisation for that age group. Should be between 0 and 1.

For an age structured population, a <data.frame> with two columns:

- `age_range`: a column with characters specifying the lower and upper bound of that age group, separated by a hyphen (-). Both bounds are inclusive (integers). For example, an age group of one to ten would be given as `"1-10"`.
- `proportion`: a column with the proportion of the population that are in that age group. Proportions must sum to one.

Value

A line list <data.frame>

Author(s)

Joshua W. Lambert, Carmen Tamayo

Examples

```
# quickly simulate a line list using the function defaults
linelist <- sim_linelist()
head(linelist)

# to simulate a more realistic line list load epiparameters from
# {epiparameter}
library(epiparameter)
contact_distribution <- epiparameter(
  disease = "COVID-19",
  epi_name = "contact distribution",
  prob_distribution = create_prob_distribution(
    prob_distribution = "pois",
```

```
    prob_distribution_params = c(mean = 2)
  )
)

infectious_period <- epiparameter(
  disease = "COVID-19",
  epi_name = "infectious period",
  prob_distribution = create_prob_distribution(
    prob_distribution = "gamma",
    prob_distribution_params = c(shape = 1, scale = 1)
  )
)

# get onset to hospital admission from {epiparameter} database
onset_to_hosp <- epiparameter_db(
  disease = "COVID-19",
  epi_name = "onset to hospitalisation",
  single_epiparameter = TRUE
)

# get onset to death from {epiparameter} database
onset_to_death <- epiparameter_db(
  disease = "COVID-19",
  epi_name = "onset to death",
  single_epiparameter = TRUE
)
# example with single hospitalisation risk for entire population
linelist <- sim_linelist(
  contact_distribution = contact_distribution,
  infectious_period = infectious_period,
  prob_infection = 0.5,
  onset_to_hosp = onset_to_hosp,
  onset_to_death = onset_to_death,
  hosp_risk = 0.5
)
head(linelist)

# example with age-stratified hospitalisation risk
# 20% for over 80s
# 10% for under 5s
# 5% for the rest
age_dep_hosp_risk <- data.frame(
  age_limit = c(1, 5, 80),
  risk = c(0.1, 0.05, 0.2)
)
linelist <- sim_linelist(
  contact_distribution = contact_distribution,
  infectious_period = infectious_period,
  prob_infection = 0.5,
  onset_to_hosp = onset_to_hosp,
  onset_to_death = onset_to_death,
  hosp_risk = age_dep_hosp_risk
)
```

```
head(linelist)
```

`sim_outbreak` *Simulate a line list and a contacts table*

Description

The line list and contacts are simulated using a branching process and parameterised with epidemiological parameters.

Usage

```
sim_outbreak(
  contact_distribution = function(x) stats::dpois(x = x, lambda = 2),
  infectious_period = function(x) stats::rlnorm(n = x, meanlog = 2, sdlog = 0.5),
  prob_infection = 0.5,
  onset_to_hosp = function(x) stats::rlnorm(n = x, meanlog = 1.5, sdlog = 0.5),
  onset_to_death = function(x) stats::rlnorm(n = x, meanlog = 2.5, sdlog = 0.5),
  onset_to_recovery = NULL,
  hosp_risk = 0.2,
  hosp_death_risk = 0.5,
  non_hosp_death_risk = 0.05,
  outbreak_start_date = as.Date("2023-01-01"),
  anonymise = FALSE,
  outbreak_size = c(10, 10000),
  population_age = c(1, 90),
  case_type_probs = c(suspected = 0.2, probable = 0.3, confirmed = 0.5),
 contact_tracing_status_probs = c(under_followup = 0.7, lost_to_followup = 0.2, unknown
    = 0.1),
  config = create_config()
)
```

Arguments

`contact_distribution`

A `function` or an `<epiparameter>` object to generate the number of contacts per infection.

The function can be defined or anonymous. The function must have a single argument in the form of an `integer` vector with elements representing the number of contacts, and return a `numeric` vector where each element corresponds to the probability of observing the number of contacts in the vector passed to the function. The index of the `numeric` vector returned is offset by one to the corresponding probability of observing the number of contacts, i.e. the first element of the output vector is the probability of observing zero contacts, the second element is the probability of observing one contact, etc.

An `<epiparameter>` can be provided. This will be converted into a probability mass function internally.

The default is an anonymous function with a Poisson probability mass function (dpois()) with a mean (λ) of 2 contacts per infection.

infectious_period

A function or an <epiparameter> object for the infectious period. This defines the duration from becoming infectious to no longer infectious. In the simulation, individuals are assumed to become infectious immediately after being infected (the latency period is assumed to be zero). The time intervals between an infected individual and their contacts are assumed to be uniformly distributed within the infectious period. Infectious periods must be strictly positive.

The function can be defined or anonymous. The function must return a vector of randomly generated real numbers representing sampled infectious periods. The function must have a single argument, the number of random infectious periods to generate.

An <epiparameter> can be provided. This will be converted into random number generator internally.

The default is an anonymous function with a lognormal distribution random number generator (rlnorm()) with meanlog = 2 and sdlog = 0.5.

prob_infection

A single numeric for the probability of a secondary contact being infected by an infected primary contact.

onset_to_hosp

A function or an <epiparameter> object for the onset-to-hospitalisation delay distribution. onset_to_hosp can also be set to NULL to not simulate hospitalisation (admission) dates.

The function can be defined or anonymous. The function must return a vector of numerics for the length of the onset-to-hospitalisation delay. The function must have a single argument.

An <epiparameter> can be provided. This will be converted into a random number generator internally.

The default is an anonymous function with a lognormal distribution random number generator (rlnorm()) with meanlog = 1.5 and sdlog = 0.5.

If onset_to_hosp is set to NULL then hosp_risk and hosp_death_risk will be automatically set to NULL if not manually specified.

onset_to_death

A function or an <epiparameter> object for the onset-to-death delay distribution. onset_to_death can also be set to NULL to not simulate dates for individuals that died.

The function can be defined or anonymous. The function must return a vector of numerics for the length of the onset-to-death delay. The function must have a single argument.

An <epiparameter> can be provided. This will be converted into a random number generator internally.

The default is an anonymous function with a lognormal distribution random number generator (rlnorm()) with meanlog = 2.5 and sdlog = 0.5.

If onset_to_death is set to NULL then non_hosp_death_risk and hosp_death_risk will be automatically set to NULL if not manually specified.

onset_to_recovery

A function or an <epiparameter> object for the onset-to-recovery delay distribution. onset_to_recovery can also be NULL to not simulate dates for individuals that recovered.

The function can be defined or anonymous. The function must return a vector of numerics for the length of the onset-to-recovery delay. The function must have a single argument.

An <epiparameter> can be provided. This will be converted into a random number generator internally.

The default is NULL so by default cases that recover get an NA in the $date_outcome line list column.

hosp_risk
: Either a single numeric for the hospitalisation risk of everyone in the population, or a <data.frame> with age specific hospitalisation risks. Default is 20% hospitalisation (0.2) for the entire population. If the onset_to_hosp argument is set to NULL this argument will automatically be set to NULL if not specified or can be manually set to NULL. See details and examples for more information.

hosp_death_risk
: Either a single numeric for the death risk for hospitalised individuals across the population, or a <data.frame> with age specific hospitalised death risks Default is 50% death risk in hospitals (0.5) for the entire population. If the onset_to_death argument is set to NULL this argument will automatically be set to NULL if not specified or can be manually set to NULL. See details and examples for more information. The hosp_death_risk can vary through time if specified in the time_varying_death_risk element of config, see vignette("time-varying-cfr", package = "simulist") for more information.

non_hosp_death_risk
: Either a single numeric for the death risk for outside of hospitals across the population, or a <data.frame> with age specific death risks outside of hospitals. Default is 5% death risk outside of hospitals (0.05) for the entire population. If the onset_to_death argument is set to NULL this argument will automatically be set to NULL if not specified or can be manually set to NULL. See details and examples for more information. The non_hosp_death_risk can vary through time if specified in the time_varying_death_risk element of config, see vignette("time-varying-cfr", package = "simulist") for more information.

outbreak_start_date
: A date for the start of the outbreak.

anonymise
: A logical boolean for whether case names should be anonymised. Default is FALSE.

outbreak_size
: A numeric vector of length 2 defining the minimum and the maximum number of infected individuals for the simulated outbreak. Default is c(10, 1e4), so the minimum outbreak size is 10 infected individuals, and the maximum outbreak size is 10,000 infected individuals. Either number can be changed to increase or decrease the maximum or minimum outbreak size to allow simulating larger or smaller outbreaks. If the minimum outbreak size cannot be reached after running the simulation for many iterations (internally) then the function errors, whereas if the maximum outbreak size is exceeded the function returns the data early and a warning stating how many cases and contacts are returned.

population_age
: Either a numeric vector with two elements or a <data.frame> with age structure in the population. Use a numeric vector to specific the age range of the population, the first element is the lower bound for the age range, and and the

second is the upper bound for the age range (both inclusive, i.e. [lower, upper]). The <data.frame> with age groups and the proportion of the population in that group. See details and examples for more information.

`case_type_probs`
: A named `numeric` vector with the probability of each case type. The names of the vector must be `"suspected"`, `"probable"`, `"confirmed"`. Values of each case type must sum to one.

`contact_tracing_status_probs`
: A named `numeric` vector with the probability of each contact tracing status. The names of the vector must be `"under_followup"`, `"lost_to_followup"`, `"unknown"`. Values of each contact tracing status must sum to one.

`config`
: A list of settings to adjust the randomly sampled delays and Ct values. See `create_config()` for more information.

Details

For age-stratified hospitalised and death risks a <data.frame> will need to be passed to the `hosp_risk` and/or `hosp_death_risk` arguments. This <data.frame> should have two columns:

- `age_limit`: a column with one `numeric` per cell for the lower bound (minimum) age of the age group (inclusive).
- `risk`: a column with one `numeric` per cell for the proportion (or probability) of hospitalisation for that age group. Should be between 0 and 1.

For an age structured population, a <data.frame> with two columns:

- `age_range`: a column with characters specifying the lower and upper bound of that age group, separated by a hyphen (-). Both bounds are inclusive (integers). For example, an age group of one to ten would be given as `"1-10"`.
- `proportion`: a column with the proportion of the population that are in that age group. Proportions must sum to one.

Value

A list with two elements:

1. A line list <data.frame>
2. A contacts <data.frame>

Author(s)

Joshua W. Lambert

Examples

```
# quickly simulate an outbreak using the function defaults
outbreak <- sim_outbreak()
head(outbreak$linelist)
head(outbreak$contacts)
```

```
# to simulate a more realistic outbreak load epiparameters from
# {epiparameter}
library(epiparameter)
contact_distribution <- epiparameter(
  disease = "COVID-19",
  epi_name = "contact distribution",
  prob_distribution = create_prob_distribution(
    prob_distribution = "pois",
    prob_distribution_params = c(mean = 2)
  )
)

infectious_period <- epiparameter(
  disease = "COVID-19",
  epi_name = "infectious period",
  prob_distribution = create_prob_distribution(
    prob_distribution = "gamma",
    prob_distribution_params = c(shape = 1, scale = 1)
  )
)

# get onset to hospital admission from {epiparameter} database
onset_to_hosp <- epiparameter_db(
  disease = "COVID-19",
  epi_name = "onset to hospitalisation",
  single_epiparameter = TRUE
)

# get onset to death from {epiparameter} database
onset_to_death <- epiparameter_db(
  disease = "COVID-19",
  epi_name = "onset to death",
  single_epiparameter = TRUE
)

outbreak <- sim_outbreak(
  contact_distribution = contact_distribution,
  infectious_period = infectious_period,
  prob_infection = 0.5,
  onset_to_hosp = onset_to_hosp,
  onset_to_death = onset_to_death
)
```

Index