Package 'SIR'

January 25, 2026

Type Package

Title Simulation and Inference for Deterministic Epidemic Models

Version 0.1.6

Description Tools for simulating and analyzing deterministic compartmental epidemic models using ordinary differential equations. The package currently supports classical SIR and SIRS models, including cumulative infection tracking and stochastic observation processes based on Poisson or negative binomial likelihoods. The output of simulations is returned as a dedicated S3 object with associated plotting and summary methods, facilitating reproducible epidemic modeling workflows.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Depends R (>= 4.1.0)

Imports deSolve,
stats

Suggests ggplot2,
devtools,
roxygen2,
knitr,
rmarkdown

VignetteBuilder knitr

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

URL https://github.com/vcastellar/SIR

BugReports https://github.com/vcastellar/SIR/issues

Contents

`compute_growth_metrics`
Compute short-horizon growth metrics from an infection time series

---

Description

Computes two related growth metrics from a univariate time series of (typically) **cumulative** counts:

1. **Daily percent increase** (`inc_pct`): for each time step t,

$$inc\_pct_t = (x_{t+1} - x_t)/x_t$$

   This is the one-step *relative* change, using the current value x_t as the denominator.

2. **Doubling time in days** (`doubling_days`): first computes a 4-point **left-aligned** moving average of `inc_pct` (call it α_t), then converts that average growth rate into an implied doubling time under constant-rate exponential growth:

$$\alpha_t = mean(inc\_pct_t, inc\_pct_{t+1}, inc\_pct_{t+2}, inc\_pct_{t+3})$$

$$doubling\_days_t = \log(2)/\log(1 + \alpha_t)$$

The doubling-time formula comes from solving $2 = (1 + \alpha)^k$ for k. If $1 + \alpha_t \leq 0$ or $\alpha_t = 0$ then the logarithm or division may yield non-finite values; infinite results are replaced with `NA`.

Usage

```
compute_growth_metrics(infected)
```

Arguments

`infected`	A univariate time series (typically a `ts`) of counts ordered in time. In most epidemiological uses this is a **cumulative** count series, but the function will work for any numeric series with positive values.

Details

Windowing / alignment

The percent increase is computed for adjacent pairs (x_t, x_{t+1}). The 4-point moving average of `inc_pct` is **left-aligned**, meaning the value at position t summarizes growth from t through $t + 3$.

Practical notes

- If `infected` contains zeros, `inc_pct` will be `Inf` or `NaN` at those points. This will propagate into `alpha` and `doubling_days`.
- Negative or decreasing series can produce $\alpha < 0$, for which $\log(1 + \alpha)$ may be undefined when $1 + \alpha \leq 0$.

Value

A named list with three numeric vectors:

inc_pct Vector of length `length(infected) - 1` with the one-step relative increases.

doubling_days Vector of length `length(infected) - 4` with the implied doubling time (in days) computed from the 4-point moving average of `inc_pct`.

alpha Vector of length `length(infected) - 4` with the 4-point moving average of `inc_pct`.

Examples

```
infected <- ts(c(100, 110, 121, 133, 146, 161))
out <- compute_growth_metrics(infected)
out$inc_pct
out$doubling_days
```

`fit_epi_model` *Fit an epidemic model by trajectory matching*

Description

Fits a deterministic compartmental epidemic model to observed data by minimizing a trajectory-matching loss (RMSE or log-RMSE) between the observed time series and the corresponding model-predicted trajectory.

The observable used for fitting is selected via the `target` argument and must correspond to one of the outputs declared by the underlying `epi_model` (e.g. `"incidence"`, `"I"`, `"S"`).

No probabilistic observation model is assumed. Parameter estimates should be interpreted as providing the best deterministic approximation to the observed trajectory under the chosen loss.

Usage

```
fit_epi_model(
  x,
  model = SIR_MODEL,
  loss = c("logrmse", "rmse"),
  init = NULL,
  target = "incidence",
  n_starts = 100,
  control = list(maxit = 500),
  seed = NULL,
  eps = 1e-06,
  verbose = TRUE,
  optim_method = "L-BFGS-B",
  ...
)
```

Arguments

x	Numeric vector of observed data used for fitting.
model	An object of class "epi_model" defining the epidemic model.
loss	Loss function to minimize. One of "logrmse" (default) or "rmse".
init	Named list or vector specifying the initial state of the model. Names must match model$state_names.
target	Character string specifying which model output is matched to x. Must be one of model$outputs (e.g. "incidence", "I", "S").
n_starts	Integer. Number of random multi-start initializations used to reduce sensitivity to local minima.
control	Control list passed to optim for the final optimization step (e.g. maxit).
seed	Optional integer. If provided, sets the random seed for reproducible multi-start initialization.
eps	Small positive constant used for numerical stability in the log-RMSE loss.
verbose	Logical. If TRUE, progress information from the multi-start and final optimization phases is printed.
optim_method	Character string specifying the optimization algorithm passed to optim. Defaults to "L-BFGS-B". Methods that do not support box constraints (e.g. "Nelder-Mead") are handled automatically.
...	Additional arguments passed directly to ode for numerical integration of the ODE system (e.g. method, rtol, atol).

Value

An object of class "fit_epi_model" containing the fitted parameters, optimization diagnostics, the selected fitting target, and the ODE control settings used during fitting.

See Also

simulate_epi, predict.fit_epi_model

Examples

```
## Not run:
## ------------------------------------------------------------
## Example 1: Fit incidence using log-RMSE
## ------------------------------------------------------------
sim <- simulate_epi(
  model = SIR_MODEL,
  times = 0:200,
  parms = SIR_MODEL$defaults,
  init  = list(S = 1e6, I = 20, R = 0),
  seed  = 22
)

x_inc <- sim$incidence$inc

fit_inc <- fit_epi_model(
  x = x_inc,
  model = SIR_MODEL,
  target = "incidence",
  loss = "logrmse",
  init = list(S = 1e6, I = 6, R = 0)
)

print(fit_inc)

## ------------------------------------------------------------
## Example 2: Fit a state variable using a different optimizer
## ------------------------------------------------------------
fit_I <- fit_epi_model(
  x = sim$states$I,
  model = SIR_MODEL,
  target = "I",
  init = list(S = 1e6, I = 10, R = 0),
  optim_method = "Nelder-Mead",
  method = "rk4",
  rtol = 1e-8,
  atol = 1e-10
)

print(fit_I)

## End(Not run)
```

`fit_epi_model-class` *Fitted epidemic model (trajectory matching)*

Description

Objects of class "fit_epi_model" represent the result of fitting a deterministic epidemic model to incidence data by minimizing a trajectory-matching loss (RMSE or log-RMSE).

No probabilistic observation model is assumed. The fitted parameters should be interpreted as providing the best deterministic approximation to the observed incidence trajectory under the chosen loss.

Usage

```
new_fit_epi_model(
  model,
  par,
  optim,
  loss,
  target,
  init,
  ini0,
  x,
  ode_control = NULL
)
```

Contents

A "fit_epi_model" object is a named list with components:

model The fitted epi_model object.

par Named numeric vector of fitted parameters.

loss Loss minimized ("rmse" or "logrmse").

target Model output used for fitting (e.g. "incidence", "I").

value Final loss value.

optim Object returned by stats::optim().

convergence Convergence code from optim().

message Optional convergence message.

init Initial condition arguments passed by the user.

ini0 Initial state vector used internally.

x Observed data used for fitting.

See Also

fit_epi_model, predict.fit_epi_model

`new_epi_model` *Create a new epidemic model object*

Description

Constructs an object of class `"epi_model"` representing a deterministic compartmental epidemic model. In addition to the ODE system, the model must explicitly declare which output of the right-hand side corresponds to the latent incidence rate.

Usage

```
new_epi_model(
  name,
  rhs,
  state_names,
  par_names,
  outputs = state_names,
  lower = NULL,
  upper = NULL,
  defaults = NULL,
  init = NULL
)
```

Arguments

`name`	Character scalar. Human-readable name of the model (e.g. `"SIR"`).
`rhs`	Function defining the ODE system. Must have signature `function(time, state, parms)` and return a list whose first element is the vector of state derivatives.
`state_names`	Character vector of state variable names.
`par_names`	Character vector of parameter names.
`lower`	Optional named numeric vector of lower parameter bounds.
`upper`	Optional named numeric vector of upper parameter bounds.
`defaults`	Optional named numeric vector of default parameter values.
`init`	Optional named numeric vector of default initial conditions.
`incidence`	Character scalar. Name of the `rhs` output corresponding to the latent incidence rate.

Details

Incidence:

The `rhs` function may return additional named outputs besides the state derivatives. One of these outputs must represent the **latent incidence rate** (e.g. force of infection or case-generation rate).

The `incidence` argument specifies the name of this output. This makes the definition of epidemic cases explicit and allows generic functions such as `simulate_epi()`, `fit_epi_model()`, and `summary()` to operate consistently across models.

Value

An object of class "epi_model".

Examples

```
sir_model <- new_epi_model(
  name = "SIR",
  rhs = sir_rhs,
  state_names = c("S", "I", "R"),
  par_names = c("beta", "gamma"),
  incidence = "incidence"
)
```

---

plot.epi_model_predict

Plot predictions from a fitted epidemic model

Description

Plot method for objects of class "epi_model_predict". The function displays the observed incidence data used for fitting together with the model-predicted incidence trajectory.

Usage

```
## S3 method for class 'epi_model_predict'
plot(x, type_obs = "h", col_obs = "black", col_pred = "red", lwd_pred = 2, ...)
```

Arguments

x	An object of class "epi_model_predict".
type_obs	Plot type for observed data (default: "h").
col_obs	Color for observed incidence.
col_pred	Color for predicted incidence.
lwd_pred	Line width for predicted incidence.
...	Further graphical parameters passed to plot.

Value

Invisibly returns the input object x.

`plot.sim_epi` *Plot a simulated epidemic*

Description

Plot method for objects of class `"sim_epi"` as returned by `simulate_epi`. The function can display either the compartmental state trajectories defined by the underlying `epi_model` or the observed incidence time series, depending on the value of the `what` argument.

Unlike earlier versions, this method does not assume a specific compartmental structure (e.g. SIR or SEIR). The set of states to be plotted is taken directly from the `epi_model` used to generate the simulation.

Usage

```
## S3 method for class 'sim_epi'
plot(
  x,
  what = c("states", "incidence"),
  scale = c("auto", "full", "small", "log"),
  ...
)
```

Arguments

`x`	Object of class `"sim_epi"` as returned by `simulate_epi`.
`what`	Character string specifying the type of plot to produce. Either `"states"` to plot compartmental state trajectories or `"incidence"` to plot observed incidence counts.
`scale`	Character string specifying the scale for state plots. One of `"auto"`, `"full"`, `"small"`, or `"log"`. This argument is ignored when `what = "incidence"`.
`...`	Additional graphical parameters passed to base plotting functions such as `plot` and `matplot`.

Details

When `what = "states"`, the function plots the time evolution of all state variables defined in `x$model$state_names`, using the simulated trajectories stored in `x$states`. The time variable is taken from the `time` column of `x$states` and is not considered a state.

When `what = "incidence"`, the function plots the observed incidence counts generated by the observation model and stored in `x$incidence`. If no observation model was specified during simulation (i.e. `obs = "none"`), this option results in an error.

If available, the time unit stored in the `sim_epi` object (i.e. `x$time_unit`) is used to label the time axis. This affects only the plot labels and does not change the numerical values of time.

Value

Invisibly returns the input object `x`.

See Also

`simulate_epi`, `summary.sim_epi`, `print.sim_epi`

Examples

```
sim <- simulate_epi(
  model = SIR_MODEL,
  times = 0:200,
  time_unit = "days",
  parms = c(beta = 0.3, gamma = 0.1),
  init  = c(S = 999990, I = 10, R = 0, C = 10),
  obs   = "poisson",
  seed  = 1
)

# Plot all model states
plot(sim)

# Plot observed incidence (requires an observation model)
plot(sim, what = "incidence")
```

`predict.fit_epi_model` *Predict epidemic dynamics from a fitted epidemic model*

Description

Computes forward predictions from a fitted epidemic model by solving the underlying ODE system using the estimated parameters.

Usage

```
## S3 method for class 'fit_epi_model'
predict(object, n_days, init = NULL, times = NULL, method = "lsoda", ...)
```

Arguments

`object`	An object of class `"fit_epi_model"`.
`n_days`	Integer. Number of days to predict ahead.
`init`	Named numeric vector of initial states. If `NULL`, the initial state is reconstructed using `object$ini0`.
`times`	Optional numeric vector of time points. Overrides `n_days`. Defaults to `0:(n_days - 1)` when `NULL`.
`method`	Integration method passed to `deSolve::ode()`.
`...`	Currently unused.

Value

An object of class "epi_model_predict".

Examples

```
## Not run:
sim <- simulate_epi(
  model = SIR_MODEL,
  n_days = 200,
  parms = c(beta = 0.30, gamma = 0.10),
  init = list(S = 1e6, I = 20, R = 0),
  obs = "poisson"
)
plot(sim)
inc_obs <- sim$incidence$inc
plot(seq_along(inc_obs) - 1, inc_obs,
     type = "l", xlab = "Day", ylab = "Incidence")
fit <- fit_epi_model(inc_obs,
                     loss = "logrmse",
                     model = SIRS_MODEL,
                     init = list(I0 = 6, N = 1e6))
init <- tail(sim$states, n = 1)[, -1]
pred <- predict(
  object = fit,
  n_days = 1000,
  init = init
)
plot(pred)
plot(sim)
plot(pred$states$time, pred$states$I)
plot(pred$states$I, col = "red", lty = 2)
lines(pred$states$S, col = "blue", lty = 2)

## End(Not run)
```

---

print.epi_model *Print an epidemic model object*

Description

Provides a concise, human-readable summary of an epi_model object. The printed output includes the model name, state variables, parameters, the declared model outputs, and the underlying system of differential equations.

This method is automatically called when an object of class "epi_model" is printed at the console.

Usage

```
## S3 method for class 'epi_model'
print(x, ...)
```

Arguments

x	An object of class "epi_model".
...	Further arguments (ignored).

Details

The epi_model class explicitly declares the set of model outputs via the outputs field. These outputs may include state variables, derived quantities such as incidence, or any other observable returned by the model's right-hand side (rhs) function.

Parameter bounds are shown when available. The model equations are printed by deparsing the rhs function for inspection.

Value

Invisibly returns the input object x.

Examples

```
SIR_MODEL
```

---

print.fit_epi_model *Print a fitted epidemic model*

Description

Print method for objects of class "fit_epi_model".

The function displays a concise, human-readable summary of the result of fitting an epidemic model to incidence data. The printed output includes the model name, the loss function used for fitting, the final objective value, convergence information from the optimizer, and the estimated model parameters.

This method is automatically called when an object of class "fit_epi_model" is printed at the console.

Usage

```
## S3 method for class 'fit_epi_model'
print(x, ...)
```

Arguments

x	An object of class "fit_epi_model" as returned by fit_epi_model.
...	Further arguments (ignored).

Details

The reported objective value corresponds to the value of the loss function evaluated at the estimated parameter vector. Convergence information and optional messages are taken directly from the underlying optimization routine.

The printed parameter estimates represent point estimates obtained by minimizing the chosen loss function; no uncertainty quantification is provided by this method.

Value

Invisibly returns the input object `x`.

Examples

```
## Not run:
fit <- fit_epi_model(
  x = incidence_data,
  model = SIR_MODEL
)

fit

## End(Not run)
```

`print.sim_epi` *Print a simulated epidemic*

Description

Print method for objects of class `"sim_epi"` as returned by `simulate_epi`. The function provides a concise, human-readable summary of a simulated epidemic, including the underlying epidemic model, the simulation time horizon, the parameter values used, and selected outcome metrics derived from the simulation results.

Unlike earlier versions, this method does not assume a specific compartmental structure (such as SIR or SEIR). Model-specific information is obtained directly from the `epi_model` object stored in `x$model`.

Usage

```
## S3 method for class 'sim_epi'
print(x, ...)
```

Arguments

`x`	Object of class `"sim_epi"` as returned by `simulate_epi`.
`...`	Additional arguments ignored by this method; included for compatibility with the generic `print` function.

Details

This method is automatically dispatched when an object of class `"sim_epi"` is printed. It is intended to give a quick overview of the simulation without displaying the full internal structure of the object.

If available, the time unit stored in the `sim_epi` object (i.e. `x$time_unit`) is used to label the simulation horizon and reported event times. This affects only the printed labels and does not change the numerical values of time.

The printed output typically includes:

- the name of the epidemic model,
- the simulated time horizon,
- the parameter values used in the simulation,
- basic outcome summaries (e.g. peak prevalence or total infections), when such quantities are available.

Value

Invisibly returns the input object `x`.

See Also

`simulate_epi`, `summary.sim_epi`, `plot.sim_epi`

Examples

```
sim <- simulate_epi(
  model = SIR_MODEL,
  times = 0:200,
  time_unit = "days",
  parms = c(beta = 0.30, gamma = 0.10),
  init  = c(S = 999990, I = 10, R = 0, C = 10),
  seed  = 1
)

sim
```

SEIRS_MODEL *SEIRS epidemic model with latent period and waning immunity*

Description

An `epi_model` object representing a deterministic **SEIRS** (Susceptible–Exposed–Infectious–Recovered–Susceptible) compartmental epidemic model with a latent (exposed) period and waning immunity.

The model describes the spread of an infection in a closed population where susceptible individuals become infected at rate $\lambda(t)$ and enter the exposed compartment `E`. Exposed individuals progress to the infectious compartment at rate `sigma`. Infectious individuals recover at rate `gamma`, and recovered individuals lose immunity at rate `omega`, returning to the susceptible compartment.

Usage

```
SEIRS_MODEL
```

Format

An object of class `"epi_model"`.

Details

State variables:

The model is defined in terms of the following state variables:

S(t) Number of susceptible individuals at time t.

E(t) Number of exposed (infected but not yet infectious) individuals at time t.

I(t) Number of infectious (actively infected) individuals at time t.

R(t) Number of recovered individuals with temporary immunity at time t.

The total population size is conserved:

$$N = S(t) + E(t) + I(t) + R(t).$$

Model outputs:

The SEIRS model declares the following outputs:

`"S"` Susceptible population size.

`"E"` Exposed (latent) population size.

`"I"` Infectious population size.

`"R"` Recovered (temporarily immune) population size.

`"incidence"` Rate of progression from `E` to `I`, $\sigma E(t)$, representing the instantaneous incidence of new infectious cases returned by the model's right-hand side.

All declared outputs may be used as observables in generic utilities such as `fit_epi_model` via the `target` argument.

Parameters:

The SEIRS model depends on the following parameters:

beta Transmission rate (per day).

sigma Rate of progression from exposed to infectious (per day); $1/\sigma$ is the mean latent period.

gamma Recovery/removal rate from infectious to recovered (per day); $1/\gamma$ is the mean infectious period.

omega Rate of waning immunity from `R` back to `S` (per day); $1/\omega$ is the mean immunity duration.

Model equations:

New infections occur at rate

$$\lambda(t) = \beta \frac{S(t)\, I(t)}{N}.$$

Case incidence (entries into `I`) occurs at rate

$$\text{incidence}(t) = \sigma E(t).$$

The system of ordinary differential equations is:

$$\begin{aligned}
\frac{dS}{dt} &= -\lambda(t) + \omega R(t), \\
\frac{dE}{dt} &= \lambda(t) - \sigma E(t), \\
\frac{dI}{dt} &= \sigma E(t) - \gamma I(t), \\
\frac{dR}{dt} &= \gamma I(t) - \omega R(t).
\end{aligned}$$

Usage:

This predefined model object is intended to be used with generic utilities such as `simulate_epi`, `fit_epi_model`, and `predict.fit_epi_model` that operate on `epi_model` objects.

See Also

`simulate_epi`, `fit_epi_model`, `new_epi_model`

Examples

```
## Simulate a SEIRS epidemic
sim <- simulate_epi(
  model = SEIRS_MODEL,
  times = 0:300,
  parms = c(beta = 0.3, sigma = 0.2, gamma = 0.14, omega = 0.01),
  init  = c(S = 1e6, E = 0, I = 20, R = 0),
  obs   = "poisson"
)

plot(sim)

## Plot observed incidence
plot(sim, what = "incidence")

## Fit the model to observed incidence
fit_inc <- fit_epi_model(
  x = sim$incidence$inc,
  model = SEIRS_MODEL,
  init = SEIRS_MODEL$init,
  target = "incidence"
)

fit_inc
```

SEIR_MODEL *SEIR epidemic model with latent (exposed) period*

Description

An `epi_model` object representing a deterministic **SEIR** (Susceptible–Exposed–Infectious–Recovered) compartmental epidemic model with a latent (exposed) period.

The model describes the spread of an infection in a closed population where susceptible individuals become infected at rate $\lambda(t)$ and enter the exposed compartment `E`. Exposed individuals progress to the infectious compartment at rate `sigma` and subsequently recover with permanent immunity.

Usage

```
SEIR_MODEL
```

Format

An object of class `"epi_model"`.

Details

State variables:

The model is defined in terms of the following state variables:

S(t) Number of susceptible individuals at time t.

E(t) Number of exposed (infected but not yet infectious) individuals at time t.

I(t) Number of infectious (actively infected) individuals at time t.

R(t) Number of recovered (immune) individuals at time t.

The total population size is conserved:

$$N = S(t) + E(t) + I(t) + R(t).$$

Model outputs:

The SEIR model declares the following outputs:

`"S"` Susceptible population size.

`"E"` Exposed (latent) population size.

`"I"` Infectious population size.

`"R"` Recovered (immune) population size.

`"incidence"` Rate of progression from `E` to `I`, $\sigma E(t)$, representing the instantaneous incidence of new infectious cases returned by the model's right-hand side.

All declared outputs may be used as observables in generic utilities such as `fit_epi_model` via the `target` argument.

Parameters:

The SEIR model depends on the following parameters:

beta Transmission rate (per day).

sigma Rate of progression from exposed to infectious (per day); $1/\sigma$ is the mean latent period.

gamma Recovery/removal rate from infectious to recovered (per day); $1/\gamma$ is the mean infectious period.

Model equations:

New infections occur at rate

$$\lambda(t) = \beta \frac{S(t)\, I(t)}{N}.$$

Progression from exposed to infectious occurs at rate

$$\text{incidence}(t) = \sigma E(t).$$

The system of ordinary differential equations is:

$$\begin{aligned}
\frac{dS}{dt} &= -\lambda(t), \\
\frac{dE}{dt} &= \lambda(t) - \sigma E(t), \\
\frac{dI}{dt} &= \sigma E(t) - \gamma I(t), \\
\frac{dR}{dt} &= \gamma I(t).
\end{aligned}$$

Usage:

This predefined model object is intended to be used with generic utilities such as `simulate_epi`, `fit_epi_model`, and `predict.fit_epi_model` that operate on `epi_model` objects.

See Also

`simulate_epi`, `fit_epi_model`, `new_epi_model`

Examples

```
## Simulate a SEIR epidemic
sim <- simulate_epi(
  model = SEIR_MODEL,
  times = 0:200,
  parms = c(beta = 0.3, sigma = 0.2, gamma = 0.14),
  init  = c(S = 1e6, E = 5, I = 10, R = 0),
  obs   = "poisson"
)

plot(sim)

## Plot observed incidence
plot(sim, what = "incidence")

## Fit the model to observed incidence
fit_inc <- fit_epi_model(
  x = sim$incidence$inc,
```

```
  model = SEIR_MODEL,
  init = SEIR_MODEL$init,
  target = "incidence"
)

fit_inc
```

`simulate_epi` *Simulate an epidemic model defined by an* `epi_model` *object*

Description

Simulates a deterministic compartmental epidemic model specified via an `epi_model` object. The model is solved as a system of ordinary differential equations (ODEs) using `ode`. Optionally, a stochastic observation process can be applied to a model-defined incidence process to generate reported incidence counts.

The function is fully model-agnostic: all model-specific information (state variables, parameters, default values, ODE right-hand side, and optional incidence definition) is read directly from the supplied `epi_model`.

Usage

```
simulate_epi(
  model,
  times,
  time_unit = "days",
  parms = NULL,
  init = NULL,
  obs = c("none", "poisson", "negbin"),
  size = 20,
  seed = NULL,
  ...
)
```

Arguments

`model`	An object of class `"epi_model"` defining the epidemic model to simulate.
`times`	Numeric vector of time points at which to solve the ODE system. Must be strictly increasing and of length >= 2. The first value typically corresponds to the initial time (e.g. `0`).
`time_unit`	Character string specifying the unit of time associated with `times`. One of `"days"`, `"weeks"`, `"month"`, or `"year"`. This is used only for printing and plotting labels and does not affect the numerical simulation. The optional `time_unit` argument can be used to label the time axis in printed summaries and plots, but does not affect the numerical solution.

parms	Named numeric vector of model parameters. Names must match model$par_names. Any missing parameters are taken from model$defaults, if available.
init	Named numeric vector giving the initial values of the state variables. Names must exactly match model$state_names.
obs	Character string specifying the observation model. One of "none", "poisson", or "negbin".
size	Numeric. Dispersion (size) parameter for the negative binomial observation model.
seed	Optional integer. If provided, sets the random seed for reproducible observation draws.
...	Additional arguments passed directly to ode, such as method, rtol, or atol.

Details

Model structure:

The model argument must be an object of class "epi_model", typically created with new_epi_model. At minimum, the model must define:

- an ODE right-hand side function (model$rhs);
- the state variables (model$state_names);
- the model parameters (model$par_names).

Optionally, the model may define a latent incidence process by returning a variable named "incidence" as an additional output of the ODE right-hand side function. If no such variable is defined, no incidence time series is produced.

Time grid:

The model is simulated at the time points specified by the numeric vector times. This vector must be strictly increasing and of length >= 2. The first value typically corresponds to the initial time (e.g. 0).

This design follows the interface of ode and provides full control over the temporal resolution of the simulation, including irregular or non-integer time grids.

Initial conditions:

The initial state init must be supplied explicitly as a named numeric vector whose names match model$state_names. No automatic construction of initial conditions is performed.

Observation model:

If the model defines a latent incidence process, it can optionally be converted into reported incidence counts using a simple observation model:

obs = "poisson" Reported counts are drawn from a Poisson distribution with mean equal to the latent incidence.

obs = "negbin" Reported counts are drawn from a negative binomial distribution with mean equal to the latent incidence and dispersion parameter size.

obs = "none" No stochastic observation model is applied; the observed incidence is taken to be equal to the latent incidence.

If the model does not define a latent incidence process, the arguments obs and size are ignored and no incidence outputs are produced.

Numerical integration:

The ODE system is solved using ode. Additional arguments passed via ... are forwarded directly to deSolve::ode(), allowing full control over the numerical integration. In particular, the integration method can be specified using the method argument.

If method is not supplied, deSolve::ode() uses its default integration method (typically "lsoda").

Value

An object of class "sim_epi", containing:

model The epi_model object used to generate the simulation.

params A list of model parameter values used in the simulation.

states A data frame with columns time and the model state variables, containing the simulated state trajectories.

incidence A data frame with columns time and inc containing the observed (reported) incidence counts, or NULL if no incidence process is defined by the model.

incidence_cum A data frame with columns time and cases_cum containing cumulative observed cases, or NULL.

See Also

new_epi_model, ode, plot.sim_epi, print.sim_epi

Examples

```
## ------------------------------------------------------------------
## Example 1: SIR model with model-defined incidence
## ------------------------------------------------------------------

sim <- simulate_epi(
  model = SIR_MODEL,
  times = 0:200,
  time_unit = "week",
  parms = c(beta = 0.30, gamma = 0.10),
  init  = c(S = 1e6 - 10, I = 10, R = 0, C = 10),
  seed  = 1,
  method = 'lsoda'
)

# Plot state trajectories defined by the model
plot(sim)

# Plot observed incidence (requires the model to define incidence)
plot(sim, what = "incidence")


## ------------------------------------------------------------------
## Example 2: Using a different numerical integration method
## ------------------------------------------------------------------
```

```
sim_rk4 <- simulate_epi(
  model = SIR_MODEL,
  times = 0:200,
  parms = c(beta = 0.30, gamma = 0.10),
  init  = c(S = 1e6 - 10, I = 10, R = 0, C = 10),
  method = "rk4"
)

plot(sim_rk4)


## ------------------------------------------------------------------
## Example 3: Model without an incidence definition
## ------------------------------------------------------------------

sim <- simulate_epi(
  model = SI_MODEL,
  times = 30:100,
  parms = c(beta = 0.25),
  init  = c(S = 999, I = 1)
)

plot(sim)
plot(sim, what = "incidence")


## ------------------------------------------------------------------
## Example 4: Stochastic observation model
## ------------------------------------------------------------------

sim <- simulate_epi(
  model = SIR_MODEL,
  times = 0:150,
  parms = c(beta = 0.35, gamma = 0.12),
  init  = c(S = 1e5 - 5, I = 5, R = 0, C = 5),
  obs   = "negbin",
  size  = 20,
  seed  = 123
)

plot(sim, what = "incidence")
```

SIRS_MODEL *SIRS epidemic model with waning immunity*

Description

An `epi_model` object representing a deterministic **SIRS** (Susceptible–Infectious–Recovered–Susceptible) compartmental epidemic model with waning immunity.

The model describes the spread of an infection in a closed population where susceptible individuals become infectious, subsequently recover, and may lose immunity over time, returning to the susceptible compartment at rate `omega`.

Usage

```
SIRS_MODEL
```

Format

An object of class `"epi_model"`.

Details

State variables:

The model is defined in terms of the following state variables:

S(t) Number of susceptible individuals at time t.

I(t) Number of infectious (actively infected) individuals at time t.

R(t) Number of recovered individuals with temporary immunity at time t.

The total population size is conserved:

$$N = S(t) + I(t) + R(t).$$

Model outputs:

The SIRS model declares the following outputs:

`"S"` Susceptible population size.

`"I"` Infectious population size.

`"R"` Recovered (temporarily immune) population size.

`"incidence"` Instantaneous rate of new infections $\lambda(t)$ returned by the model's right-hand side.

All declared outputs may be used as observables in generic utilities such as `fit_epi_model` via the `target` argument.

Parameters:

The SIRS model depends on the following parameters:

beta Transmission rate (per day).

gamma Recovery/removal rate (per day).

omega Rate of waning immunity from `R` back to `S` (per day).

Model equations:

New infections occur at rate

$$\lambda(t) = \beta \frac{S(t)\, I(t)}{N}.$$

The system of ordinary differential equations is:

$$\begin{aligned}
\frac{dS}{dt} &= -\lambda(t) + \omega R(t), \\
\frac{dI}{dt} &= \lambda(t) - \gamma I(t), \\
\frac{dR}{dt} &= \gamma I(t) - \omega R(t).
\end{aligned}$$

Usage:

This predefined model object is intended to be used with generic utilities such as `simulate_epi`, `fit_epi_model`, and `predict.fit_epi_model` that operate on `epi_model` objects.

See Also

`simulate_epi`, `fit_epi_model`, `new_epi_model`

Examples

```
## Simulate a SIRS epidemic
sim <- simulate_epi(
  model = SIRS_MODEL,
  times = 0:200,
  parms = c(beta = 0.3, gamma = 0.1, omega = 0.02),
  init  = c(S = 1e6, I = 20, R = 0)
)

plot(sim)

## Plot observed incidence (if an observation model is used)
plot(sim, what = "incidence")

## Fit the model to observed incidence
fit_inc <- fit_epi_model(
  x = sim$incidence$inc,
  model = SIRS_MODEL,
  init = SIRS_MODEL$init,
  target = "incidence"
)

fit_inc
```

SIR_MODEL *SIR epidemic model*

Description

An `epi_model` object representing a deterministic **SIR** (Susceptible–Infectious–Recovered) compartmental epidemic model.

The model describes the spread of an infection in a closed population where susceptible individuals become infectious and subsequently recover with permanent immunity.

Usage

```
SIR_MODEL
```

Format

An object of class `"epi_model"`.

Details

State variables:

The model is defined in terms of the following state variables:

S(t) Number of susceptible individuals at time t.

I(t) Number of infectious (actively infected) individuals at time t.

R(t) Number of removed/recovered individuals at time t.

The total population size is conserved:

$$N = S(t) + I(t) + R(t).$$

Model outputs:

The SIR model declares the following outputs:

`"S"` Susceptible population size.

`"I"` Infectious population size.

`"R"` Recovered (removed) population size.

`"incidence"` Instantaneous rate of new infections $\lambda(t)$ returned by the model's right-hand side.

All declared outputs may be used as observables in generic utilities such as `fit_epi_model` via the `target` argument.

Parameters:

The SIR model depends on the following parameters:

beta Transmission rate (per day).

gamma Recovery/removal rate (per day).

Model equations:

New infections occur at rate

$$\lambda(t) = \beta \frac{S(t)\, I(t)}{N}.$$

The system of ordinary differential equations is:

$$\begin{aligned} \frac{dS}{dt} &= -\lambda(t), \\ \frac{dI}{dt} &= \lambda(t) - \gamma I(t), \\ \frac{dR}{dt} &= \gamma I(t). \end{aligned}$$

Usage:

This predefined model object is intended to be used with generic utilities such as `simulate_epi`, `fit_epi_model`, and `predict.fit_epi_model` that operate on `epi_model` objects.

See Also

`simulate_epi`, `fit_epi_model`, `new_epi_model`

Examples

```
## Simulate a SIR epidemic
sim <- simulate_epi(
  model = SIR_MODEL,
  times = 0:200,
  parms = c(beta = 0.3, gamma = 0.1),
  init  = c(S = 1e6, I = 10, R = 0)
)

plot(sim)

## Plot observed incidence (if an observation model is used)
plot(sim, what = "incidence")

## Fit the model to observed incidence
fit_inc <- fit_epi_model(
  x = sim$incidence$inc,
  model = SIR_MODEL,
  init = SIR_MODEL$init,
  target = "incidence"
)

fit_inc
```

SI_MODEL *SI epidemic model*

Description

An `epi_model` object representing a deterministic **SI** (Susceptible–Infectious) compartmental epidemic model.

The model describes the spread of an infection in a closed population where individuals move irreversibly from the susceptible compartment `S` to the infectious compartment `I`. No recovery or removal process is included.

Usage

```
SI_MODEL
```

Format

An object of class `"epi_model"`.

Details

State variables:

The model is defined in terms of the following state variables:

S(t) Number of susceptible individuals at time t.

I(t) Number of infectious individuals at time t.

The total population size is conserved:

$$N = S(t) + I(t).$$

Model outputs:

The SI model declares the following outputs:

`"S"` Susceptible population size.

`"I"` Infectious population size.

`"incidence"` Instantaneous rate of new infections $\lambda(t)$ returned by the model's right-hand side.

All declared outputs may be used as observables in generic utilities such as `fit_epi_model` via the `target` argument.

Parameters:

The SI model depends on a single parameter:

beta Transmission rate (per day).

Model equations:

New infections occur at rate

$$\lambda(t) = \beta \frac{S(t)\, I(t)}{N}.$$

The system of ordinary differential equations is:

$$\frac{dS}{dt} = -\lambda(t),$$
$$\frac{dI}{dt} = \lambda(t).$$

Usage:

This predefined model object is intended to be used with generic utilities such as `simulate_epi`, `fit_epi_model`, and `predict.fit_epi_model` that operate on `epi_model` objects.

See Also

`simulate_epi`, `fit_epi_model`, `new_epi_model`

Examples

```
## Simulate an SI epidemic
sim <- simulate_epi(
  model = SI_MODEL,
  times = 0:100,
  parms = c(beta = 0.4),
  obs = "negbin",
  init = SI_MODEL$init
)

plot(sim)
```

```
## Plot observed incidence
plot(sim, what = "incidence")

## Fit the model to observed incidence
fit_inc <- fit_epi_model(
  x = sim$incidence$inc,
  model = SI_MODEL,
  init = SI_MODEL$init,
  target = "incidence"
)

fit_inc
```

`summary.sim_epi` *Summarize a simulated epidemic*

Description

Summary method for objects of class `"sim_epi"` as returned by `simulate_epi`. The function computes and returns a small set of epidemiologically meaningful summary quantities derived from the simulation.

Usage

```
## S3 method for class 'sim_epi'
summary(object, ...)
```

Arguments

`object`	Object of class `"sim_epi"` as returned by `simulate_epi`.
`...`	Currently unused; included for compatibility with generic `summary`.

Details

The summary includes:

model The epidemic model simulated (e.g. `"sir"` or `"sirs"`).

R0 The basic reproduction number, computed as $R_0 = \beta/\gamma$.

peak_I The maximum number of infectious individuals observed during the simulation.

total_infections The total number of infection events, computed as the maximum value of the cumulative infection variable `C(t)`.

This method is automatically dispatched when calling `summary()` on an object of class `"sim_epi"`.

Value

A named list with summary statistics describing the simulated epidemic.

See Also

simulate_epi, plot.sim_epi

Examples

```
sim <- simulate_epi(n_days = 300, model = SIRS_MODEL, omega = 1/180, seed = 1)

summary(sim)
```

Index