Package: shinySIR (via r-universe)

August 24, 2024

Type Package

Title Interactive Plotting for Mathematical Models of Infectious Disease Spread

Version 0.1.2

Date 2020-09-30

Maintainer Sinead E. Morris <sinead.morris@columbia.edu>

Description Provides interactive plotting for mathematical models of infectious disease spread. Users can choose from a variety of common built-in ordinary differential equation (ODE) models (such as the SIR, SIRS, and SIS models), or create their own. This latter flexibility allows 'shinySIR' to be applied to simple ODEs from any discipline. The package is a useful teaching tool as students can visualize how changing different parameters can impact model dynamics, with minimal knowledge of coding in R. The built-in models are inspired by those featured in Keeling and Rohani (2008) <doi:10.2307/j.ctvcm4gk0> and Bjornstad (2018) <doi:10.1007/978-3-319-97487-3>.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

Depends dplyr (>= 0.8.0.1), tidyr (>= 0.8.3), ggplot2 (>= 3.1.1), shiny (>= 1.3.2), deSolve (>= 1.2.1)

Suggests knitr (>= 1.22), rmarkdown (>= 1.12), testthat (>= 2.2.0)

VignetteBuilder knitr

NeedsCompilation no

Author Sinead E. Morris [aut, cre] (<https://orcid.org/0000-0001-8626-1698>), Ottar N. Bjornstad [ctb]

Repository CRAN

Date/Publication 2020-10-18 18:50:02 UTC

Contents

`default_models` *Model help*

Description

This function prints a list of all built in models, along with their parameter arguments and a short description.

Usage

```
default_models()
```

Value

data frame of model descriptions.

Examples

```
default_models()
```

get_ics *Get default initial conditions*

Description

This function returns the default parameter vectors for a particular built-in model.

Usage

```
get_ics(model)
```

Arguments

model	name of the model to be solved. Examples include: SIR and SIR vaccination.

Value

named vector of default initial conditions.

get_name *Get model display names*

Description

This function gets the display name for any built-in model.

Usage

```
get_name(model)
```

Arguments

model	character specifying the name of the built-in model.

Value

character of the corresponding display name.

Examples

```
get_name(model = "SIR")
```

get_params *Get default parameters*

Description

This function returns the default parameter vectors for a particular built-in model.

Usage

```
get_params(model)
```

Arguments

model	name of the model to be solved. Examples include: SIR and SIR vaccination.

Value

list of default parameter vectors.

Examples

```
get_params(model = "SIR")
```

plot_model *Plot model output. This function plots the output of a fitted model data frame.*

Description

Plot model output. This function plots the output of a fitted model data frame.

Usage

```
plot_model(output, linesize, textsize, xlabel, ylabel, legend_title,
  levels, values, ...)
```

Arguments

output	data frame output from solve_eqns().
linesize	numeric value for line width in ggplot.
textsize	numeric value for textsize in ggplot.
xlabel	character string for x axis label.
ylabel	character string for y axis label.
legend_title	character string for legend title.

levels	character vector of the variable names in the order they should be plotted. Default is to obtain the order from the initial conditions vector 'ics'.
values	vector specifying manual color scale. Length must equal the number of model variables.
...	extra argument to be passed through to ggplot scale_colour_manual: use 'labels' to change the legend names

Value

ggplot object

`run_shiny` *Solve equations*

Description

This function solves an ODE model using 'deSolve' and returns the output as a data frame.

Usage

```
run_shiny(model = "SIR", neweqns = NULL, ics = NULL, tstart = 0,
  timestep = 1, tmax = 365, parm0 = NULL, parm_names = NULL,
  parm_min = NULL, parm_max = NULL, sigfigs = 4, showtable = TRUE,
  linesize = 1.2, textsize = 14, xlabel = "Time",
  ylabel = "Number of individuals", legend_title = "Compartment",
  slider_steps = NULL, values = NULL, ...)
```

Arguments

model	name of the model to be solved. Examples of built-in models are: "SIR", "SIR vaccination". Default is "SIR".
neweqns	function specifying the equations of the user-defined model. Only to be used if a model is required that is not built-in. Default is NULL.
ics	named numeric vector specifying the initial conditions i.e. the initial values of all model variables. Default is c(S = 9999, I = 1, R = 0) for the SIR model.
tstart	numerical value of form c(tmin, tmax) indicating the time to start simulations. Default value is 0.
timestep	numerical value indicating time step be used when solving equations. Default value is 1/365.
tmax	numerical value indicating maximum time point to be considered.
parm0	named numeric vector of starting parameter values. Names must correspond to those used in the model equations.
parm_names	character vector of parameter names to be displayed in shiny menu. Must be in the same order as 'parm0'.

parm_min	named numeric vector of minimum parameter values.
parm_max	named numeric vector of maximum parameter values.
sigfigs	number of significant figures to round parameter input vectors. Default is 4.
showtable	logical TRUE/FALSE. Should the table of transformed parameters be shown? Only applies to built-in models. Default is TRUE.
linesize	numeric value for line width in ggplot output. Default is 1.2.
textsize	numeric value for textsize in ggplot output. Default is 14.
xlabel	character string for x axis plotting label. Default is "Time".
ylabel	character string for y axis plotting label. Default is "Number of individuals".
legend_title	character string for legend title. Default is "Compartment".
slider_steps	numeric vector of step size to include between slider input values. Should be NULL or a vector with an entry for each parameter input. Default is NULL.
values	vector specifying manual color scale (if desired). Length must equal the number of model variables.
...	extra argument to be passed through to ggplot scale_colour_manual: use 'labels' to change the legend names.

Value

data frame of model solutions in long format.

Examples

```
run_shiny(model = "SIR")
```

`seir.app` *Launch a shiny-app simulating the seasonal SEIR model*

Description

#' This launches an app running the SEIR model i.e. a model incorporating latency and seasonal forcing in transmission.

Usage

```
seir.app
```

Format

An object of class `shiny.appobj` of length 5.

Details

Launch app for details

Examples

```
## Not run: seir.app
```

`seirs.app` *Launch a shiny-app simulating the SEIRS model*

Description

This launches an app running the SEIRS model i.e. a model incorporating latency and loss of immunity.

Usage

```
seirs.app
```

Format

An object of class `shiny.appobj` of length 5.

Details

Launch app for details

Examples

```
## Not run: seirs.app
```

`SIR` *SIR model*

Description

These equations describe the classic SIR model with no births or deaths.

Usage

```
SIR(t, y, parms)
```

Arguments

`t`	numeric vector of time points.
`y`	numeric vector of variables.
`parms`	named vector of model parameters.

Value

equation list

SIRbirths *SIR model with demography*

Description

These equations describe the classic SIR model with equal births and deaths.

Usage

```
SIRbirths(t, y, parms)
```

Arguments

t	numeric vector of time points.
y	numeric vector of variables.
parms	named vector of model parameters.

Value

equation list

SIRS *SIRS model*

Description

These equations describe the classic SIRS model without births or deaths.

Usage

```
SIRS(t, y, parms)
```

Arguments

t	numeric vector of time points.
y	numeric vector of variables.
parms	named vector of model parameters.

Value

equation list

SIRSbirths *SIRS model with demography*

Description

These equations describe the classic SIRS model with equal birth and death rates.

Usage

```
SIRSbirths(t, y, parms)
```

Arguments

t	numeric vector of time points.
y	numeric vector of variables.
parms	named vector of model parameters.

Value

equation list

SIRSvaccination *SIRS model with vaccination at birth*

Description

These equations describe the classic SIRS model with equal birth and death rates and vaccination at birth.

Usage

```
SIRSvaccination(t, y, parms)
```

Arguments

t	numeric vector of time points.
y	numeric vector of variables.
parms	named vector of model parameters.

Value

equation list

SIRvaccination *SIR model with vaccination at birth*

Description

These equations describe the classic SIR model with births and deaths, constant population size, and (optional) vaccination at birth.

Usage

```
SIRvaccination(t, y, parms)
```

Arguments

t	numeric vector of time points
y	numeric vector of variables
parms	named vector of model parameters.

Value

equation list

SIS *SIS model*

Description

These equations describe the classic SIS model with no births or deaths.

Usage

```
SIS(t, y, parms)
```

Arguments

t	numeric vector of time points
y	numeric vector of variables
parms	named vector of model parameters.

Value

equation list

SISbirths *SIS model with demography*

Description

These equations describe the classic SIR model with equal births and deaths.

Usage

```
SISbirths(t, y, parms)
```

Arguments

t	numeric vector of time points.
y	numeric vector of variables.
parms	named vector of model parameters.

Value

equation list

solve_eqns *Solve equations*

Description

This function solves an ODE model using 'deSolve' and returns the output as a data frame.

Usage

```
solve_eqns(eqns, ics, times, parms)
```

Arguments

eqns	name of the model to be solved. Examples include: SIR and SIR vaccination.
ics	named numeric vector specifying the initial conditions i.e. the initial values of all model variables.
times	numerical vector indicating the time points at which the equation should be solved.
parms	named numeric vector of parameter values.

Value

data frame of model solutions in long format.

Index