Package: epifitter (via r-universe)

September 17, 2024

Type Package

Title Analysis and Simulation of Plant Disease Progress Curves

Version 0.3.0

Description Analysis and visualization of plant disease progress curve data. Functions for fitting two-parameter population dynamics models (exponential, monomolecular, logistic and Gompertz) to proportion data for single or multiple epidemics using either linear or no-linear regression. Statistical and visual outputs are provided to aid in model selection. Synthetic curves can be simulated for any of the models given the parameters. See Laurence V. Madden, Gareth Hughes, and Frank van den Bosch (2007) <doi:10.1094/9780890545058> for further information on the methods.

Maintainer Kaique dos S. Alves <kaiquedsalves@gmail.com>

Depends R (>= 3.2)

Imports deSolve, dplyr, stats, ggplot2, knitr, tidyr, DescTools, minpack.lm, magrittr, tibble

Suggests rmarkdown, ggridges, cowplot

License MIT + file LICENSE

Encoding UTF-8

LazyData true

VignetteBuilder knitr

Date 2021-06-13 22:30:00 UTC

URL https://github.com/AlvesKS/epifitter

BugReports https://github.com/AlvesKS/epifitter/issues

X-schema.org-applicationCategory Tools

Repository https://alvesks.r-universe.dev

RemoteUrl https://github.com/alvesks/epifitter

RemoteRef HEAD

RemoteSha 00af96a8a8c73ef15f548e3850dc42791d2d52b4

Contents

`AUDPC` *Area under disease progress curve*

Description

Calculates the area under disease progress curves.

Usage

```
AUDPC(time, y, y_proportion = TRUE, type = "absolute")
```

Arguments

`time`	A vector object of time.
`y`	A vector object of disease intensity.
`y_proportion`	Logical. If disease intensity is proportion (TRUE) or percentage(FALSE).
`type`	Set if is absolute or relative AUDPC. type = "absolute" is default.

Author(s)

Kaique dos S. Alves

References

Madden, L. V., Hughes, G., and van den Bosch, F. 2007. The Study of Plant Disease Epidemics. American Phytopathological Society, St. Paul, MN.

Examples

```
epi =  sim_logistic(N = 30, y0 = 0.01,dt = 5, r = 0.3, alpha = 0.5, n = 1)
AUDPC(time = epi$time, y = epi$y, y_proportion = TRUE)
```

AUDPS *Area under disease progress stairs*

Description

Calculates the area under disease progress stairs.

Usage

```
AUDPS(time, y, y_proportion = TRUE, type = "absolute")
```

Arguments

time	A vector object of time.
y	A vector object of disease intensity.
y_proportion	Logical. If disease intensity is proportion (TRUE) or percentage(FALSE)
type	Set if is absolute or relative AUDPC. type = "absolute" is default.

Author(s)

Kaique dos S. Alves

References

Simko, I., and Piepho, H.-P. 2012. The area under the disease progress stairs: Calculation, advantage, and application. Phytopathology 102:381- 389.

Examples

```
epi =  sim_logistic(N = 30, y0 = 0.01,dt = 5, r = 0.3, alpha = 0.5, n = 1)
AUDPS(time = epi$time, y = epi$y, y_proportion = TRUE)
```

expo_fun *Function for Exponential model*

Description

Base function for the Exponential model. This function is used in the Exponential model simulation function `sim_exponential()`

Usage

```
expo_fun(t, y, par)
```

Arguments

`t`	Vector of time
`y`	Vector of disease intensity
`par`	List of parameters

fit_lin *Fits epidemic models using data linearization*

Description

Fits epidemic models (Exponential, Monomolecular, Logistic and Gompertz) to data using data linearization

Usage

```
fit_lin(time,y)
```

Arguments

`time`	Numeric vector which refers to the time steps in the epidemics
`y`	Numeric vector which refers to the disease intensity

Author(s)

Kaique dos S. Alves

Examples

```
set.seed(1)
epi1 <- sim_logistic(N = 30,
                     y0 = 0.01,
                     dt = 5,
                     r = 0.3,
                     alpha = 0.2,
                     n = 4)
data = data.frame(time =  epi1[,2], y = epi1[,4])
fit_lin( time = data$time, y =  data$y)
```

`fit_multi` *Estimate model parameters for multiple disease progress curves*

Description

Estimate model parameters for multiple disease progress curves

Usage

```
fit_multi(time_col,
             intensity_col,
             data,
             strata_cols ,
             starting_par = list(y0 = 0.01, r = 0.03, K =  0.8),
             maxiter=500,
             nlin = FALSE,
             estimate_K = FALSE)
```

Arguments

`time_col`	Character name specifying the column for the time. eg: time_col = "days".
`intensity_col`	Character name specifying the column for the disease intensity.
`data`	`data.frame` object containing the variables for model fitting.
`strata_cols`	Character name or vector specifying the columns for stratification.
`starting_par`	Starting value for initial inoculun (y0) and apparent infection rate (r). Please informe in that especific order
`maxiter`	Maximum number of iterations. Only used if is `nlin = TRUE`
`nlin`	Logical. If `FALSE` estimates parameters using data linearization. If `nlin=TRUE`, estimates nonlinear approach. `fit_nlin`.
`estimate_K`	Logical. If `nlin=TRUE`, estimates maximum disease intensity. (default: `nlin=FALSE`) `fit_nlin2`.

Value

Returns a `data.frame` containing estimated parameters for individual strata levels.

See Also

fit_lin,fit_nlin,fit_nlin2

Examples

```
set.seed(1)
# create stratified dataset
data_A1 = sim_gompertz(N = 30, y0 = 0.01,dt = 5, r = 0.3, alpha = 0.5, n = 4)
data_A1 = dplyr::mutate(data_A1,
                        fun = "A",
                        cultivar = "BR1")
set.seed(1)
data_B1 = sim_gompertz(N = 30, y0 = 0.01, dt = 5, r = 0.2, alpha = 0.5, n = 4)
data_B1 = dplyr::mutate(data_B1,
                        fun = "B",
                        cultivar = "BR1")
set.seed(1)
data_A2 = sim_gompertz(N = 30, y0 = 0.01,dt = 5, r = 0.1, alpha = 0.5, n = 4)
data_A2 = dplyr::mutate(data_A2,
                        fun = "A",
                        cultivar = "BR2")
set.seed(1)
data_B2 = sim_gompertz(N = 30, y0 = 0.01,dt = 5, r = 0.1, alpha = 0.5, n = 4)
data_B2 = dplyr::mutate(data_B2,
                        fun = "B",
                        cultivar = "BR2")

data = dplyr::bind_rows(data_A1, data_B1,data_A2, data_B2)

fit_multi(time_col = "time",
             intensity_col = "random_y",
             data = data,
             strata_col = c("fun","cultivar"),
             starting_par = list(y0 = 0.01, r = 0.03),
             maxiter = 1024,
             nlin = FALSE,
             estimate_K = FALSE)
```

`fit_nlin` *Fits epidemic models using nonlinear aproach*

Description

Fits epidemic models (Exponential, Monomolecular, Logistic and Gompertz) using nonlinear approach for estimate parameters.

Usage

```
fit_nlin(time,
  y,
  starting_par = list(y0 = 0.01, r = 0.03),
  maxiter = 50)
```

Arguments

time	Numeric vector which refers to the time steps in the epidemics
y	Numeric vector which refers to the disease intensity
starting_par	Starting value for initial inoculun (y0) and apparent infection rate (r). Please informe in that especific order
maxiter	Maximun number of iterations

Author(s)

Kaique dos S. Alves

Examples

```
set.seed(1)
epi1 <- sim_logistic(N = 30,
                     y0 = 0.01,
                     dt = 5,
                     r = 0.3,
                     alpha = 0.5,
                     n = 4)
data = data.frame(time =  epi1[,2], y = epi1[,4])
fit_nlin(time = data$time, y =  data$y, starting_par = list(y0 = 0.001, r = 0.03), maxiter = 1024)
```

---

fit_nlin2 *Fits epidemic models using nonlinear aproach. This function also estimates the maximum disease intensity parameter K*

Description

Fits epidemic models (Exponential, Monomolecular, Logistic and Gompertz) using nonlinear approach for estimate parameters. This function also estimates the maximum disease intensity parameter K.

Usage

```
fit_nlin2(time,
  y,
  starting_par = list(y0 = 0.01, r = 0.03, K =  0.8),
  maxiter = 50)
```

Arguments

time	Numeric vector which refers to the time steps in the epidemics.
y	Numeric vector which refers to the disease intensity.
starting_par	starting value for initial inoculun (y0) and apparent infection rate (r), and maximum disease intensity (K). Please informe in that especific order
maxiter	Maximun number of iterations.

Examples

```
set.seed(1)

epi1 <- sim_logistic(N = 30,
                     y0 = 0.01,
                     dt = 5,
                     r = 0.3,
                     alpha = 0.5,
                     n = 4)

data = data.frame(time =  epi1[,2], y = epi1[,4])
fit_nlin2(time = data$time,
          y =  data$y,
          starting_par = list(y0 = 0.01, r = 0.03, K = 1),
          maxiter = 1024)
```

---

gompi_fun *Function for Gompertz model*

Description

Base function for the Gompertz model. This function is used in the Gompertz model simulation function `sim_gompertz()`

Usage

```
gompi_fun(t, y, par)
```

Arguments

t	Vector of time
y	Vector of disease intensity
par	List of parameters

logi_fun *Function for logistic model*

Description

Base function for the Logistic model. This function is used in the Logistic model simulation function `sim_logistic()`

Usage

```
logi_fun(t, y, par)
```

Arguments

`t`	Vector of time
`y`	Vector of disease intensity
`par`	List of parameters

mono_fun *Function for Monomolecular model*

Description

Base function for the Monomolecular model. This function is used in the Monomolecular model simulation function `sim_monomolecular()`

Usage

```
mono_fun(t, y, par)
```

Arguments

`t`	Vector of time
`y`	Vector of disease intensity
`par`	List of parameters

plot_fit *Creates a plot panel for the fitted models*

Description

Create a ggplot2-style plot with the fitted models curves and the epidemic data.

Usage

```
plot_fit(object,
  point_size =1.2,
  line_size = 1,
  models = c("Exponential","Monomolecular", "Logistic", "Gompertz"))
```

Arguments

object	A fit_lin or a fit_nlin object
point_size	Point size
line_size	Line size
models	Select the models to be displayed in the panel

Details

It is possible to add more ggplot components by using the + syntax. See examples below.

Examples

```
epi1 <- sim_logistic(N = 30,
                     y0 = 0.01,
                     dt = 5,
                     r = 0.3,
                     alpha = 0.5,
                     n = 4)
data = data.frame(time =  epi1[,2], y = epi1[,4])
fitted = fit_lin( time = data$time, y =  data$y)
plot_fit(fitted)

# adding ggplot components
library(ggplot2)
plot_fit(fitted)+
  theme_minimal()+
  ylim(0,1)+
  labs(y = "Disease internsity", x = "Time")
```

PowderyMildew *Dataset powdery mildew disease progress curves*

Description

Dataset containing experimental data of disease progress curves of powdery mildew under different irrigation systems and soil moisture levels in organic tomato.

Usage

```
data("PowderyMildew")
```

Format

A data frame with 240 observations on the following 2 variables.

`irrigation_type` Irrigations Systems: MS = Micro Sprinkler

`moisture` Levels of soils moisture

`block` Experimental blocks

`time` a numeric vector containing the time points

`sev` a numeric vector containg disease severity data in proportinal scales

References

Lage, D. A. C., Marouelli, W. A., and Café-Filho, A. C. 2019. Management of powdery mildew and behaviour of late blight under different irrigation configurations in organic tomato. Crop Protection. 125:104886.

Examples

```
data(PowderyMildew)
## maybe str(PowderyMildew) ; plot(PowderyMildew) ...
```

print.fit_lin *Print* `fit_lin()` *or* `fit_nlin()` *outputs*

Description

The `print` method for density objects.

Usage

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

Arguments

x	output from fit_lin() or fit_nlin()
...	...

---

print.fit_nlin2 *Print* fit_nlin2() *outputs*

Description

The print method for density objects.

Usage

```
## S3 method for class 'fit_nlin2'
print(x, ...)
```

Arguments

x	output from fit_nlin2()
...	...

---

sim_exponential *Simulate an epidemic using the Exponential model*

Description

Simulate a stochastic epidemic curve using the Exponential model.

Usage

```
sim_exponential(N = 10,dt = 1, y0 = 0.01, r, n,  alpha = 0.2)
```

Arguments

N	Total time course of the epidemic
dt	Time step
y0	Initial inoculum or initial disease intensity
r	Infection rate
n	Number or replicates or sample size for each time step
alpha	Variation parameter. stands for the variation for the replicates for each time step. The standard deviation is calculated as sd = alpha * y * (1 - y), being y the disease intensity for each time step.

Value

rep	Replicates
time	Time after epidemic start
y	Disease intensity
random_y	Disease intensity after applying the random alpha error

Examples

```
sim_exponential(N = 30, y0 = 0.01,dt = 5, r = 0.1, alpha = 0.5, n = 4)
```

---

sim_gompertz *Simulate an epidemic using the Gompertz model*

Description

Simulate a stochastic epidemic curve using the Gompertz model.

Usage

```
sim_gompertz(N = 10,dt = 1, y0 = 0.01, r, K = 1, n,  alpha = 0.2)
```

Arguments

N	Total time course of the epidemic
dt	Time step
y0	Initial inoculum or initial disease intensity
r	Infection rate
K	Maximum asymptote
n	Number or replicates or sample size for each time step
alpha	Variation parameter. stands for the variation for the replicates for each time step. The standard deviation is calculated as sd = alpha * y * (1 - y), being y the disease intensity for each time step.

Value

rep	Replicates
time	Time after epidemic start
y	Disease intensity
random_y	Disease intensity after applying the random alpha error

Examples

```
sim_gompertz(N = 30, y0 = 0.01,dt = 5, r = 0.3, K = 1, alpha = 0.5, n = 4)
```

`sim_logistic` *Simulate an epidemic using the logistic model*

Description

Simulate a stochastic epidemic curve using the logistic model.

Usage

```
sim_logistic(N = 10,dt = 1, y0 = 0.01, r, K = 1, n,  alpha = 0.2)
```

Arguments

`N`	Total time course of the epidemic
`dt`	Time step
`y0`	Initial inoculum or initial disease intensity
`r`	Infection rate
`K`	Maximum asymptote
`n`	Number or replicates or sample size for each time step
`alpha`	Variation parameter. stands for the variation for the replicates for each time step. The standard deviation is calculated as sd = alpha * y * (1 - y), being y the disease intensity for each time step.

Value

`rep`	Replicates
`time`	Time after epidemic start
`y`	Disease intensity
`random_y`	Disease intensity after applying the random `alpha` error

Examples

```
sim_logistic(N = 30, y0 = 0.01,dt = 5, r = 0.3, K = 1, alpha = 0.5, n = 4)
```

`sim_monomolecular` *Simulate an epidemic using the Monomolecular model*

Description

Simulate a stochastic epidemic curve using the Monomolecular model.

Usage

```
sim_monomolecular(N = 10,dt = 1, y0 = 0.01, r,K = 1, n,  alpha = 0.2)
```

Arguments

`N`	Total time course of the epidemic
`dt`	Time step
`y0`	Initial inoculum or initial disease intensity
`r`	Infection rate
`K`	Maximum asymptote
`n`	Number or replicates or sample size for each time step
`alpha`	Variation parameter. stands for the variation for the replicates for each time step. The standard deviation is calculated as sd = alpha * y * (1 - y), being y the disease intensity for each time step.

Value

`rep`	Replicates
`time`	Time after epidemic start
`y`	Disease intensity
`random_y`	Disease intensity after applying the random `alpha` error

Examples

```
sim_monomolecular(N = 30, y0 = 0.01,dt = 5, r = 0.3, K = 1, alpha = 0.5, n = 4)
```

Index