Package 'nlstimedist'

May 15, 2019

Type Package

Title Non-Linear Model Fitting of Time Distribution of Biological Phenomena

Version 1.1.4

Description Fit biologically meaningful distribution functions to time-sequence data (phenology), estimate parameters to draw the cumulative distribution function and probability density function and calculate standard statistical moments and percentiles.

URL https://github.com/nathaneastwood/nlstimedist

BugReports https://github.com/nathaneastwood/nlstimedist/issues

Depends R (>= 3.5.0)

Imports broom (>= 0.5.0), dplyr (>= 0.4.3), ggplot2 (>= 2.1.0), lazyeval (>= 0.2.0), minpack.lm (>= 1.2-0), nlstools (>= 1.0-2)

License GPL-2

LazyData TRUE

RoxygenNote 6.1.1

Suggests knitr, rmarkdown, testthat

VignetteBuilder knitr

NeedsCompilation no

Author Nathan Eastwood [cre, prg, trl],
Miguel Franco [aut],
Paul Ramsay [aut],
Nicola Steer [aut]

Maintainer Nathan Eastwood <nathan.eastwood@icloud.com>

Repository CRAN

Date/Publication 2019-05-15 21:40:03 UTC

R topics documented:

augmentMultiple . . . 2
glance.timedist . . . 2
lobelia . . . 3
nlstimedist . . . 4
pupae . . . 4
tdCdfPlot . . . 5
tdData . . . 5
tdMoments . . . 6
tdPDF . . . 7
tdPercentiles . . . 8
tdRSS . . . 8
tilia . . . 9
timedist . . . 9

Index **11**

`augmentMultiple` *Create the data for the plots*

Description

Augment the data from a model output to be in a form suitable for ggplot

Usage

```
augmentMultiple(...)
```

Arguments

`...`	A list of models

`glance.timedist` *Construct a single row summary "glance" of a timedist model*

Description

glance methods always return either a one-row data frame, or NULL

Usage

```
## S3 method for class 'timedist'
glance(x, ...)
```

Arguments

x	An object of class timedist.
...	Additional arguments (not used).

Value

glance returns one row with the columns

sigma	the square root of the estimated residual variance
isConv	whether the fit successfully converged
finTol	the achieved convergence tolerance
logLik	the data's log-likelihood under the model
AIC	the Akaike Information Criterion
BIC	the Bayesian Information Criterion
deviance	deviance
df.residual	residual degrees of freedom
RSS	corrected residual sum of squares

---

lobelia *Lobelia urens seeds data*

Description

This data describes the number of germinating lobelia urens seeds at different temperatures.

Usage

```
lobelia
```

Format

A data frame with 231 rows and 3 variables:

Day The day number

Temperature The temperature

Germination The number which germinated

Details

The total numbers which failed to germinate are 59, 52, 35, 22, 10, 7 and 12 for temperatures 9.8, 12.5, 16.7, 20.2, 24.3, 28.5 and 32.0, respectively.

Examples

```
lobelia
```

nlstimedist *Fit the time-course of biological phenomena*

Description

nlstimedist fits a biologically meaningful distribution function to time-sequence data (phenology), estimates parameters to draw the cumulative distribution function and probability density function and calculates standard statistical moments and percentiles.

pupae *Emergence of butterflies data*

Description

This data describes the emergence of butterflies from their pupae from four different cohorts.

Usage

```
pupae
```

Format

A data frame with 64 rows and 3 variables:

Day The day number

Cohort The cohort number

Emergence The number of butterflies to emerge

Examples

```
pupae
```

tdCdfPlot *Plot the timedist PDF or CDF*

Description

Given a model (or models) of class `timedist`, produce a cumulative distribution plot for each of them.

Usage

```
tdCdfPlot(..., S = NULL, xVals = NULL)

tdPdfPlot(..., S = NULL, xVals = NULL)
```

Arguments

`...`	A model (or a list of models) of class `timedist`.
`S`	Scaling factor for the PDF.
`xVals`	A sequence of values between the x limits (x1, x2) of the plot.

Examples

```
tdTilia <- tdData(tilia, x = "Day", y = "Trees")
model <- timedist(data = tdTilia, x = "Day", y = "propMax", r = 0.1, c = 0.5,
                  t = 120)
tdCdfPlot(model)

tdPdfPlot(model)
```

tdData *Prepare* `nlstimedist` *data*

Description

The data for `nlstimedist` needs to be in a particular format. This function prepares the data for the model.

Usage

```
tdData(data, x, y, group = NULL)
```

Arguments

data	The raw data to be cleaned.
x	The time variable.
y	The number of events.
group	The run numbers. This is NULL by default if you are only using the function for one run.

Value

A list containing

- raw The raw data supplied to the function, i.e. data.
- clean item The cleaned data to be supplied to timedist.

Examples

```
tdTilia <- tdData(tilia, x = "Day", y = "Trees")
tdTilia
```

---

tdMoments *Calculate moments for the fitted timedist model*

Description

Individual functions are provided as well as a wrapper to calculate the moments for your fitted model.

Usage

```
tdMoments(r, c, t, ...)

tdMean(r, c, t, upper = t * 10, ...)

tdVariance(r, c, t, upper = t * 10, ...)

tdSkew(r, c, t, upper = t * 10, ...)

tdKurtosis(r, c, t, upper = t * 10, alternative = FALSE, ...)

tdEntropy(r, c, t, upper = t * 10, ...)
```

Arguments

r, c, t	Parameters of the Franco distribution
...	Additional arguments to be passed to integrate
upper	The upper limit of integration. Defaults to t * 10. Can be infinite for all moment functions except for entropy.
alternative	An alternative calculation method.

Value

A single value, or in the case of tdMoments, a data.frame of values.

Examples

```
tdMoments(r = 0.1, c = 0.5, t = 120)

tdMean(r = 0.1, c = 0.5, t = 120)
tdVariance(r = 0.1, c = 0.5, t = 120)
tdSkew(r = 0.1, c = 0.5, t = 120)
tdKurtosis(r = 0.1, c = 0.5, t = 120)
tdKurtosis(r = 0.1, c = 0.5, t = 120, alternative = TRUE)
tdEntropy(r = 0.1, c = 0.5, t = 120)
```

---

tdPDF *Calculate the PDF and CDF*

Description

Calculate values of the probability density function.

Calculate values of the cumulative distribution function

Usage

```
tdPDF(x, S = 1, r, c, t)

tdCDF(x, S = 1, r, c, t)
```

Arguments

x	Points at which to calculate the the pdf.
S	Scaling factor for the PDF.
r, c, t	Parameter values within the model.

Value

A vector of values from the pdf.

A vector of values from the cdf.

tdPercentiles *Calculate percentiles*

Description

Calculate the percentiles for a given model output

Usage

```
tdPercentiles(model, n, upper = model$m$getPars()["t"] * 10, ...)
```

Arguments

model	An object of class timedist.
n	A vector of percentiles to be calculated.
upper	The upper end point of the interval to search.
...	Additional parameters to be passed to uniroot.

Examples

```
tdTilia <- tdData(tilia, x = "Day", y = "Trees")
model <- timedist(data = tdTilia, x = "Day", y = "propMax", r = 0.1, c = 0.5,
                  t = 120)
model
tdPercentiles(model, n = 0.5)
tdPercentiles(model, n = seq(0, 0.9, 0.1))
```

tdRSS *Calculate the corrected residual sum of squares*

Description

Calculate the corrected residual sum of squares for a model of class timedist.

Usage

```
tdRSS(model)
```

Arguments

model	An object of class timedist.

Value

A single value.

Examples

```
tdTilia <- tdData(tilia, x = "Day", y = "Trees")
model <- timedist(data = tdTilia, x = "Day", y = "propMax", r = 0.1, c = 0.5,
                  t = 120)
model
tdRSS(model)
```

`tilia` *Leafing phenology of tilia cordata*

Description

This data describes the leafing phenology of lime trees (tilia cordata).

Usage

```
tilia
```

Format

A data frame with 34 rows and 2 variables:

Day The day number

Trees The number of trees

Examples

```
tilia
```

`timedist` *Fit the Franco model*

Description

Fit the Franco model

Usage

```
timedist(data, x, y, r, c, t, ...)
```

Arguments

`data`	The data to be included in the model.
`x, y`	The x and y values in the data, where the y values are the proportions.
`r, c, t`	The starting parameters for the model.
`...`	Additional parameters to be passed to `nlsLM`.

Details

The nlsLM function is used instead of the nls function in order to use the Levenberg-Marquardt algorithm which is an extremely robust method of curve-fitting as it is able to switch between Gauss-Newton and gradient descent. This allows it to cope with far-off-optimal starting values. The standard nls function does not use Levenberg-Marquardt; it instead uses the Gauss-Newton type, the PORT routines and a partial linear fit.

Examples

```
tdTilia <- tdData(tilia, x = "Day", y = "Trees")
model <- timedist(data = tdTilia, x = "Day", y = "propMax", r = 0.1, c = 0.5,
                  t = 120)
model
```

Index

augmentMultiple, 2

glance.timedist, 2

integrate, *7*

lobelia, 3

nls, *10*
nlsLM, *9*, *10*
nlstimedist, 4
nlstimedist-package *(*nlstimedist*)*, 4

pupae, 4

tdCDF *(*tdPDF*)*, 7
tdCdfPlot, 5
tdData, 5
tdEntropy *(*tdMoments*)*, 6
tdKurtosis *(*tdMoments*)*, 6
tdMean *(*tdMoments*)*, 6
tdMoments, 6
tdPDF, 7
tdPdfPlot *(*tdCdfPlot*)*, 5
tdPercentiles, 8
tdRSS, 8
tdSkew *(*tdMoments*)*, 6
tdVariance *(*tdMoments*)*, 6
tilia, 9
timedist, 9

uniroot, *8*