Package ‘morse’

June 12, 2018

Type Package

Title Modelling Tools for Reproduction and Survival Data in Ecotoxicology

Version 3.1.1

Author Virgile Baudrot [aut], Sandrine Charles [aut], Marie Laure Delignette-Muller [aut], Wandrille Duchemin [ctb], Guillaume Kon-Kam-King [ctb], Christelle Lopes [ctb], Philippe Ruiz [aut], Philippe Veber [aut]

Maintainer Philippe Veber <philippe.veber@univ-lyon1.fr>

URL https://cran.r-project.org/package=morse

BugReports https://github.com/pveber/morse

Description Tools for ecotoxicologists and regulators dedicated to the mathematical and statistical modelling of toxicity test data. They use advanced and innovative methods for a valuable quantitative environmental risk assessment.

Depends R (>= 3.0.0)

SystemRequirements JAGS (>= 4.0.0) (see http://mcmc-jags.sourceforge.net)

Imports coda, dplyr, epitools, graphics, grDevices, ggplot2 (>= 2.1.0), grid, gridExtra, magrittr, methods, reshape2, rjags (>= 4.0), stats, tibble, tidyr, zoo

License GPL (>= 2)

Suggests knitr, rmarkdown, testthat

VignetteBuilder knitr

RoxygenNote 6.0.1

LazyData true

NeedsCompilation no

Repository CRAN

Date/Publication 2018-06-12 20:27:55 UTC
### R topics documented:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>morse-package</td>
<td>3</td>
</tr>
<tr>
<td>cadmium1</td>
<td>5</td>
</tr>
<tr>
<td>cadmium2</td>
<td>5</td>
</tr>
<tr>
<td>chlordan</td>
<td>6</td>
</tr>
<tr>
<td>copper</td>
<td>7</td>
</tr>
<tr>
<td>dichromate</td>
<td>8</td>
</tr>
<tr>
<td>is_exposure_constant</td>
<td>9</td>
</tr>
<tr>
<td>LCx</td>
<td>9</td>
</tr>
<tr>
<td>LCx.survFit</td>
<td>10</td>
</tr>
<tr>
<td>MFx</td>
<td>11</td>
</tr>
<tr>
<td>MFx.survFit</td>
<td>12</td>
</tr>
<tr>
<td>plot.LCx</td>
<td>14</td>
</tr>
<tr>
<td>plot.MFx</td>
<td>15</td>
</tr>
<tr>
<td>plot.reproData</td>
<td>16</td>
</tr>
<tr>
<td>plot.reproFitTT</td>
<td>17</td>
</tr>
<tr>
<td>plot.survDataCstExp</td>
<td>19</td>
</tr>
<tr>
<td>plot.survDataVarExp</td>
<td>20</td>
</tr>
<tr>
<td>plot.survFitCstExp</td>
<td>21</td>
</tr>
<tr>
<td>plot.survFitPredict</td>
<td>22</td>
</tr>
<tr>
<td>plot.survFitTKTD</td>
<td>23</td>
</tr>
<tr>
<td>plot.survFitTT</td>
<td>25</td>
</tr>
<tr>
<td>plot.survFitVarExp</td>
<td>26</td>
</tr>
<tr>
<td>plotDoseResponse</td>
<td>28</td>
</tr>
<tr>
<td>plotDoseResponse.reproData</td>
<td>29</td>
</tr>
<tr>
<td>plotDoseResponse.survDataCstExp</td>
<td>30</td>
</tr>
<tr>
<td>ppc</td>
<td>32</td>
</tr>
<tr>
<td>ppc.reproFitTT</td>
<td>32</td>
</tr>
<tr>
<td>ppc.survFitCstExp</td>
<td>33</td>
</tr>
<tr>
<td>ppc.survFitTKTD</td>
<td>34</td>
</tr>
<tr>
<td>ppc.survFitTT</td>
<td>35</td>
</tr>
<tr>
<td>ppc.survFitVarExp</td>
<td>36</td>
</tr>
<tr>
<td>predict.survFit</td>
<td>37</td>
</tr>
<tr>
<td>print.reproFitTT</td>
<td>38</td>
</tr>
<tr>
<td>print.survFitCstExp</td>
<td>39</td>
</tr>
<tr>
<td>print.survFitTKTD</td>
<td>40</td>
</tr>
<tr>
<td>print.survFitTT</td>
<td>41</td>
</tr>
<tr>
<td>print.survFitVarExp</td>
<td>41</td>
</tr>
<tr>
<td>priors_distribution</td>
<td>42</td>
</tr>
<tr>
<td>priors_distribution.survFit</td>
<td>43</td>
</tr>
<tr>
<td>priors_survData</td>
<td>43</td>
</tr>
<tr>
<td>propiconazole</td>
<td>44</td>
</tr>
<tr>
<td>propiconazole_pulse_exposure</td>
<td>45</td>
</tr>
<tr>
<td>reproData</td>
<td>45</td>
</tr>
<tr>
<td>reproDataCheck</td>
<td>46</td>
</tr>
<tr>
<td>reproFitTT</td>
<td>47</td>
</tr>
<tr>
<td>summary.reproData</td>
<td>49</td>
</tr>
</tbody>
</table>
Description

Provides tools for the analysis of survival/reproduction toxicity test data in quantitative environmental risk assessment. It can be used to explore/visualize experimental data, and to get estimates of \( LC_x \) (X\% Lethal Concentration) or \( EC_x \) (X\% Effective Concentration) by fitting exposure-response curves. The \( LC_x \), \( EC_x \) and parameters of the curve are provided along with an indication of the uncertainty of the estimation. \texttt{morse} can also be used to get an estimation of the \( NEC \) (No Effect Concentration) by fitting a Toxico-Kinetic Toxico-Dynamic (TK-TD) model.

Details

Estimation procedures in \texttt{morse} can be used without a deep knowledge of their underlying probabilistic model or inference methods. Rather, they were designed to behave as well as possible without requiring a user to provide values for some obscure parameters. That said, \texttt{morse} models can also be used as a first step to tailor new models for more specific situations.

The package currently handles survival and reproduction data. Functions dedicated to survival (resp. reproduction) analysis start with a \texttt{surv} (resp. \texttt{repro}) prefix. \texttt{morse} provides a similar workflow in both cases:

1. create and validate a dataset
2. explore a dataset
3. plot a dataset
4. fit a model on a dataset and output the expected estimates
5. check goodness of fit with posterior preditive check plot (ppc)
Those steps are presented in more details in the "Tutorial" vignette, while a more formal description of the estimation procedures are provided in the vignette called "Models in morse package". Please refer to these documents for further introduction to the use of morse.

This reference manual is a detailed description of the functions exposed in the package.

**Getting started** The package uses the *rjags* package (Plummer, 2013), an R interface to the JAGS library for Bayesian model estimation. Note that the *rjags* package does not include a copy of the JAGS library: you need to install it separately. For instructions on downloading JAGS, see the home page at [http://mcmc-jags.sourceforge.net](http://mcmc-jags.sourceforge.net). Once done, simply follow the steps described in the tutorial vignette.

<table>
<thead>
<tr>
<th>Package</th>
<th>morse</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Package</td>
</tr>
<tr>
<td>Version</td>
<td>3.0.0</td>
</tr>
<tr>
<td>Date</td>
<td>2017-09-15</td>
</tr>
<tr>
<td>License</td>
<td>GPL (&gt;=2)</td>
</tr>
</tbody>
</table>

**Author(s)**

Virgile Baudrot <virgile.baudrot@posteo.net>, Sandrine Charles <sandrine.charles@univ-lyon1.fr>, Marie Laure Delignette-Muller <marielaure.delignettemuller@vetagro-sup.fr>, Wandrille Duchemin <wandrille.duchemin@insa-lyon.fr>, Guillaume Kon-Kam-king <guillaume.kon-kam-king@univ-lyon1.fr>, Christelle Lopes <christelle.lopes@univ-lyon1.fr>, Philippe Ruiz <philippe.ruiz@univ-lyon1.fr>, Philippe Veber <philippe.veber@univ-lyon1.fr>

Maintainer: Philippe Veber <philippe.veber@univ-lyon1.fr>

**References**


**See Also**

[rjags](rjags), [ggplot2](ggplot2)
Description

Reproduction and survival datasets of chronic laboratory toxicity tests with Daphnia magna freshwater invertebrate exposed to five concentrations of cadmium during 21 days. Five concentrations were tested, with four replicates per concentration. Each replicate contained 10 organisms. Reproduction and survival were monitored at 10 time points.

Usage
data(cadmium1)

Format

A data frame with 200 observations of the following five variables:

- replicate A vector of class numeric with the replicate code (Q to RP).
- conc A vector of class numeric with the cadmium concentrations in µg.L⁻¹.
- time A vector of class integer with the time points (in days from the beginning of the experiment \( t = 0 \)).
- Nsurv A vector of class integer with the number of alive individuals at each time point for each concentration and each replicate.
- Nrepro A vector of class integer with the number of offspring at each time point for each concentration and each replicate.

References


Description

Reproduction and survival datasets of chronic laboratory toxicity tests with snails (Lymnaea stagnalis) exposed to six concentrations of cadmium during 28 days. Six concentrations were tested, with six replicates per concentration. Each replicate contained five organisms. Reproduction and survival were monitored at 17 time points.
Usage
data(cadmium2)

Format

A data frame with 612 observations of the following five variables:

- `replicate` A vector of class numeric with the replicate code (1 to 36).
- `conc` A vector of class integer with the cadmium concentrations in $\mu g.L^{-1}$.
- `time` A vector of class integer with the time points (in days from the beginning of the experiment $t=0$).
- `nsurv` A vector of class integer with the number of alive individuals at each time point for each concentration and each replicate.
- `nrepro` A vector of class integer with the number of clutches at each time point for each concentration and each replicate.

References


---

**chlordan**

Reproduction and survival datasets for *Daphnia magna* exposed to chlordan during 21 days

**Description**

Reproduction and survival datasets of chronic laboratory toxicity tests with *Daphnia magna* freshwater invertebrate exposed to six concentrations of one organochlorine insecticide (chlordan) during 21 days. Six concentrations were tested, with 10 replicates per concentration. Each replicate contained one organism. Reproduction and survival were monitored at 22 time points.

**Usage**

data(chlordan)
copper

**Format**

A data frame with 1320 observations of the following five variables:

- `replicate` A vector of class numeric with the replicate code (1 to 60).
- `conc` A vector of class numeric with the chlordane concentrations in \( \mu g.L^{-1} \).
- `time` A vector of class integer with the time points (in days from the beginning of the experiment \( t = 0 \)).
- `nsurv` A vector of class integer with the number of alive individuals at each time point for each concentration and each replicate.
- `nrepro` A vector of class integer with the number of offspring at each time point for each concentration and each replicate.

**References**


---

Reproduction and survival datasets for Daphnia magna exposed to copper during 21 days

**Description**

Reproduction and survival datasets of chronic laboratory toxicity tests with *Daphnia magna* freshwater invertebrate exposed to five concentrations of copper during 21 days. Five concentrations were tested, with three replicates per concentration. Each replicate contained 20 organisms. Reproduction and survival were monitored at 16 time points.

**Usage**

data(copper)

**Format**

A data frame with 240 observations of the following five variables:

- `replicate` A vector of class numeric with the replicate code (1 to 15).
- `conc` A vector of class numeric with the copper concentrations in \( \mu g.L^{-1} \).
- `time` A vector of class integer with the time points (in days from the beginning of the experiment \( t = 0 \)).
- `nsurv` A vector of class integer with the number of alive individuals at each time point for each concentration and each replicate.
- `nrepro` A vector of class integer with the number of offspring at each time point for each concentration and each replicate.
References


---

dichromate

Survival dataset for Daphnia magna exposed to dichromate during 21 days

Description

Survival dataset of chronic laboratory toxicity tests with *Daphnia magna* freshwater invertebrate exposed to six concentrations of one oxidizing agent (potassium dichromate) during 21 days. Six concentrations were tested with one replicate of 50 organisms per concentration. Survival is monitored at 10 time points.

Usage

data(dichromate)

Format

A data frame with 60 observations on the following four variables:

- **replicate** A vector of class numeric with the replicate code (1).
- **conc** A vector of class numeric with dichromate concentrations in mg.L$^{-1}$.
- **time** A vector of class integer with the time points (in days from the beginning of the experiment $t = 0$).
- **nsurv** A vector of class integer with the number of alive individuals at each time point for each concentration and each replicate.

References

is_exposure_constant

Description
Test in a well-formed argument to function 'survData' if the concentration is constant and different from NA for each replicate (each time-serie)

Usage
is_exposure_constant(x)

Arguments
x an object of class data.frame

Value
a boolean TRUE if concentration in replicate is constant, or FALSE if the concentration in at least one of the replicates is time-variable, and/or if NA occurs.

Examples

# (1) Load the survival dataset and test if concentration in replicates is constant
data("propiconazole")
is_exposure_constant(propiconazole)
is_exposure_constant(survData(propiconazole))

# (1) Load the survival dataset and test if concentration in replicates is constant
data("propiconazole_pulse_exposure")
is_exposure_constant(propiconazole_pulse_exposure)

LCx Predict X% Lethal Concentration at the maximum time point (default).

Description
Predict median and 95% credible interval of the x% Lethal Concentration.

Usage
LCx(object, ...)

Arguments

object An object used to select a method
...
Further arguments to be passed to generic methods

Details

When class of object is survFit, see LCx.survFit.

lcxNsurvfit
Predict x% Lethal Concentration at any specified time point for a survFit object.

Description

The function LCx, x% Lethal Concentration (LCx), is use to compute the dose required to kill x% of the members of a tested population after a specified test duration (time_tLCx) (default is the maximum time point of the experiment).

Mathematical definition of x% Lethal Concentration at time t, denoted LC(x,t), is:
\[ S(LC(x,t), t) = S(0, t) \times (1 - x/100), \]
where \( S(LC(x,t), t) \) is the survival rate at concentration LC(x,t) at time t, and \( S(0, t) \) is the survival rate at no concentration (i.e. concentration is 0) at time t which reflect the background mortality \( h_b \):
\[ S(0, t) = exp(-h_b \times t). \]

In the function LCx, we use the median of \( S(0, t) \) to rescale the x% Lethal Concentration at time t.

Usage

```r
## S3 method for class 'survFit'
LCx(object, X, time_LCx = NULL, conc_range = NULL, 
    npoints = 100, ...)  
```

Arguments

object An object of class survFit
X Percentage of individuals dying (e.g., 50 for LC50, 10 for LC10, ...)
time_LCx A number giving the time at which LCx has to be estimated. If NULL, the latest time point of the experiment is used.
conc_range A vector of length 2 with minimal and maximal value of the range of concentration. If NULL, the range is defined between 0 and the highest tested concentration of the experiment.
npoints Number of time point in conc_range between 0 and the maximal concentration. 100 by default.
... Further arguments to be passed to generic methods
Value

The function returns an object of class \texttt{LCx}, which is a list with the following information:

- \texttt{X\_prop} \quad \text{Survival rate of individuals surviving considering the median of the background mortality (i.e. } S(0, t) \ast (1 - x/100))
- \texttt{X\_prop\_provided} \quad \text{Survival rate of individuals surviving as provided in arguments (i.e. } (100 - X)/100)
- \texttt{time\_LCx} \quad \text{A number giving the time at which } LC_x \text{ has to be estimated as provided in arguments or if NULL, the latest time point of the experiment is used.}
- \texttt{df\_LCx} \quad \text{A } \texttt{data.frame} \text{ with quantiles (median, 2.5\% and 97.5\%) of } LC_X \text{ at time } time\_LCx \text{ for } X\% \text{ of individuals}
- \texttt{df\_dose} \quad \text{A } \texttt{data.frame} \text{ with four columns: concentration, and median q50 and 95\% credible interval (qinf95 and qsup95) of the survival rate at time } time\_LCx

Examples

```r
# (1) Load the data
data("propiconazole")

# (2) Create an object of class 'survData'
dataset <- survData(propiconazole)

# Not run:
# (3) Run the survFit function with model_type SD (or IT)
out_SD <- survFit(dataset, model_type = "SD")

# (4) estimate LC50 at time 4
LCx(out_SD, X = 50, time_LCx = 4)

# End(Not run)
```

\textbf{MFx} \quad \textit{Predict the Multiplication Factor leading to } x\% \text{ of reduction in survival at a specific time.}

Description

Generic method for MFx, a function denoted } MF(x, t) \text{ for } x\% \text{ Multiplication Factor at time } t.

Usage

\texttt{MFx(object, ...)}
MFx.survFit

Arguments

object An object used to select a method

... Further arguments to be passed to generic methods

Details

When class of object is survFit, see MFx.survFit.

Description

The function MFx, x% Multiplication Factor at time t, \( MF(x, t) \), is used to compute the multiplication factor applied to the concentration exposure profile in order to reduce by \( x \% \) (argument \( X \)) the survival rate at a specified test duration \( t \) (argument time_MFx) (default is the maximum time point of the experiment).

Mathematical definition of \( x \% \) Multiplication Factor at time \( t \) (at the end of a time series \( T = \{0, \ldots, t\} \)), denoted \( MF(x, t) \), is given by:
\[
S(MF(x, t) \times C_w(\tau \in T), t) = S(C_w(\tau \in T), t) \times (1 - x/100),
\]
where \( C_w(\tau \in T) \) is the initial exposure profile without multiplication factor. And so the expression \( S(MF(x, t) \times C_w(\tau \in T), t) \) is the survival rate after an exposure profile \( MF(x, t) \times C_w(\tau \in T) \) at time \( t \).

Usage

## S3 method for class 'survFit'
MFx(object, data_predict, \( X = 50 \), time_MFx = NULL,
    MFx_range = c(0, 1000), mcmc_size = 1000, hb_value = TRUE,
    spaghetti = FALSE, accuracy = 0.01, quiet = FALSE,
    threshold_iter = 100, ...)

Arguments

object An object of class survFit.

data_predict A dataframe with two columns time and conc.

\( X \) Percentage of survival change (e.g., 50 for survival decrease of 50%, or \(-50\) for survival increase of 50%). The default is 50. Only time series computed during the adaptation using a binary search in \( O(log(n)) \) are returned. However, if NULL, all time series computed from the vector MFx_range are returned.

time_MFx A number giving the time at which \( MF(x, t) \) has to be estimated. If NULL, the latest time point of the profile is used.
MFx.survFit

MFx_range
A vector from which lower and upper bound of the range of the multiplication factor MFx are generated. The default is a vector c(0, 1000). If argument X is NULL, then all the time series generated with MFx_range are returned.

mcmc_size
Can be used to reduce the number of MCMC samples in order to speed up the computation. The default is 1000.

hb_value
If TRUE, the background mortality hb is taken into account from the posterior. If FALSE, parameter hb is set to 0. The default is TRUE.

spaghetti
If TRUE, return a set of survival curves using parameters drawn from the posterior distribution.

accuracy
Accuracy of the multiplication factor. The default is 0.01.

quiet
If FALSE, print the evolution of accuracy.

threshold_iter
Threshold number of iteration.

Value
The function returns an object of class MFx, which is a list with the following information:

x_prop
Survival rate for x percent of reduction of the initial median survival rate at time time_MFx.

x_prop_provided
A number giving the proportion of reduction in survival.

time_MFx
A number giving the time at which MF(x, t) has to be estimated as provided in arguments or if NULL, the latest time point of the profile is used.

df_MFx
A data.frame with quantiles (median, 2.5% and 97.5%) of MF(x, t) at time t, time_MFx, for x% of survival reduction.

df_dose
A data.frame with quantiles (median, 2.5% and 97.5%) of survival rate along the computed multiplication factor and at time time_MFx.

MFx_tested
A vector of all multiplication factors computed.

ls_predict
A list of all object of class survfitPredict obtained from computing survival rate for every profiles build from the vector of multiplication factors MFx_tested.

Examples

# (1) Load the data
data("propiconazole")

# (2) Create an object of class 'survData'
dataset <- survData(propiconazole)

## Not run:
# (3) Run the survFit function with model_type SD (or IT)
out_SD <- survFit(dataset, model_type = "SD")

# (4) data to predict
data_4prediction <- data.frame(time = 1:10, conc = c(0,0.5,3,3,0,0,0.5,3,1.5,0))
plot.LCx

Plotting method for LCx objects

Description
This is the generic plot S3 method for the \code{LCx} class. It plots the survival rate as a function of concentration.

Usage

## S3 method for class 'LCx'
plot(x, xlab = "concentration",
     ylab = "survival rate median and 95 CI", main = NULL,
     subtitle = NULL, ...
)

Arguments

x An object of class LCx.

xlab A label for the X-axis, by default Concentration.

ylab A label for the Y-axis, by default Survival rate median and 95 CI.

main A main title for the plot.

subtitle A subtitle for the plot

... Further arguments to be passed to generic methods.

Examples

# (1) Load the data
data("propiconazole")

# (2) Create an object of class 'survData'
dataset <- survData(propiconazole)

## Not run:
# (3) Run the survFit function with model_type SD (or IT)
```r
out_SD <- survFit(dataset, model_type = "SD")

# (4) estimate LC50 at time 4
LCx_SD <- LCx(out_SD, X = 50, time_LCx = 4)

# (5) plot the object of class 'LCx'
plot(LCx_SD)
```

---

**plot.MFx**  
*Plotting method for MFx objects*

**Description**

This is the generic plot S3 method for the MFx class. It plots the survival rate as a function of the multiplication factor applied or as a function of time.

**Usage**

```r
## S3 method for class 'MFx'
plot(x, x_variable = "MFx", xlab = NULL,
     ylab = "Survival rate \n median and 95 CI", main = NULL,
     log_scale = FALSE, ncol = 3, ...)
```

**Arguments**

- **x**  
  An object of class MFx.
- **x_variable**  
  A character to define the variable for the X-axis, either "MFx" or "Time". The default is "MFx".
- **xlab**  
  A label for the X-axis, by default NULL and depend on the argument x_variable.
- **ylab**  
  A label for the Y-axis, by default Survival rate \n median and 95 CI.
- **main**  
  A main title for the plot.
- **log_scale**  
  If TRUE, the x-axis is log-scaled. Default is FALSE.
- **ncol**  
  An integer for the number of columns when several panels are plotted.
- **...**  
  Further arguments to be passed to generic methods.

**Examples**

```r
# (1) Load the data
data("propiconazole")

# (2) Create an object of class 'survData'
dataset <- survData(propiconazole)
```
## Not run:

# (3) Run the survFit function with model_type SD (or IT)
out_SD <- survFit(dataset, model_type = "SD")

# (4) data to predict
data_4prediction <- data.frame(time = 1:10, conc = c(0, 0.5, 3, 3, 0, 0.5, 3, 1.5, 0))

# (5) estimate MF for 30% reduction of survival at time 4
MFX_SD_30.4 <- MFX(out_SD, data_predict = data_4prediction, X = 30, time_MFX = 4)

# (6) plot the object of class 'MFX'
plot(MFX_SD_30.4)

# (6bis) plot with log-scale of x-axis
plot(MFX_SD_30.4, log_scale = TRUE)

# (6ter) plot with "Time" as the x-axis
plot(MFX_SD_30.4, x_variable = "time")

# (7) plot when X = NULL and along a MFX_range from 5 to 10:
MFX_SD_range <- MFX(out_SD, data_predict = data_4prediction,
                    X = NULL, time_MFX = 4, MFX_range = seq(5, 10, length.out = 50))
plot(MFX_SD_range)
plot(MFX_SD_range, x_variable = "Time", ncol = 10)

## End(Not run)

---

### plot.reproData

**Plotting method for reproData objects**

**Description**

This is the generic plot S3 method for the reproData class. It plots the cumulated number of offspring as a function of time.

**Usage**

```r
## S3 method for class 'reproData'
plot(x, xlab, ylab = "Cumulated Number of offspring",
     main = NULL, concentration = NULL, style = "ggplot",
     pool.replicate = FALSE, addlegend = FALSE, remove.someLabels = FALSE,
     ...)```

**Arguments**

- `x`: an object of class reproData
- `xlab`: label of the X-axis
- `ylab`: label of the Y-axis, by default Cumulated Number of offspring
**plot.reproFitTT**

```
main               main title for the plot
concentration      a numeric value corresponding to some concentration in data. If concentration = NULL, draws a plot for each concentration
style              graphical backend, can be 'ggplot' or 'generic'
pool.replicate     if TRUE, the datapoints of each replicate are summed for a same concentration
addlegend          if TRUE, adds a default legend to the plot
remove.someLabels   if TRUE, removes 3/4 of X-axis labels in 'ggplot' style to avoid the label overlap
...                 Further arguments to be passed to generic methods
```

**Note**

When style = "generic", the function calls the generic function `plot`
When style = "ggplot", the function return an object of class `gg` and `ggplot`, see function `ggplot`

**Examples**

```r
# (1) Load the data
data(cadmium1)

# (2) Create an object of class 'reproData'
cadmium1 <- reproData(cadmium1)

# (3) Plot the reproduction data
plot(cadmium1)

# (4) Plot the reproduction data for a fixed concentration
plot(cadmium1, concentration = 4.36, style = "generic")
```

---

**plot.reproFitTT**  
*Plotting method for reproFitTT objects*

**Description**

This is the generic plot S3 method for the reproFitTT class. It plots the concentration-effect fit under target time reproduction analysis.

**Usage**

```r
## S3 method for class 'reproFitTT'
plot(x, xlab = "Concentration", ylab = "Nb of offspring per ind/day", main = NULL, fitcol = "orange", fitlty = 1, fitlw = 1, spaghetti = FALSE, cicol = "orange", cility = 2, cilwd = 1, ribcol = "grey70", addlegend = FALSE, log.scale = FALSE, style = "ggplot", ...)
```
plot.reproFitTT

Arguments

- **x**: an object of class reproFitTT
- **xlab**: a label for the X-axis, by default Concentration
- **ylab**: a label for the Y-axis, by default Nb of offspring per ind/day
- **main**: main title for the plot
- **fitcol**: color of the fitted curve
- **fitlty**: line type of the fitted curve
- **fitlwd**: width of the fitted curve
- **spaghetti**: if TRUE, the credible interval is represented by multiple curves
- **cilcol**: color of the 95% credible limits
- **cilty**: line type of the 95% credible limits
- **cilwd**: width of the 95% credible limits
- **ribcol**: color of the ribbon between lower and upper credible limits. Transparent if NULL
- **addlegend**: if TRUE, adds a default legend to the plot
- **logNscale**: if TRUE, displays X-axis in log-scale
- **style**: graphical backend, can be 'ggplot' or 'generic'
- **...**: Further arguments to be passed to generic methods

Details

The fitted curve represents the estimated reproduction rate at the target time as a function of the chemical compound concentration. The function plots 95% credible intervals for the estimated reproduction rate (by default the grey area around the fitted curve). Typically a good fit is expected to display a large overlap between the two types of intervals. If spaghetti = TRUE, the credible intervals are represented by two dotted lines limiting the credible band, and a spaghetti plot is added to this band. It consists of the representation of simulated curves using parameter values sampled in the posterior distribution (10% of the MCMC chains are randomly taken for this sample).

Note

When style = "generic", the function calls the generic function plot
When style = "ggplot", the function return an object of class ggplot, see function ggplot

Examples

```r
# (1) Load the data
data(cadmium1)

# (2) Create an object of class "reproData"
dataset <- reproData(cadmium1)

## Not run:
# (3) Run the reproFitTT function with the log-logistic gamma-Poisson model
```
out <- reproFitTF(dataset, stoc.part = "gammapoisson",
                   ecx = c(5, 10, 15, 20, 30, 50, 80), quiet = TRUE)

# (4) Plot the fitted curve with generic style
plot(out, xlab = expression("Concentration in" ~ mg.L^(-1)),
     fitcol = "blue", cicol = "lightblue",
     main = "Log-logistic response to concentration")

## End(Not run)

---

**plot.survDataCstExp**

*Plotting method for survData objects*

---

**Description**

This is the generic plot S3 method for the survData class. It plots the number of survivors as a function of time.

**Usage**

```r
## S3 method for class 'survDataCstExp'
plot(x, xlab = "Time",
     ylab = "Number of survivors", main = NULL, concentration = NULL,
     style = "ggplot", pool.replicate = FALSE, addlegend = FALSE,
     remove.someLabels = FALSE, ...)
```

**Arguments**

- `x` an object of class survData
- `xlab` a label for the X-axis, by default Time
- `ylab` a label for the Y-axis, by default Number of survivors
- `main` main title for the plot
- `concentration` a numeric value corresponding to some concentration(s) in data. If `concentration = NULL`, draws a plot for each concentration
- `style` graphical backend, can be 'generic' or 'ggplot'
- `pool.replicate` if TRUE, the datapoints of each replicate are summed for a same concentration
- `addlegend` if TRUE, adds a default legend to the plot
- `remove.someLabels` if TRUE, removes 3/4 of X-axis labels in 'ggplot' style to avoid label overlap
- `...` Further arguments to be passed to generic methods

**Note**

When `style = "ggplot"` (default), the function calls function `ggplot` and returns an object of class ggplot.
Examples

# (1) Load the data
data(zinc)
zinc <- survData(zinc)

# (2) Plot survival data with a ggplot style
plot(zinc)

# (3) Plot the survival data for one specific concentration
plot(zinc, concentration = 0.66)

plot.survDataVarExp  # Plotting method for survDataVarExp objects

Description

This is the generic plot S3 method for the survDataVarC class. It plots the number of survivors as a function of time.

Usage

## S3 method for class 'survDataVarExp'
plot(x, xlab = "Time",
     ylab = "Number of survivors", main = NULL, one.plot = FALSE,
     faceting_level = NULL, ...)

Arguments

x          an object of class survDataVarExp
xlab       a label for the X-axis, by default Time
ylab       a label for the Y-axis, by default Number of survivors
main       main title for the plot
one.plot   if TRUE, draws all the points in one plot instead of one per replicate
faceting_level
            a vector of characters to rank replicates in the multi plot (i.e. one.plot == FALSE)
...        Further arguments to be passed to generic methods

Value

an object of class ggplot, see function ggplot
plot.survFitCstExp  

Plotting method for survFit objects

Description

This is the generic plot S3 method for the survFit. It plots the fit obtained for each concentration of chemical compound in the original dataset.

Usage

```r
# S3 method for class 'survFitCstExp'
plot(x, xlab = "Time", ylab = "Survival rate",
     main = NULL, concentration = NULL, spaghetti = FALSE,
     one.plot = FALSE, adddata = FALSE, addlegend = FALSE,
     style = "ggplot", ...)
```

Arguments

- **x**: An object of class survFit.
- **xlab**: A label for the X-axis, by default Time.
- **ylab**: A label for the Y-axis, by default Survival rate.
- **main**: A main title for the plot.
- **concentration**: A numeric value corresponding to some specific concentrations in data. If concentration = NULL, draws a plot for each concentration.
- **spaghetti**: if TRUE, draws a set of survival curves using parameters drawn from the posterior distribution
- **one.plot**: if TRUE, draws all the estimated curves in one plot instead of one plot per concentration.
- **adddata**: if TRUE, adds the observed data to the plot with (frequentist binomial) confidence intervals
- **addlegend**: if TRUE, adds a default legend to the plot.
- **style**: graphical backend, can be 'generic' or 'ggplot'
- **...**: Further arguments to be passed to generic methods.

Details

The fitted curves represent the estimated survival rate as a function of time for each concentration. The black dots depict the observed survival rate at each time point. Note that since our model does not take inter-replicate variability into consideration, replicates are systematically pooled in this plot. The function plots both 95% credible intervals for the estimated survival rate (by default the grey area around the fitted curve) and 95% binomial confidence intervals for the observed survival rate (as black error bars if adddata = TRUE). Both types of intervals are taken at the same level. Typically a good fit is expected to display a large overlap between the two types of intervals. If spaghetti = TRUE, the credible intervals are represented by two dotted lines limiting the credible
band, and a spaghetti plot is added to this band. This spaghetti plot consists of the representation of simulated curves using parameter values sampled in the posterior distribution (2% of the MCMC chains are randomly taken for this sample).

---

**plot.survFitPredict**  
*Plotting method for survFitPredict objects*

**Description**

This is the generic plot S3 method for the survFitPredict. It plots the fit obtained for each concentration of chemical compound in the provided dataset.

**Usage**

```r
## S3 method for class 'survFitPredict'
plot(x, xlab = "Time", ylab = "Survival rate", 
     main = NULL, spaghetti = FALSE, one.plot = FALSE, adddata = FALSE, 
     mcmc_size = NULL, ...) 
```

**Arguments**

- `x`  
  An object of class survFitPredict.

- `xlab`  
  A label for the X-axis, by default `Time`.

- `ylab`  
  A label for the Y-axis, by default `Survival rate`.

- `main`  
  A main title for the plot.

- `spaghetti`  
  If TRUE, draws a set of survival curves using parameters drawn from the posterior distribution

- `one.plot`  
  if TRUE, draws all the estimated curves in one plot instead of one plot per concentration.

- `adddata`  
  if TRUE, adds the observed data to the plot

- `mcmc_size`  
  A numerical value referring by default to the size of the mcmc in object survFitPredict. This option is specific to survFitPredict objects for which computing time may be long. mcmc_size can be used to reduce the number of mcmc samples in order to speed up the computation.

- `...`  
  Further arguments to be passed to generic methods.

**Details**

The fitted curves represent the **estimated survival rate** as a function of time for each concentration. The black dots depict the **observed survival rate** at each time point. The function plots both 95% binomial credible intervals for the estimated survival rate. If spaghetti = TRUE, the credible intervals are represented by two dotted lines limiting the credible band, and a spaghetti plot is added to this band. This spaghetti plot consists of the representation of simulated curves using parameter values sampled in the posterior distribution (10% of the MCMC chains are randomly taken for this sample).
Examples

```r
# (1) Load the survival data
data("propiconazole_pulse_exposure")

# (2) Create an object of class "survData"
dataset <- survData(propiconazole_pulse_exposure)

## Not run:
# (3) Run the survFit function
out <- survFit(dataset, model_type = "SD")

# (4) Create a new data table for prediction
data_4prediction <- data.frame(time = 1:10, conc = c(0,5,5,5,0,5,5,5,5), replicate = rep("predict", 10))

# (5) Predict on a new dataset
predict_out <- predict(out, data_predict = data_4prediction, spaghetti = TRUE)

# (6) Plot the predicted curve
plot(predict_out)
plot(predict_out, spaghetti = TRUE)

## End(Not run)
```

---

**plot.survFitTKTD**

*Plotting method for survFitTKTD objects*

**Description**

This is the generic plot S3 method for the survFitTKTD. It plots the fit obtained for each concentration of chemical compound in the original dataset.

**Usage**

```r
## S3 method for class 'survFitTKTD'
plot(x, xlab = "Time", ylab = "Survival rate", main = NULL, concentration = NULL, spaghetti = FALSE, one.plot = FALSE, adddata = FALSE, addlegend = FALSE, style = "ggplot", ...)
```

**Arguments**

- `x` An object of class survFitTKTD.
- `xlab` A label for the X-axis, by default Time.
- `ylab` A label for the Y-axis, by default Survival rate.
- `main` A main title for the plot.
concentration A numeric value corresponding to some specific concentration in data. If concentration = NULL, draws a plot for each concentration.

spaghetti if TRUE, draws a set of survival curves using parameters drawn from the posterior distribution

one.plot if TRUE, draws all the estimated curves in one plot instead of one plot per concentration.

adddata if TRUE, adds the observed data to the plot with (frequentist binomial) confidence intervals

addlegend if TRUE, adds a default legend to the plot.

style graphical backend, can be 'generic' or 'ggplot'

Details
The fitted curves represent the estimated survival rate as a function of time for each concentration. When adddata = TRUE the black dots depict the observed survival rate at each time point. Note that since our model does not take inter-replicate variability into consideration, replicates are systematically pooled in this plot. The function plots both 95% credible intervals for the estimated survival rate (by default the grey area around the fitted curve) and 95% binomial confidence intervals for the observed survival rate (as black error bars if adddata = TRUE). Both types of intervals are taken at the same level. Typically a good fit is expected to display a large overlap between the two types of intervals. If spaghetti = TRUE, the credible intervals are represented by two dotted lines limiting the credible band, and a spaghetti plot is added to this band. This spaghetti plot consists of the representation of simulated curves using parameter values sampled in the posterior distribution (2% of the MCMC chains are randomly taken for this sample).

Examples

# (1) Load the survival data
data(propiconazole)

# (2) Create an object of class "survData"
dataset <- survData(propiconazole)

## Not run:
# (3) Run the survFitTKTD function ('SD' model only)
out <- survFitTKTD(dataset)

# (4) Plot the fitted curves in one plot
plot(out)

# (5) Plot one fitted curve per concentration with credible limits as
# spaghetti, data and confidence intervals
# and with a ggplot style
plot(out, spaghetti = TRUE, adddata = TRUE, one.plot = FALSE,
     style = "ggplot")

# (6) Plot fitted curve for one specific concentration
plot(out, concentration = 36, style = "ggplot")

## End(Not run)

---

**plot.survFitTT**

**Plotting method for survFitTT objects**

**Description**

This is the generic plot S3 method for the survFitTT class. It plots concentration-response fit under target time survival analysis.

**Usage**

```r
## S3 method for class 'survFitTT'
plot(x, xlab = "Concentration", ylab = "Survival rate",
     main = NULL, fitcol = "orange", fitlty = 1, fitlwd = 1,
     spaghetti = FALSE, cicol = "orange", cilty = 2, cilwd = 1,
     ribcol = "grey70", adddata = FALSE, addlegend = FALSE,
     log.scale = FALSE, style = "ggplot", ...)
```

**Arguments**

- `x` an object of class `survFitTT`
- `xlab` a label for the X-axis, default is `Concentration`
- `ylab` a label for the Y-axis, default is `Survival rate`
- `main` main title for the plot
- `fitcol` color of the fitted curve
- `fitlty` line type of the fitted curve
- `fitlwd` width of the fitted curve
- `spaghetti` if `TRUE`, the credible interval is represented by multiple curves
- `cicol` color of the 95% credible interval limits
- `cilty` line type for the 95% credible interval limits
- `cilwd` width of the 95% credible interval limits
- `ribcol` color of the ribbon between lower and upper credible limits. Transparent if `NULL`
- `adddata` if `TRUE`, adds the observed data with confidence intervals to the plot
- `addlegend` if `TRUE`, adds a default legend to the plot
- `log.scale` if `TRUE`, displays X-axis in log-scale
- `style` graphical backend, can be 'generic' or 'ggplot'
- `...` Further arguments to be passed to generic methods
Details

The fitted curve represents the **estimated survival rate** at the target time as a function of the concentration of chemical compound; When `adddata = TRUE` the black dots depict the **observed survival rate** at each tested concentration. Note that since our model does not take inter-replicate variability into consideration, replicates are systematically pooled in this plot. The function plots both 95% credible intervals for the estimated survival rate (by default the grey area around the fitted curve) and 95% binomial confidence intervals for the observed survival rate (as black segments if `adddata = TRUE`). Both types of intervals are taken at the same level. Typically a good fit is expected to display a large overlap between the two intervals. If `spaghetti = TRUE`, the credible intervals are represented by two dotted lines limiting the credible band, and a spaghetti plot is added to this band. This spaghetti plot consists of the representation of simulated curves using parameter values sampled in the posterior distribution (10% of the MCMC chains are randomly taken for this sample).

Note

When `style = "ggplot"`, the function calls function `ggplot` and returns an object of class `ggplot`.

Examples

```r
# (1) Load the data
data(cadmium1)

# (2) Create an object of class "survData"
dat <- survData(cadmium1)

## Not run:
# (3) Run the survFitTT function with the log-logistic
#     binomial model
out <- survFitTT(dat, lcx = c(5, 10, 15, 20, 30, 50, 80),
                  quiet = TRUE)

# (4) Plot the fitted curve
plot(out, log.scale = TRUE, adddata = TRUE)

# (5) Plot the fitted curve with ggplot style
plot(out, xlab = expression("Concentration in" ~ mu^"g"^(-1)),
     fitcol = "blue", adddata = TRUE, cicol = "blue",
     style = "ggplot")

## End(Not run)
```
Description

This is the generic plot S3 method for the survfit. It plots the fit obtained for each concentration of chemical compound in the original dataset.

Usage

```r
## S3 method for class 'survFitVarExp'
plot(x, xlab = "Time", ylab = "Survival rate",
     main = NULL, spaghetti = FALSE, one.plot = FALSE, adddata = TRUE,
     mcmc_size = NULL, ...)
```

Arguments

- **x**: An object of class survFit.
- **xlab**: A label for the X-axis, by default Time.
- **ylab**: A label for the Y-axis, by default Survival rate.
- **main**: A main title for the plot.
- **spaghetti**: if TRUE, draws a set of survival curves using parameters drawn from the posterior distribution
- **one.plot**: if TRUE, draws all the estimated curves in one plot instead of one plot per concentration.
- **adddata**: if TRUE, adds the observed data to the plot with (frequentist binomial) confidence intervals.
- **mcmc_size**: A numerical value referring by default to the size of the mcmc in object survFit. This option is specific to survFitVarExp objects for which computing time may be long. mcmc_size can be used to reduce the number of mcmc samples in order to speed up the computation.
- **...**: Further arguments to be passed to generic methods.

Details

The fitted curves represent the estimated survival rate as a function of time for each concentration. The black dots depict the observed survival rate at each time point. Note that since our model does not take inter-replicate variability into consideration, replicates are systematically pooled in this plot. The function plots both 95% binomial credible intervals for the estimated survival rate (by default the grey area around the fitted curve) and 95% binomial confidence intervals for the observed survival rate (as black segments if adddata = TRUE). Both types of intervals are taken at the same level. Typically a good fit is expected to display a large overlap between the two types of intervals. If spaghetti = TRUE, the credible intervals are represented by two dotted lines limiting the credible band, and a spaghetti plot is added to this band. This spaghetti plot consists of the representation of simulated curves using parameter values sampled in the posterior distribution (10% of the MCMC chains are randomly taken for this sample).
Examples

# (1) Load the survival data
data("propiconazole_pulse_exposure")

# (2) Create an object of class "survData"
dataset <- survData(propiconazole_pulse_exposure)

## Not run:
# (3) Run the survFit function
out <- survFit(dataset, model_type = "SD")

# (4) Summary look the estimated values (parameters)
summary(out)

# (5) Plot the fitted curve
plot(out, adddata = FALSE)

# (6) Plot the fitted curve with ggplot style and CI as spaghetti
plot(out, spaghetti = TRUE)

## End(Not run)

---

plotDoseResponse  

**Plot dose-response from raw data**

Description

Plots the response of the effect as a function of the concentration at a given target time.

Usage

plotDoseResponse(x, ...)

Arguments

- `x` an object used to select a method plotDoseResponse
- `...` Further arguments to be passed to generic methods
**plotDoseResponse.reproData**

*Plot dose-response from reproData objects*

**Description**

This is the generic `plotDoseResponse` S3 method for the `reproData` class. It plots the number of offspring per individual-days as a function of concentration at a given target time.

**Usage**

```r
## S3 method for class 'reproData'
plotDoseResponse(x, xlab = "Concentration", ylab = "Nb of offspring per ind.day", main = NULL, ylim = NULL, target.time = NULL, style = "ggplot", log.scale = FALSE, remove.someLabels = FALSE, axis = TRUE, addlegend = TRUE, ...)
```

**Arguments**

- `x`: an object of class `reproData`
- `xlab`: a label for the X-axis, by default `Concentration`
- `ylab`: a label for the Y-axis, by default `Nb of offspring per ind.day`
- `main`: main title for the plot
- `ylim`: Y-axis limits
- `target.time`: a numeric value corresponding to some observed time points in data
- `style`: graphical backend, can be `"ggplot"` or `"generic"`
- `log.scale`: if TRUE, displays X-axis in log-scale
- `remove.someLabels`: if TRUE, removes 75% of X-axis labels in `"ggplot"` style to avoid the label overlap
- `axis`: if TRUE displays ticks and label axis
- `addlegend`: if TRUE, adds a default legend to the plot
- `...`: Further arguments to be passed to generic methods

**Details**

The function plots the observed values of the reproduction rate (number of reproduction outputs per individual-day) at a given time point as a function of concentration. The 95% Poisson confidence interval is added to each reproduction rate. It is calculated using function `pois.exact` from package `epitools`. As replicates are not pooled in this plot, overlapped points are shifted on the x-axis to help the visualization of replicates.
Note

When style = "generic", the function calls the generic function plot
When style = "ggplot", the function return an object of class ggplot, see function ggplot

See Also

pois.exact

Examples

# (1) Load the data
data(zinc)

# (2) Create an object of class 'reproData'
zinc_rpr <- reproData(zinc)

# (3) Plot dose-response
plotDoseResponse(zinc_rpr)

# (4) Plot dose-response with a generic style
plotDoseResponse(zinc_rpr, style = "generic")
target.time: a numeric value corresponding to some observed time in data

style: graphical backend, can be 'ggplot' or 'generic'

log.scale: if TRUE, displays X-axis in log-scale

remove.someLabels: if TRUE, removes 75% of X-axis labels in 'ggplot' style to avoid the label overlap

addLegend: if TRUE, adds a default legend to the plot

... Further arguments to be passed to generic methods

Details

The function plots the observed values of the survival rate at a given time point as a function of concentration. The 95% binomial confidence interval is added to each survival rate. It is calculated using function `binom.test` from package `stats`. Replicates are systematically pooled in this plot.

Note

When style = "generic", the function calls the generic function `plot`

When style = "ggplot", the function return an object of class `ggplot`, see function `ggplot`

See Also

`binom.test`

Examples

```r
library(ggplot2)

# (1) Load the data
data(zinc)

# (2) Create an object of class 'survData'
zinc <- survData(zinc)

# (3) Plot dose-response
plotDoseResponse(zinc)

# (4) Plot dose-response with a generic style
plotDoseResponse(zinc, style = "generic")
```
**Description**

Plots posterior predictive check for `reproFitTT`, `survFitTT`, `survFitTKTD`, `survFitCstExp` and `survFitVarExp` objects.

**Usage**

```r
ppc(x, ...)  # Style: ggplot
```

**Arguments**

- `x`: an object used to select a method `ppc`
- `...`: Further arguments to be passed to generic methods

**Details**

Depending on the class of the object `x` see their links. For class `reproFitTT`: `ppc.reproFitTT`; for class `survFitTT`: `ppc.survFitTT`; for class `survFitTKTD`: `ppc.survFitTKTD`; for class `survFitCstExp`: `ppc.survFitCstExp` and for class `survFitVarExp`: `ppc.survFitVarExp`.

**Description**

This is the generic ppc S3 method for the `reproFitTT` class. It plots the predicted values with 95% credible intervals versus the observed values.

**Usage**

```r
## S3 method for class 'reproFitTT'
ppc(x, style = "ggplot",
xlab = "Observed Cumul. Nbr. of offspring",
ylab = "Predicted Cumul. Nbr. of offspring", main = NULL, ...)
```

**Arguments**

- `x`: An object of class `reproFitTT`
- `style`: Graphical backend, can be 'generic' or 'ggplot'
- `xlab`: A label for the X-axis, by default 'Observed Cumul. Nbr. of offspring'
- `ylab`: A label for the Y-axis, by default 'Predicted Cumul. Nbr. of offspring'
- `main`: Main title for the plot
- `...`: Further arguments to be passed to generic methods
Details

The coordinates of black points are the observed values of the cumulated number of reproduction outputs for a given concentration ($X$-scale) and the corresponding predicted values ($Y$-scale). 95% prediction intervals are added to each predicted value, colored in green if this interval contains the observed value and in red in the other case. As replicates are not pooled in this plot, overlapped points are shifted on the $X$-axis to help the visualization of replicates. The bisecting line ($y = x$) is added to the plot in order to see if each prediction interval contains each observed value. As replicates are shifted on the $X$-axis, this line may be represented by steps.

Examples

```r
# (1) Load the data
data(cadmium1)

# (2) Create an object of class "reproData"
dataset <- reproData(cadmium1)

## Not run:
# (3) Run the reproFitTT function with the log-logistic gamma-Poisson model
out <- reproFitTT(dataset, stoc.part = "gammapoisson",
                  ecx = c(5, 10, 15, 20, 30, 50, 80), quiet = TRUE)

# (4) Plot observed versus predicted values
ppc(out)

## End(Not run)
```

### ppc.survFitCstExp

Posterior predictive check plot for `survFitCstExp` objects

Description

This is the generic ppc S3 method for the `survFitCstExp` class. It plots the predicted values along with 95% credible intervals versus the observed values for `survFit` objects.

Usage

```r
## S3 method for class 'survFitCstExp'
ppc(x, style = "ggplot", main = NULL, ...)
```

Arguments

- `x`: An object of class `survFitCstExp`
- `style`: graphical backend, can be 'generic' or 'ggplot'
- `main`: main title for the plot
- `...`: Further arguments to be passed to generic methods
Details

The black points show the observed number of survivors (pooled replicates, on X-axis) against the corresponding predicted number (Y-axis). Predictions come along with 95% prediction intervals, which are depicted in green when they contain the observed value and in red otherwise. Samples with equal observed value are shifted on the X-axis. For that reason, the bisecting line (y = x), is represented by steps when observed values are low. That way we ensure green intervals do intersect the bisecting line.

Examples

```r
# (1) Load the data
data(propiconazole)

# (2) Create an object of class "survData"
dataset <- survData(propiconazole)

## Not run:
# (3) Run the survFitTKTD function with the TKTD model ('SD' or 'IT')
out <- survFit(dataset, model_type = "SD")

# (4) Plot observed versus predicted values
ppc(out)

## End(Not run)
```

Description

This is the generic ppc S3 method for the survFitTKTD class. It plots the predicted values along with 95% credible intervals versus the observed values for survFitTKTD objects.

Usage

```r
## S3 method for class 'survFitTKTD'
ppc(x, style = "ggplot", main = NULL, ...)
```

Arguments

- `x`: An object of class survFitTKTD
- `style`: Graphical backend, can be 'generic' or 'ggplot'
- `main`: Main title for the plot
- `...`: Further arguments to be passed to generic methods
Details

The black points show the observed number of survivors (pooled replicates, on $X$-axis) against the corresponding predicted number ($Y$-axis). Predictions come along with 95% prediction intervals, which are depicted in green when they contain the observed value and in red otherwise. Samples with equal observed value are shifted on the $X$-axis. For that reason, the bisecting line ($y = x$), is represented by steps when observed values are low. That way we ensure green intervals do intersect the bisecting line.

Examples

```r
# (1) Load the data
data(propiconazole)

# (2) Create an object of class "survData"
dat <- survData(propiconazole)

## Not run:
# (3) Run the survFitTKTD function with the TKTD model ('SD' only)
out <- survFitTKTD(dat)

# (4) Plot observed versus predicted values
ppc(out)

## End(Not run)
```

---

**Description**

This is the generic ppc S3 method for the survFitTT class. It plots the predicted values with 95% credible intervals versus the observed values for survFitTT objects.

**Usage**

```r
## S3 method for class 'survFitTT'
ppc(x, style = "ggplot", main = NULL, ...)
```

**Arguments**

- **x**: An object of class survFitTT
- **style**: graphical backend, can be 'generic' or 'ggplot'
- **main**: main title for the plot
- **...**: Further arguments to be passed to generic methods
Details

The coordinates of black points are the observed values of the number of survivors (pooled replicates) for a given concentration (X-axis) and the corresponding predicted values (Y-axis). 95% prediction intervals are added to each predicted value, colored in green if this interval contains the observed value and in red otherwise. The bisecting line (y = x) is added to the plot in order to see if each prediction interval contains each observed value. As replicates are shifted on the x-axis, this line is represented by steps.

Examples

```r
# (1) Load the data
data(cadmium1)

# (2) Create an object of class "survData"
dat <- survData(cadmium1)

## Not run:
# (3) Run the survFitTT function with the log-logistic binomial model
out <- survFitTT(dat, lcx = c(5, 10, 15, 20, 30, 50, 80), quiet = TRUE)

# (4) Plot observed versus predicted values
ppc(out)

## End(Not run)
```

---

**Description**

This is the generic ppc S3 method for the survFitVarExp class. It plots the predicted values along with 95% credible intervals versus the observed values for survFit objects.

**Usage**

```r
## S3 method for class 'survFitVarExp'
ppc(x, xlab = "Observed nb of survivors", ylab = "Predicted nb of survivors", main = NULL, ...)
```

**Arguments**

- `x` An object of class survFitVarExp
- `xlab` A label for the X-axis, by default Observed nb of survivors.
- `ylab` A label for the Y-axis, by default Predicted nb of survivors.
- `main` A main title for the plot.
- `...` Further arguments to be passed to generic methods
predict.survFit

Details
The black points show the observed number of survivors (on X-axis) against the corresponding predicted number (Y-axis). Predictions come along with 95% prediction intervals, which are depicted in green when they contain the observed value and in red otherwise.

Examples

```r
# (1) Load the data
data(propiconazole_pulse_exposure)

# (2) Create an object of class "survData"
dat <- survData(propiconazole_pulse_exposure)

## Not run:
# (3) Run the survFitTKTD function with the TKTD model ('SD' or 'IT')
out <- survfit(dat, model_type = "SD")

# (4) Plot observed versus predicted values
ppc(out)

## End(Not run)
```

predict.survFit  Predict method for survFit objects

Description
This is the generic predict S3 method for the survFit class. It provides simulation for "SD" or "IT" models under constant or time-variable exposure.

Usage

```r
## S3 method for class 'survFit'
predict(object, data_predict = NULL, spaghetti = FALSE,
         mcmc_size = NULL, hb_value = TRUE, ratio_no.NA = 0.95, ...)
```

Arguments

- **object**: An object of class survFit
- **data_predict**: A dataframe with three columns time, conc and replicate used for prediction. If NULL, prediction is based on x object of class survFit used for fitting.
- **spaghetti**: If TRUE, return a set of survival curves using parameters drawn from the posterior distribution.
- **mcmc_size**: Can be used to reduce the number of mcmc samples in order to speed up the computation.
hb_value If TRUE, the background mortality hb is taken into account from the posterior. If FALSE, parameter hb is set to 0. The default is TRUE.

ratio_no.NA A numeric between 0 and 1 standing for the proportion of non-NA values required to compute quantile. The default is 0.95.

Further arguments to be passed to generic methods

Examples

# (1) Load the survival data
data("propiconazole_pulse_exposure")

# (2) Create an object of class "survData"
dataset <- survData(propiconazole_pulse_exposure)

## Not run:
# (3) Run the survFit function
out <- survFit(dataset, model_type = "SD")

# (4) Create a new data table for prediction
data_4prediction <- data.frame(time = 1:10,
                            conc = c(0,5,30,30,0,0,5,30,15,0),
                            replicate= rep("predict", 10))

# (5) Predict on a new dataset
predict_out <- predict(object = out, data_predict = data_4prediction, spaghetti = TRUE)

## End(Not run)

print.reproFitTT

Print of reproFitTT object

Description

This is the generic print S3 method for the reproFitTT class. It prints the underlying JAGS model and some information on the Bayesian inference procedure.

Usage

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

Arguments

x An object of class reproFitTT

Further arguments to be passed to generic methods
print.survFitCstExp

Examples

# (1) Load the data
data(cadmium1)

# (2) Create an object of class 'reproData'
cadmium1 <- reproData(cadmium1)

## Not run:
# (3) Run the reproFitTT function with the log-logistic
# model
out <- reproFitTT(cadmium1, ecx = c(5, 10, 15, 20, 30, 50, 80), quiet = TRUE)

# (4) Print the reproFitTT object
print(out)

## End(Not run)

print.survFitCstExp  Print of survFit object

Description

This is the generic print S3 method for the survFitCstExp class. It prints the underlying JAGS
model and some information on the Bayesian inference procedure.

Usage

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

Arguments

x An object of class survFitCstExp

... Further arguments to be passed to generic methods.

Examples

# (1) Load the data
data(propiconazole)

# (2) Create an object of class 'survData'
dat <- survData(propiconazole)

## Not run:
# (3) Run the survFit function with TK-TD model 'SD' or 'IT'
out <- survFit(dat, quiet = TRUE, model_type="SD")
print.survFitTKTD

Print of survFitTKTD object

Description

This is the generic print S3 method for the survFitTKTD class. It prints the underlying JAGS model and some information on the Bayesian inference procedure.

Usage

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

Arguments

x An object of class survFitTKTD
...

Further arguments to be passed to generic methods.

Examples

# (1) Load the data
data(propiconazole)

# (2) Create an object of class 'survData'
dat <- survData(propiconazole)

## Not run:
# (3) Run the survFitTKTD function
out <- survFitTKTD(dat, quiet = TRUE)

# (4) Print the survFitTKTD object
print(out)

## End(Not run)
Description

This is the generic `print` S3 method for the `survFitTT` class. It prints the underlying JAGS model and some information on the Bayesian inference procedure.

Usage

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

Arguments

- `x` An object of class `survFitTT`
- `...` Further arguments to be passed to generic methods

Examples

```r
# (1) Load the data
data(cadmium1)

# (2) Create an object of class 'survData'
cadmium1 <- survData(cadmium1)

## Not run:
# (3) Run the survFitTT function with the log-logistic # binomial model
out <- survFitTT(cadmium1, lcx = c(5, 10, 15, 20, 30, 50, 80), quiet = TRUE)

# (4) Print the survFitTT object
print(out)

## End(Not run)
```

Description

This is the generic `print` S3 method for the `survFitVarExp` class. It prints the underlying JAGS model and some information on the Bayesian inference procedure.
Usage

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

Arguments

- `x` : An object of class `survFitVarExp`
- `...` : Further arguments to be passed to generic methods.

Examples

```r
# (1) Load the data
data(propiconazole_pulse_exposure)

# (2) Create a survData object
dataset <- survData(propiconazole_pulse_exposure)

## Not run:
# (3) Run the survFit function with TK-TD model 'SD' or 'IT'
out <- survFit(dataset, model_type="SD")

# (4) Print the survFit object
print(out)

## End(Not run)
```

---

**priors_distribution**

*Density distribution of priors.*

Description

Return a data.frame with priors density distribution of parameters used in object.

Usage

```r
priors_distribution(object, ...)
```

Arguments

- `object` : An object used to select a method
- `...` : Further arguments to be passed to generic methods

Details

When the object is of class `survFit`, see `priors_distribution.survFit`
priors_distribution.survFit

Density distribution of priors from a survFit object.

Description

Return a data.frame with priors distribution of parameters used in object.

Usage

## S3 method for class 'survFit'
priors_distribution(object, size_sample = 1000, ...)

Arguments

- **object**: An object of class survFit.
- **size_sample**: Size of the random generation of the distribution. Default is 1e3.
- **...**: Further arguments to be passed to generic methods.

priors_survData

Create a list of scalars giving priors to use in Bayesian modelling

Description

Create a list of scalars giving priors to use in Bayesian modelling

Usage

priors_survData(x, model_type = NULL)

Arguments

- **x**: An object of class survData
- **model_type**: TKTD model type ('SD' or 'IT')

Value

A list for parameterization of priors for Bayesian modeling with JAGS
Examples

# (1) Load the data
data(cadmium1)

# (2) Create a survData object
dat <- survData(cadmium1)

# (3) Create priors for SD model_type
priors_survData(dat, model_type = "SD")

# (4) Create priors for IT model_type
priors_survData(dat, model_type = "IT")

---

propiconazole Survival dataset for Gammarus pulex exposed to propiconazole during four days

Description

Survival dataset of chronic laboratory toxicity tests with *Gammarus pulex* freshwater invertebrate exposed to eight concentrations of one fungicide (propiconazole) during four days. Eight concentrations were tested with two replicates of 10 organisms per concentration. Survival is monitored at five time points.

Usage

data(propiconazole)

Format

A dataframe with 75 observations on the following four variables:

- **replicate**: A vector of class `factor` with the replicate code (SC for the control and A1 to G2 for other profiles).
- **conc**: A vector of class `numeric` with propiconazole concentrations in $\mu\text{mol.L}^{-1}$.
- **time**: A vector of class `integer` with the time points (in days from the beginning of the experiment $t = 0$).
- **nsurv**: A vector of class `integer` with the number of alive individuals at each time point for each concentration and each replicate.

References

propiconazole_pulse_exposure

Survival dataset for Gammarus pulex exposed to propiconazole during 10 days with time-variable exposure concentration (non-standard pulsed toxicity experiments)

Description

Survival dataset of laboratory toxicity tests with Gammarus pulex freshwater invertebrates exposed to several profiles of concentrations (time-variable concentration for each time series) of one fungicide (propiconazole) during 10 days.

Usage

data(propiconazole_pulse_exposure)

Format

A data frame with 74 observations on the following four variables:

- replicate: A vector of class factor with the replicate code (varControl, varA, varB and varC).
- conc: A vector of class numeric with propiconazole concentrations in $\mu$mol.L$^{-1}$.
- time: A vector of class integer with the time points (in days from the beginning of the experiment $t = 0$).
- Nsurv: A vector of class integer with the number of alive individuals at each time point for each concentration and each replicate.

References


reproData

Creates a dataset for reproduction toxicity analysis

Description

This function creates a reproData object from experimental data provided as a data.frame. The resulting object can then be used for plotting and model fitting. The reproData class is a sub-class of survData, meaning that all functions and method available for survival analysis can be used with reproData objects.

Usage

reproData(x)
Arguments

x a dataframe as expected by \texttt{survdata} containing one additional \texttt{Nrepro} column
of class \texttt{integer} with positive values only. This column should provide the
number of offspring produced since the last observation.

Details

The \texttt{x} argument contains the experimental data, and should have the same structure than the argu-
ment of \texttt{survdata}, plus a single additional column providing the total number of offspring observed
since the last time point. The function fails if \texttt{x} does not meet the expected requirements. Please
run \texttt{reprodDataCheck} to ensure \texttt{x} is well-formed.

Note that experimental data with time-variable exposure are not supported.

Value

An object of class \texttt{reproData}.

Examples

```r
# (1) Load reproduction dataset
data(cadmium1)

# (2) Create an object of class "reproData"
dat <- reproData(cadmium1)
class(dat)
```

reprodDataCheck \textit{Checks if an object can be used to perform reproduction toxicity data
analysis}

Description

The \texttt{reprodDataCheck} function can be used to check if an object containing data from a reproduction
toxicity assay meets the expectations of the function \texttt{reproData}.

Usage

\texttt{reprodDataCheck(data, diagnosis.plot = TRUE)}

Arguments

data any object
diagnosis.plot if \texttt{TRUE}, produces a diagnosis plot
Details

Since in morse’ reproduction datasets are a special case of survival datasets, reproDataCheck performs the same verifications than survDataCheck plus additional ones that are specific to reproduction data.

Value

The function returns a data.frame similar to the one returned by survDataCheck, except that it may contain the following additional error ids:

- nreproInteger: column Nrepro contains values of class other than integer
- Nrepro0T0: Nrepro is not 0 at time 0 for each concentration and each replicate
- Nsurv0Nreprotp1P: at a given time \( T \), the number of alive individuals is null and the number of collected offspring is not null for the same replicate and the same concentration at time \( T+1 \)

Note

If an error of type dataframeExpected or missingColumn is detected, the function reproDataCheck is stopped. When no error is detected the reproDataCheck function returns an empty dataframe.

See Also

reproData

Examples

```r
# Run the check data function
data(copper)
reproDataCheck(copper)

# Now we insert an error in the dataset, by setting a non-zero number of
# offspring at some time, although there is no surviving individual in the
# replicate from the previous time point.
copper[148, "Nrepro"] <- as.integer(1)
reproDataCheck(copper)
```

reproFitTT 

Fits a Bayesian concentration-effect model for target-time reproduction analysis

Description

This function estimates the parameters of a concentration-effect model for target-time reproduction analysis using Bayesian inference. In this model the endpoint is the cumulated number of reproduction outputs over time, with potential mortality all along the experiment.
reproFitTT

**Usage**

```r
ttrep(data, stoc.part = "bestfit", target.time = NULL, ecx = c(5, 10, 20, 50), n.chains = 3, quiet = FALSE)
```

**Arguments**

- `data`: an object of class `reproData`
- `stoc.part`: stochastic part of the model. Possible values are "bestfit", "poisson" and "gammapoisson"
- `target.time`: defines the target time point at which to analyse the repro data. By default the last time point
- `ecx`: desired values of $x$ (in percent) for which to compute $EC_x$
- `n.chains`: number of MCMC chains. The minimum required number of chains is 2
- `quiet`: if `TRUE`, does not print messages and progress bars from JAGS

**Details**

Because some individuals may die during the observation period, the reproduction rate alone is not sufficient to account for the observed number of offspring at a given time point. In addition, we need the time individuals have stayed alive during this observation period. The `reproFitTT` function estimates the number of individual-days in an experiment between its start and the target time. This covariable is then used to estimate a relation between the chemical compound concentration and the reproduction rate per individual-day.

The `reproFitTT` function fits two models, one where inter-individual variability is neglected ("Poisson" model) and one where it is taken into account ("gamma-Poisson" model). When setting `stoc.part` to "bestfit", a model comparison procedure is used to choose between both. More details are presented in the vignette accompanying the package.

**Value**

The function returns an object of class `reproFitTT` which is a list of the following objects:

- `dic`: DIC value of the selected model
- `estim.EC`: a table of the estimated 5, 10, 20 and 50 % effective concentrations (by default) and their 95 % credible intervals
- `estim.par`: a table of the estimated parameters as medians and 95 % credible intervals
- `mcmc`: an object of class `mcmc.list` with the posterior distribution
- `model`: a JAGS model object
- `warnings`: a data.frame with warning messages
- `model.label`: a character string, "P" if the Poisson model is used, "GP" if the gamma-Poisson is used
- `parameters`: a list of the parameter names used in the model
- `n.chains`: an integer value corresponding to the number of chains used for the MCMC computation
summary.reproData

n.iter  a list of two indices indicating the beginning and the end of monitored iterations
n.thin  a numerical value corresponding to the thinning interval
jags.data a list of the data passed to the jags model
transformed.data the survData object passed to the function
dataTT  the dataset with which the parameters are estimated

Examples

# (1) Load the data
data(cadmium1)

# (2) Create an object of class "reproData"
dataset <- reproData(cadmium1)

## Not run:
# (3) Run the reproFitTT function with the log-logistic gamma-Poisson model
out <- reproFitTT(dataset, stoc.part = "gammmapoisson",
                  ecx = c(5, 10, 15, 20, 30, 50, 80), quiet = TRUE)

## End(Not run)

summary.reproData  Summary of reproData object

Description

This is the generic summary S3 method for the reproData class. It provides information about the structure of the dataset and the experimental design.

Usage

## S3 method for class 'reproData'
summary(object, quiet = FALSE, ...)

Arguments

object an object of class reproData
quiet if TRUE, does not print
... Further arguments to be passed to generic methods

Value

The function returns a list with the same information than summary.survDataCstExp plus an additional one:

NboffTimeConc  nb of offspring for all concentrations and time points
Examples

```r
# (1) Load the data
data(cadmium1)

# (2) Create a reproData object
cadmium1 <- reproData(cadmium1)

# (3) Summarize the dataset
summary(cadmium1)
```

---

**summary.reproFitTT**

**Summary of reproFitTT object**

**Description**

This is the generic `summary` S3 method for the `reproFitTT` class. It shows the quantiles of priors and posteriors on parameters and the quantiles of the posterior on the ECx estimates.

**Usage**

```r
## S3 method for class 'reproFitTT'
summary(object, quiet = FALSE, ...)
```

**Arguments**

- `object`: an object of class `reproFitTT`
- `quiet`: when `TRUE`, does not print
- `...`: Further arguments to be passed to generic methods

**Value**

The function returns a list with the following information:

- `qpriors`: quantiles of the model priors
- `qposteriors`: quantiles of the model posteriors
- `qECx`: quantiles of ECx estimates

**Examples**

```r
# (1) Load the data
data(cadmium1)

# (2) Create a reproData object
cadmium1 <- reproData(cadmium1)

## Not run:
# (3) Run the reproFitTT function with the log-logistic
```
summary.survDataCstExp

# model
out <- reproFitTT(cadmium1, ecx = c(5, 10, 15, 20, 30, 50, 80), quiet = TRUE)

# (4) summarize the reproFitTT object
summary(out)

## End(Not run)

---

**Summary of survDataCstExp object**

**Description**

The generic `summary` S3 method for the `survDataCstExp` class provides information about the structure of the dataset and the experimental design.

**Usage**

```r
## S3 method for class 'survDataCstExp'
summary(object, quiet = FALSE, ...)
```

**Arguments**

- `object`: an object of class `survDataCstExp`
- `quiet`: when `TRUE`, does not print
- `...`: Further arguments to be passed to generic methods

**Value**

The function returns a list with the following information:

- `NbreptTimeConc`: nb of replicates for all concentrations and time points
- `NbsurvTimeConc`: nb of survivors for all concentrations and time points

**Examples**

```r
# (1) Load the data
data(cadmium1)

# (2) Create a survDataCstExp object
dat <- survData(cadmium1)

# (3) Summarize the dataset
summary(dat)
```
summary.survDataVarExp

Summary of survDataVarExp object

Description

The generic summary S3 method for the survDataVarExp class provides information about the structure of the dataset and the experimental design.

Usage

```r
## S3 method for class 'survDataVarExp'
summary(object, quiet = FALSE, ...)
```

Arguments

- `object` an object of class survDataVarExp
- `quiet` when TRUE, does not print
- `...` Further arguments to be passed to generic methods

Value

The function returns a list with the following information:

- `OccRepTime` Occurence of replicates for all time points
- `NbsurvTimeRep` nb of survivors for all replicates and time points
- `ConcTimeRep` Concentration for all replicates and time points

Examples

```r
# (1) Load the data
data(propiconazole_pulse_exposure)

# (2) Create a survDataVarExp object
out <- survData(propiconazole_pulse_exposure)

# (3) Summarize the dataset
summary(out)
```
Summary of survFit object

Description

This is the generic summary S3 method for the survFit class. It shows the quantiles of priors and posteriors on parameters.

Usage

## S3 method for class 'survFit'
summary(object, quiet = FALSE, ...)

Arguments

- object: an object of class survFit
- quiet: when TRUE, does not print
- ...: Further arguments to be passed to generic methods.

Value

The function returns a list with the following information:

- q priors: quantiles of the model priors
- q posteriors: quantiles of the model posteriors

Examples

```r
# (1) Load the data
data(propiconazole)

# (2) Create a survData object
dat <- survData(propiconazole)

## Not run:
# (3) Run the survFit function
out <- survFit(dat, model_type = "SD")

# (4) summarize the survFit object
summary(out)

## End(Not run)
```
Description

This is the generic summary S3 method for the survFitTKTD class. It shows the quantiles of priors and posteriors on parameters.

Usage

```r
## S3 method for class 'survFitTKTD'
summary(object, quiet = FALSE, ...)
```

Arguments

- `object`: an object of class survFitTKTD
- `quiet`: when TRUE, does not print
- `...`: Further arguments to be passed to generic methods.

Value

The function returns a list with the following information:

- `qpriors`: quantiles of the model priors
- `qposteriors`: quantiles of the model posteriors

Examples

```r
# (1) Load the data
data(propiconazole)

# (2) Create a survData object
dat <- survData(propiconazole)

## Not run:
# (3) Run the survFitTKTD function
out <- survFitTKTD(dat)

# (4) summarize the survFitTKTD object
summary(out)

## End(Not run)
```
Summary of `survFitTT` object

Description

This is the generic summary S3 method for the `survFitTT` class. It shows the quantiles of priors and posteriors on parameters and the quantiles of the posteriors on the LCx estimates.

Usage

```r
## S3 method for class 'survFitTT'
summary(object, quiet = FALSE, ...)
```

Arguments

- `object`: an object of class `survFitTT`
- `quiet`: when TRUE, does not print
- `...`: Further arguments to be passed to generic methods

Value

The function returns a list with the following information:

- `qpriors`: quantiles of the model priors
- `qposteriors`: quantiles of the model posteriors
- `QLCx`: quantiles of LCx estimates

Examples

```r
# (1) Load the data
data(cadmium1)

# (2) Create a survData object
cadmium1 <- survData(cadmium1)

## Not run:
# (3) Run the survFitTT function with the log-logistic
# binomial model
out <- survFitTT(cadmium1, lcx = c(5, 10, 15, 20, 30, 50, 80),
                 quiet = TRUE)

# (4) Summarize the survFitTT object
summary(out)

## End(Not run)
```
survData: Creates a dataset for survival analysis

Description

This function creates a survData object from experimental data provided as a data.frame. The resulting object can then be used for plotting and model fitting. It can also be used to generate individual-time estimates.

Usage

survData(x)

Arguments

x a data.frame containing the following four columns:
   - replicate: a vector of class integer or factor for replicate identification. A given replicate value should identify the same group of individuals followed in time
   - conc: a vector of class numeric with tested concentrations (positive values, may contain NAs)
   - time: a vector of class integer with time points, minimal value must be 0
   - Nsurv: a vector of class integer providing the number of alive individuals at each time point for each concentration and each replicate (may contain NAs)

Details

Survival datasets can be under either constant or time-variable exposure profile. The resulting object, in addition to its survData class, inherits the class survDataCstExp or survDataVarExp respectively.

The x argument describes experimental results from a survival toxicity test. Each line of the data.frame corresponds to one experimental measurement, that is a number of alive individuals at a given concentration at a given time point and in a given replicate. Note that either the concentration or the number of alive individuals may be missing. The dataset is inferred to be under constant exposure if the concentration is constant for each replicate and systematically available. The function survData fails if x does not meet the expected requirements. Please run survDataCheck to ensure x is well-formed.

Value

A dataframe of class survData.

See Also

survDataCheck
Examples

# (1) Load the survival dataset
data(zinc)

# (2) Create an objet of class 'survData'
dat <- survData(zinc)
class(dat)

---

survDataCheck  Checks if an object can be used to perform survival analysis

Description

The survDataCheck function can be used to check if an object containing survival data is formatted according to the expectations of the survData function.

Usage

survDataCheck(data, diagnosis.plot = FALSE)

Arguments

data any object
diagnosis.plot if TRUE, the function may produce diagnosis plots

Value

The function returns a dataframe of class msgTable and data.frame with two columns: id and msg of character strings. When no error is detected the object is empty. Here is the list of possible error ids with their meaning:

dataframeExpected an object of class data.frame is expected
missingColumn at least one expected column heading is missing
firstTime0 the first time point for some (concentration, replicate) couples is not 0
concNumeric column conc contains a value of class other than numeric
timeNumeric column time contains a value of class other than numeric
NsurvInteger column Nsurv contains a value of class other than integer
tablePositive some data are negative
Nsurv0T0 Nsurv is 0 at time 0 for some (concentration, replicate)
duplicateID there are two identical (replicate, time) couples
NsurvIncrease Nsurv increases at some time point of some (concentration, replicate)
maxTimeDiffer maximum time for concentration is lower than maximum time for survival
Note

If an error of type `dataframeExpected` or `missingColumn` is detected, the function `survDataCheck` is stopped before looking for other errors.

See Also

`survData`

Examples

```r
# Run the check data function
data(zinc)
survDataCheck(zinc)

# Now we insert an error in the dataset, by artificially increasing the
# number of survivors at a given time point, in such a way that the number
# of individuals increases in the corresponding replicate
zinc[25, "Nsurv"] <- as.integer(20)
survDataCheck(zinc, diagnosis.plot = TRUE)
```

Description

This function joins two datasets, one for exposure measurements, the other for survival measurements, into a single dataframe that can be used with the `survData` function.

Usage

```r
survData_join(x, y)
```

Arguments

- `x` a `data.frame` containing the following three columns:
  - `replicate`: a vector of class `integer` or `factor` for replicate identification
  - `time`: a vector of class `integer` with time points, min value must be 0
  - `Nsurv`: a vector of class `integer` providing the number of alive individuals at some or all time points for each replicate

- `y` a `data.frame` containing the following three columns:
  - `replicate`: a vector of class `integer` or `factor` for replicate identification
  - `time`: a vector of class `integer` with time points, min value must be 0
  - `conc`: a vector of class `numeric` providing the concentration at some or all time points for each replicate
**survFit**

**Value**

a dataframe suitable for 'survData'

**Examples**

```r
# (1) Load the two survival datasets
data(propiconazole_pulse_exposure)
exposure <- propiconazole_pulse_exposure[,c("replicate", "time", "conc")]
survival <- propiconazole_pulse_exposure[,c("replicate", "time", "Nsurv")]

# (2) Create an objet of class 'survData'
dat_join <- survData(survData_join(exposure, survival))
class(dat_join)
```

---

**survFit**  
*Method to fit a model for survival data using Bayesian inference*

**Description**

Method to fit a model for survival data using Bayesian inference

**Usage**

```r
survFit(data, ...)
```

**Arguments**

- **data** an object used to select a method 'survFit'
- **...** Further arguments to be passed to generic methods

---

**survFit.survDataCstExp**

*Fits a TKTD model for survival analysis using Bayesian inference for survDataCstExp object*

---

**Description**

This function estimates the parameters of a TKTD model ('SD' or 'IT') for survival analysis using Bayesian inference. In this model, the survival rate of individuals is modeled as a function of the chemical compound concentration with a mechanistic description of the effects on survival over time.
Usage

```r
## S3 method for class 'survDataCstExp'
survFit(data, model_type = NULL, quiet = FALSE,
n.chains = 3, n.adapt = 3000, n.iter = NULL, n.warmup = NULL,
thin.interval = NULL, limit.sampling = TRUE, dic.compute = FALSE,
dic.type = "pD", ...)
```

Arguments

- **data**: An object of class `survDataCstExp`.
- **model_type**: can be "SD" or "IT" to choose between "Stochastic Death" or "Individual Tolerance" models.
- **quiet**: If `FALSE`, prints logs and progress bar from JAGS.
- **n.chains**: A positive integer specifying the number of MCMC chains. The minimum required number of chains is 2.
- **n.adapt**: A positive integer specifying the number of iterations for adaptation. If `n.adapt = 0` then no adaptation takes place.
- **n.iter**: A positive integer specifying the number of iterations to monitor for each chain.
- **n.warmup**: A positive integer specifying the number of warmup (aka burnin) iterations per chain.
- **thin.interval**: A positive integer specifying the period to monitor.
- **limit.sampling**: If `FALSE` (default is `TRUE`), there is no limit to the number of iterations in MCMC imposed by the `raftery.diag` test.
- **dic.compute**: if `TRUE` (default is `FALSE`), it generates penalized deviance samples to compute the Deviance Information Criterion (DIC) with the `rjags` package.
- **dic.type**: type of penalty to use. A string identifying the type of penalty: `pD` or `popt` (see function `dic.samples`)
- **...**: Further arguments to be passed to generic methods

Details

The function `survFit` return the parameter estimates of Toxicokinetic-toxicodynamic (TK-TD) models SD for 'Stochastic Death' or IT for 'Individual Tolerance'. TK-TD models, and particularly the General Unified Threshold model of Survival (GUTS), provide a consistent process-based framework to analyse both time and concentration dependent datasets. In GUTS-SD, all organisms are assumed to have the same internal concentration threshold (denoted $z$), and, once exceeded, the instantaneous probability to die increases linearly with the internal concentration. In GUTS-IT, the threshold concentration is distributed among all the organisms, and once exceeded in one individual, this individual dies immediately.

Value

The function returns an object of class `survFitCstExp`, which is a list with the following information:

- **estim.par**: a table of the estimated parameters as medians and 95% credible intervals
mcmc: an object of class mcmc.list with the posterior distribution
model: a JAGS model object
dic: return the Deviance Information Criterion (DIC) if dic.compute is TRUE
warnings: a table with warning messages
parameters: a list of parameter names used in the model
n.chains: an integer value corresponding to the number of chains used for the MCMC computation
mcmcInfo: a table with the number of iterations, chains, adaptation, warmup and the thinning interval.
jags.data: a list of the data passed to the JAGS model
model_type: the type of TK-TD model used: SD or IT

References


Examples

# (1) Load the survival data
data(propiconazole)

# (2) Create an object of class "survData"
dataset <- survData(propiconazole)

## Not run:
# (3) Run the survFit function with TK-TD model 'SD' or 'IT'
out <- survFit(dataset, model_type = "SD")

# (4) Summarize look the estimated parameters
summary(out)

# (5) Plot the fitted curve
plot(out, adddata = TRUE)

# (6) Plot the fitted curve with ggplot style and CI as spaghetti
plot(out, spaghetti = TRUE, adddata = TRUE)

## End(Not run)
survFit.survDataVarExp

Fits a TKTD model for survival analysis using Bayesian inference for survDataVarExp object

Description

This function estimates the parameters of a TKTD ('SD' or 'IT') model for survival analysis using Bayesian inference. In this model, the survival rate of individuals is modeled as a function of the chemical compound concentration with a mechanistic description of the effects on survival over time.

Usage

## S3 method for class 'survDataVarExp'
survFit(data, model_type = NULL, quiet = FALSE,
        extend_time = 100, n.chains = 3, n.adapt = 1000, n.iter = NULL,
        n.warmup = NULL, thin.interval = NULL, limit.sampling = TRUE,
        dic.compute = FALSE, dic.type = "pD", ...)

Arguments

data An object of class survDataVarExp.
model_type can be "SD" or "IT" to choose between "Stochastic Death" or "Individual Tolerance" models (resp.). See modeling vignette for details.
quiet If FALSE, prints logs and progress bar from JAGS.
extend_time Number of for each replicate used for linear interpolation (comprise between time to compute and fitting accuracy)
n.chains A positive integer specifying the number of MCMC chains. The minimum required number of chains is 2.
n.adapt A positive integer specifying the number of iterations for adaptation. If n.adapt = 0 then no adaptation takes place.
n.iter A positive integer specifying the number of iterations to monitor for each chain.
n.warmup A positive integer specifying the number of warmup (aka burnin) iterations per chain.
thin.interval A positive integer specifying the period to monitor.
limit.sampling if FALSE (default is TRUE), there is no limit to the number of iterations in MCMC imposed by the raftery.diag test.
dic.compute if TRUE (default is FALSE), it generates penalized deviance samples to compute the Deviance Information Criterion (DIC) with the rjags package
dic.type type of penalty to use. A string identifying the type of penalty: pD or popt (see function dic.samples)
...
Further arguments to be passed to generic methods
Details

The function `survfit` return the parameter estimates of Toxicokinetic-toxicodynamic (TK-TD) models SD for 'Stochastic Death' or IT for 'Individual Tolerance'. TK-TD models, and particularly the General Unified Threshold model of Survival (GUTS), provide a consistent process-based framework to analyse both time and concentration dependent datasets. In GUTS-SD, all organisms are assumed to have the same internal concentration threshold (denoted \( z \)), and, once exceeded, the instantaneous probability to die increases linearly with the internal concentration. In GUTS-IT, the threshold concentration is distributed among all the organisms, and once exceeded in one individual, this individual dies immediately.

Value

The function returns an object of class `survfitVarExp`, which is a list with the following information:

- `estim.par`: a table of the estimated parameters as medians and 95% credible intervals
- `mcmc`: an object of class `mcmc.list` with the posterior distribution
- `model`: a JAGS model object
- `dic`: return the Deviance Information Criterion (DIC) if `dic.compute` is TRUE
- `warnings`: a table with warning messages
- `parameters`: a list of parameter names used in the model
- `n.chains`: an integer value corresponding to the number of chains used for the MCMC computation
- `mcmcInfo`: a table with the number of iterations, chains, adaptation, warmup and the thinning interval
- `jags.data`: a list of the data passed to the JAGS model
- `model_type`: the type of TK-TD model used: SD or IT

References


Examples

```
# (1) Load the survival data
data("propiconazole_pulse_exposure")

# (2) Create an object of class "survData"
dataset <- survData(propiconazole_pulse_exposure)

## Not run:
# (3) Run the survFit function with TK-TD model 'SD' or 'IT'
out <- survFit(dataset, model_type = "SD")
```
# (4) Summarize look the estimated parameters
summary(out)

# (5) Plot the fitted curve
plot(out, adddata = FALSE)

# (6) Plot the fitted curve with ggplot style and CI as spaghetti
plot(out, spaghetti = TRUE)

## End(Not run)

```r
survFitTKTD
```

**Fits a TKTD for survival analysis using Bayesian inference for survDataTKTD object**

**Description**

This function estimates the parameters of a TKTD model for survival analysis using Bayesian inference. In this model, the survival rate of individuals is modeled as a function of the chemical compound concentration with a mechanistic description of the effects on survival over time.

**Usage**

```r
survFitTKTD(data, n.chains = 3, quiet = FALSE)
```

**Arguments**

- `data` An object of class `survData`.
- `n.chains` Number of MCMC chains. The minimum required number of chains is 2.
- `quiet` If FALSE, prints logs and progress bar from JAGS.

**Value**

The function returns an object of class `survFitTKTD`, which is a list with the following information:

- `estim.par` a table of the estimated parameters as medians and 95% credible intervals
- `mcmc` an object of class `mcmc.list` with the posterior distribution
- `warnings` a table with warning messages
- `model` a JAGS model object
- `parameters` a list of parameter names used in the model
- `n.chains` an integer value corresponding to the number of chains used for the MCMC computation
- `n.iter` a list of two indices indicating the beginning and the end of monitored iterations
- `n.thin` a numerical value corresponding to the thinning interval
- `jags.data` a list of data passed to the JAGS model
References


Examples

```r
# (1) Load the survival data
data(propiconazole)

# (2) Create an object of class "survData"
dataset <- survData(propiconazole)

## Not run:
# (3) Run the survFitTKTD function
out <- survFitTKTD(dataset)

# (4) Summarize look the estimated parameters
summary(out)

# (5) Plot the fitted curve
plot(out, adddata = TRUE)

# (6) Plot the fitted curve with ggplot style and CI as spaghetti
plot(out, spaghetti = TRUE, adddata = TRUE, style = "ggplot")

## End(Not run)
```

---

**survFitTT**

*Fits a Bayesian concentration-response model for target-time survival analysis*

**Description**

Fits a Bayesian concentration-response model for target-time survival analysis

**Usage**

```r
survFitTT(data, ...)
```

**Arguments**

- `data`:
  - an object used to select a method 'survFitTT'

- `...`:
  - Further arguments to be passed to generic methods
survFitTT.survDataCstExp

Fits a Bayesian concentration-response model for target-time survival analysis

Description

This function estimates the parameters of a concentration-response model for target-time survival analysis using Bayesian inference. In this model, the survival rate of individuals at a given time point (called target time) is modeled as a function of the chemical compound concentration. The actual number of surviving individuals is then modeled as a stochastic function of the survival rate. Details of the model are presented in the vignette accompanying the package.

Usage

```r
## S3 method for class 'survDataCstExp'
survFitTT(data, target.time = NULL, lcx = c(5, 10, 20, 50), n.chains = 3, quiet = FALSE, ...)
```

Arguments

- `data`: an object of class `survData`
- `target.time`: the chosen endpoint to evaluate the effect of the chemical compound concentration, by default the last time point available for all concentrations
- `lcx`: desired values of $x$ (in percent) for which to compute $LC_x$.
- `n.chains`: number of MCMC chains, the minimum required number of chains is 2
- `quiet`: if TRUE, does not print messages and progress bars from JAGS
- `...`: Further arguments to be passed to generic methods

Details

The function returns parameter estimates of the concentration-response model and estimates of the so-called $LC_x$, that is the concentration of chemical compound required to get an $(1 - x/100)$ survival rate.

Value

The function returns an object of class `survFitTT`, which is a list with the following information:

- `estim.LCx`: a table of the estimated $LC_x$ along with their 95% credible intervals
- `estim.par`: a table of the estimated parameters (medians) and 95% credible intervals
- `det.part`: the name of the deterministic part of the used model
- `mcmc`: an object of class `mcmc.list` with the posterior distribution
- `warnings`: a table with warning messages
model  a JAGS model object
parameters  a list of parameter names used in the model
n.chains  an integer value corresponding to the number of chains used for the MCMC computation
n.iter  a list of two indices indicating the beginning and the end of monitored iterations
n.thin  a numerical value corresponding to the thinning interval
jags.data  a list of the data passed to the JAGS model
transformed.data  the survData object passed to the function
dataTT  the dataset with which the parameters are estimated

Examples

# (1) Load the data
data(cadmium1)

# (2) Create an object of class "survData"
dat <- survData(cadmium1)

## Not run:
# (3) Run the survFitTT function with the log-logistic
# binomial model
out <- survFitTT(dat, lcx = c(5, 10, 15, 20, 30, 50, 80), quiet = TRUE)

## End(Not run)

---

zinc  **Reproduction and survival datasets for Daphnia magna exposed to zinc during 21 days**

Description

Reproduction and survival datasets of a chronic laboratory toxicity tests with *Daphnia magna* freshwater invertebrate exposed to four concentrations of zinc during 21 days. Four concentrations were tested with three replicates per concentration. Each replicate contained 20 organisms. Reproduction and survival were monitored at 15 time points.

Usage

data(zinc)
Format

A data frame with 180 observations on the following five variables:

- `replicate` A vector of class `numeric` with the replicate code (1 to 12).
- `conc` A vector of class `numeric` with zinc concentrations in \( mg.L^{-1} \).
- `time` A vector of class `integer` with the time points (in days from the beginning of the experiment \( t = 0 \)).
- `nsurv` A vector of class `integer` with the number of alive individuals at each time point for each concentration and each replicate.
- `nrepro` A vector of class `integer` with the number of offspring at each time point for each concentration and each replicate.

References

Index

*Topic check
reproDataCheck, 46

*Topic dataset
cadmium1, 5
cadmium2, 5
chlordan, 6
copper, 7
dichromate, 8
propiconazole, 44
propiconazole_pulse_exposure, 45
zinc, 67

*Topic estimation
reprofitt, 47
survfit.survDataCstExp, 59
survfit.survDataVarExp, 62
survfitTKTD, 64
survfitTT.survDataCstExp, 66

*Topic package
morse-package, 3

*Topic plot
plot.LCx, 14
plot.MFx, 15
plot.reproData, 16
plot.reprofitt, 17
plot.survDataCstExp, 19
plot.survDataVarExp, 20
plot.survFitCstExp, 21
plot.survFitPredict, 22
plot.survFitTKTD, 23
plot.survFitTT, 25
plot.survFitVarExp, 26
plotDoseResponse.reproData, 29
plotDoseResponse.survDataCstExp, 30

*Topic print
print.reprofitt, 38
print.survFitCstExp, 39
print.survFitTKTD, 40
print.survFitTT, 41
print.survFitVarExp, 41

*Topic summary
summary.reproData, 49
summary.reprofitt, 50
summary.survDataCstExp, 51
summary.survDataVarExp, 52
summary.survfit, 53
summary.survfitTKTD, 54
summary.survfitTT, 55

*Topic transformation
reproData, 45
survData, 56

binom.test, 31
cadmium1, 5
cadmium2, 5
chlordan, 6
copper, 7
dic.samples, 60, 62
dichromate, 8
ggplot, 17–20, 26, 30, 31
ggplot2, 4
is_exposure_constant, 9
LCx, 9
LCx.survFit, 10, 10
MFx, 11
MFx.survFit, 12, 12
morse (morse-package), 3
morse-package, 3
plot, 17, 18, 30, 31
plot.LCx, 14
plot.MFx, 15
plot.reproData, 16
plot.reprofitt, 17