Package 'drc'

August 30, 2016

Version 3.0-1

Date 2016-08-25

Title Analysis of Dose-Response Curves

Author Christian Ritz <ritz@bioassay.dk>, Jens C. Strebig <streibig@bioassay.dk>

Maintainer Christian Ritz <ritz@bioassay.dk>

Depends R (>= 2.0.0), MASS, stats

Imports car, gtools, multcomp, plotrix, scales

LazyLoad yes

LazyData yes

Description Analysis of dose-response data is made available through a suite of flexible and versatile model fitting and after-fitting functions.

License GPL-2 | file LICENCE

URL http://www.r-project.org, http://www.bioassay.dk

NeedsCompilation no

Repository CRAN

Date/Publication 2016-08-30 01:33:38

R topics documented:

acidiq *Acifluorfen and diquat tested on Lemna minor.*

Description

Data from an experiment where the chemicals acifluorfen and diquat tested on Lemna minor. The dataset has 7 mixtures used in 8 dilutions with three replicates and 12 common controls, in total 180 observations.

Usage

```
data(acidiq)
```

Format

A data frame with 180 observations on the following 3 variables.

`dose` a numeric vector of dose values

`pct` a numeric vector denoting the grouping according to the mixtures percentages

`rgr` a numeric vector of response values (relative growth rates)

Details

The dataset is analysed in Soerensen et al (2007). Hewlett's symmetric model seems appropriate for this dataset.

Source

The dataset is kindly provided by Nina Cedergreen, Department of Agricultural Sciences, Royal Veterinary and Agricultural University, Denmark.

References

Soerensen, H. and Cedergreen, N. and Skovgaard, I. M. and Streibig, J. C. (2007) An isobole-based statistical model and test for synergism/antagonism in binary mixture toxicity experiments, *Environmental and Ecological Statistics*, **14**, 383–397.

Examples

```
## Fitting the model with freely varying ED50 values
## Ooops: Box-Cox transformation is needed
acidiq.free <- drm(rgr ~ dose, pct, data = acidiq, fct = LL.4(),
pmodels = list(~factor(pct), ~1, ~1, ~factor(pct) - 1))

## Lack-of-fit test
modelFit(acidiq.free)
summary(acidiq.free)
```

```
## Plotting isobole structure
isobole(acidiq.free, xlim = c(0, 400), ylim = c(0, 450))

## Fitting the concentration addition model
acidiq.ca <- mixture(acidiq.free, model = "CA")

## Comparing to model with freely varying e parameter
anova(acidiq.ca, acidiq.free)  # rejected

## Plotting isobole based on concentration addition -- poor fit
isobole(acidiq.free, acidiq.ca, xlim = c(0, 420), ylim = c(0, 450))  # poor fit

## Fitting the Hewlett model
acidiq.hew <- mixture(acidiq.free, model = "Hewlett")

## Comparing to model with freely varying e parameter
anova(acidiq.free, acidiq.hew)  # accepted
summary(acidiq.hew)

## Plotting isobole based on the Hewlett model
isobole(acidiq.free, acidiq.hew, xlim = c(0, 400), ylim = c(0, 450))  # good fit
```

`algae` *Volume of algae as function of increasing concentrations of a herbicide*

Description

Dataset from an experiment exploring the effect of increasing concentrations of a herbicide on the volume of the treated algae.

Usage

```
data(algae)
```

Format

A data frame with 14 observations on the following 2 variables.

`conc` a numeric vector of concentrations.

`vol` a numeric vector of response values, that is relative change in volume.

Details

This datasets requires a cubic root transformation in order to stabilise the variance.

Source

Meister, R. and van den Brink, P. (2000) *The Analysis of Laboratory Toxicity Experiments*, Chapter 4 in *Statistics in Ecotoxicology*, Editor: T. Sparks, New York: John Wiley \& Sons, (pp. 114–116).

Examples

```
algae.m1 <- drm(vol~conc, data=algae, fct=LL.3())
summary(algae.m1)

algae.m2 <- boxcox(algae.m1)
summary(algae.m2)
```

---

anova.drc *ANOVA for dose-response model fits*

Description

'anova' produces an analysis of variance table for one or two non-linear model fits.

Usage

```
  ## S3 method for class 'drc'
anova(object, ..., details = TRUE, test = NULL)
```

Arguments

object	an object of class 'drc'.
...	additional arguments.
details	logical indicating whether or not details on the models compared should be displayed. Default is TRUE (details are displayed).
test	a character string specifying the test statistic to be applied. Use "od" to assess overdispersion for binomial data.

Details

Specifying only a single object gives a test for lack-of-fit, comparing the non-linear regression model to a more general one-way or two-way ANOVA model.

If two objects are specified a test for reduction from the larger to the smaller model is given. (This only makes statistical sense if the models are nested, that is: one model is a submodel of the other model.)

Value

An object of class 'anova'.

Author(s)

Christian Ritz

References

Bates, D. M. and Watts, D. G. (1988) *Nonlinear Regression Analysis and Its Applications*, New York: Wiley \& Sons (pp. 103–104)

See Also

For comparison of nested or non-nested model the function `mselect`can also be used.

The function `anova.lm` for linear models.

Examples

```
## Comparing a Gompertz three- and four-parameter models using an F test
ryegrass.m1 <- drm(rootl ~ conc, data = ryegrass, fct = W1.4())
ryegrass.m2 <- drm(rootl ~ conc, data = ryegrass, fct = W1.3())
anova(ryegrass.m2, ryegrass.m1)  # reduction to 'W1.3' not possible (highly significant)

anova(ryegrass.m2, ryegrass.m1, details = FALSE)  # without details
```

AR *Asymptotic regression model*

Description

Providing the mean function and the corresponding self starter function for the asymptotic regression model.

Usage

```
AR.2(fixed = c(NA, NA), names = c("d", "e"), ...)

AR.3(fixed = c(NA, NA, NA), names = c("c", "d", "e"), ...)
```

Arguments

`fixed`	numeric vector. Specifies which parameters are fixed and at what value they are fixed. NAs for parameter that are not fixed.
`names`	vector of character strings giving the names of the parameters (should not contain ":").
`...`	additional arguments to be passed from the convenience functions.

Details

The asymptotic regression model is a three-parameter model with mean function:

$$f(x) = c + (d - c)(1 - \exp(-x/e))$$

The parameter c is the lower limit (at $x = 0$), the parameter d is the upper limit and the parameter $e > 0$ is determining the steepness of the increase as x.

Value

A list of class `drcMean`, containing the mean function, the self starter function, the parameter names and other components such as derivatives and a function for calculating ED values.

Note

The functions are for use with the function `drm`.

Author(s)

Christian Ritz

See Also

A very similar, but monotonously decreasing model is the exponential decay model: `EXD.2` and `EXD.3`.

Examples

```
## First model
met.as.m1<-drm(gain ~ dose, product, data = methionine, fct = AR.3(),
pmodels = list(~1, ~factor(product), ~factor(product)))
plot(met.as.m1, log = "", ylim = c(1450, 1800))
summary(met.as.m1)

## Calculating bioefficacy: approach 1
coef(met.as.m1)[5] / coef(met.as.m1)[4] * 100

## Calculating bioefficacy: approach 2
EDcomp(met.as.m1, c(50,50))

## Simplified models
met.as.m2<-drm(gain ~ dose, product, data = methionine, fct = AR.3(),
pmodels = list(~1, ~1, ~factor(product)))
anova(met.as.m2, met.as.m1)  # simplification not possible

met.as.m3 <- drm(gain ~ dose, product, data = methionine, fct = AR.3(),
pmodels = list(~1, ~factor(product), ~1))
anova(met.as.m3, met.as.m1)  # simplification not possible
```

auxins *Effect of technical grade and commercially formulated auxin herbicides*

Description

MCPA, 2,4-D, mecorprop and dichorlprop were applied either as technical grades materials (h = 1, 2, 3, 4) or as commercial formulations (herb = 5, 6, 7, 8). Each experimental unit consisted of five 1-week old seedlings grown together in a pot of nutrient solution during 14 days.

Usage

```
data(auxins)
```

Format

A data frame with 150 observations on the following 5 variables.

`r` a numeric vector

`h` a numeric vector

`w` a numeric vector

`y` a numeric vector

`dose` a numeric vector

Details

Data are parts of a larger joint action experiment with various herbicides.

The eight herbicide preparations are naturally grouped into four pairs: (1, 5), (2, 6), (3, 7), and (4, 8), and in each pair of herbicides should have the same active ingredients but different formulation constituents, which were assumed to be biologically inert. The data consist of the 150 observations y of dry weights, each observation being the weight of five plants grown in the same pot. All the eight herbicide preparations have essentially the same mode of action in the plant; they all act like the plant auxins, which are plant regulators that affect cell enlongation an other essential metabolic pathways. One of the objects of the experiment was to test if the response functions were identical except for a multiplicative factor in the dose. This is a necessary, but not a sufficient, condition for a similar mode of action for the herbicides.

Source

Streibig, J. C. (1987). Joint action of root-absorbed mixtures of auxin herbicides in Sinapis alba L. and barley (Hordeum vulgare L.) *Weed Research*, **27**, 337–347.

References

Rudemo, M., Ruppert, D., and Streibig, J. C. (1989). Random-Effect Models in Nonlinear Regression with Applications to Bioassay. *Biometrics*, **45**, 349–362.

Examples

```
## Fitting model with varying lower limits
auxins.m1 <- boxcox(drm(y ~ dose, h,
pmodels = data.frame(h, h, 1, h), fct = LL.4(), data = auxins), method = "anova")

## Fitting model with common lower limit
auxins.m2 <- boxcox(drm(y ~ dose, h,
pmodels = data.frame(h, 1, 1, h), fct = LL.4(), data = auxins), method = "anova")

## Comparing the two models
anova(auxins.m2, auxins.m1)
```

---

backfit *Calculation of backfit values from a fitted dose-response model*

Description

By inverse regression backfitted dose values are calculated for the mean response per dose.

Usage

```
backfit(drcObject)
```

Arguments

drcObject	an object of class 'drc'.

Value

Two columns with the original dose values and the corresponding backfitted values using the fitted dose-response model. For extreme dose values (e.g., high dose) the backfitted values may not be well-defined (see the example below).

Author(s)

Christian Ritz after a suggestion from Keld Sorensen.

References

??

See Also

A related function is ED.drc.

Examples

```
ryegrass.LL.4 <- drm(rootl~conc, data=ryegrass, fct=LL.4())

backfit(ryegrass.LL.4)
```

---

baro5 *The modified baro5 function*

Description

'baro5' allows specification of the baroreflex 5-parameter dose response function, under various constraints on the parameters.

Usage

```
baro5(fixed = c(NA, NA, NA, NA, NA), names = c("b1", "b2", "c", "d", "e"),
method = c("1", "2", "3", "4"), ssfct = NULL)
```

Arguments

fixed	numeric vector. Specifies which parameters are fixed and at what value they are fixed. NAs for parameter that are not fixed.
names	a vector of character strings giving the names of the parameters (should not contain ":"). The order of the parameters is: b1, b2, c, d, e (see under 'Details').
method	character string indicating the self starter function to use.
ssfct	a self starter function to be used.

Details

The five-parameter function given by the expression

$$y = c + \frac{d - c}{1 + f\exp(b1(\log(x) - \log(e))) + (1 - f)\exp(b2(\log(x) - \log(e)))}$$

$$f = 1/(1 + \exp((2b1b2/|b1 + b2|)(\log(x) - \log(e))))$$

If the difference between the parameters b1 and b2 is different from 0 then the function is asymmetric.

Value

The value returned is a list containing the nonlinear model function, the self starter function and the parameter names.

Note

See the example for the dataset heartrate.

Author(s)

Christian Ritz

References

Ricketts, J. H. and Head, G. A. (1999) A five-parameter logistic equation for investigating asymmetry of curvature in baroreflex studies. *Am. J. Physiol. (Regulatory Integrative Comp. Physiol. 46)*, **277**, 441–454.

BC.5 *The Brain-Cousens hormesis models*

Description

'BC.4' and 'BC.5' provide the Brain-Cousens modified log-logistic models for describing u-shaped hormesis.

Usage

```
BC.5(fixed = c(NA, NA, NA, NA, NA), names = c("b", "c", "d", "e", "f"), ...)

BC.4(fixed = c(NA, NA, NA, NA), names = c("b", "d", "e", "f"), ...)
```

Arguments

fixed	numeric vector specifying which parameters are fixed and at which values they are fixed. NAs designate parameters that are not fixed.
names	a vector of character strings giving the names of the parameters.
...	additional arguments to be passed from the convenience functions.

Details

The model function for the Brain-Cousens model (Brain and Cousens, 1989) is

$$f(x, b, c, d, e, f) = c + \frac{d - c + fx}{1 + \exp(b(\log(x) - \log(e)))}$$

,

and it is a five-parameter model, obtained by extending the four-parameter log-logistic model (LL.4 to take into account inverse u-shaped hormesis effects.

The parameters have the following interpretations

- b: Not direct interpretation

- c: Lower horizontal asymptote
- d: Upper horizontal asymptote
- e: Not direct interpretation
- f: Size of the hormesis effect: the larger the value the larger is the hormesis effect. $f = 0$ corresponds to no hormesis effect and the resulting model is the four-parameter log-logistic model. This parameter should be positive in order for the model to make sense.

Fixing the lower limit at 0 yields the four-parameter model

$$f(x) = 0 + \frac{d - 0 + fx}{1 + \exp(b(\log(x) - \log(e)))}$$

used by van Ewijk and Hoekstra (1993).

Value

See braincousens.

Note

This function is for use with the function drm.

Author(s)

Christian Ritz

References

Brain, P. and Cousens, R. (1989) An equation to describe dose responses where there is stimulation of growth at low doses, *Weed Research*, **29**, 93–96.

van Ewijk, P. H. and Hoekstra, J. A. (1993) Calculation of the EC50 and its Confidence Interval When Subtoxic Stimulus Is Present, *Ecotoxicology and Environmental Safety*, **25**, 25–32.

See Also

More details are found for the general model function braincousens.

Examples

```
## Fitting the data in van Ewijk and Hoekstra (1993)
lettuce.bcm1 <- drm(weight ~ conc, data = lettuce, fct=BC.5())
modelFit(lettuce.bcm1)
plot(lettuce.bcm1)

lettuce.bcm2 <- drm(weight ~conc, data = lettuce, fct=BC.4())
summary(lettuce.bcm2)
ED(lettuce.bcm2, c(50))
# compare the parameter estimate and
# its estimated standard error
```

```
# to the values in the paper by
# van Ewijk and Hoekstra (1993)


## Brain-Cousens model with the constraint b>3
ryegrass.bcm1 <- drm(rootl ~conc, data = ryegrass, fct = BC.5(),
lower = c(3, -Inf, -Inf, -Inf, -Inf), control = drmc(constr=TRUE))

summary(ryegrass.bcm1)

## Brain-Cousens model with the constraint f>0
## (no effect as the estimate of f is positive anyway)
ryegrass.bcm2 <- drm(rootl ~conc, data = ryegrass, fct = BC.5(),
lower = c(-Inf, -Inf, -Inf, -Inf, 0), control = drmc(constr=TRUE))

summary(ryegrass.bcm2)
```

`boxcox.drc` *Transform-both-sides Box-Cox transformation*

Description

Finds the optimal Box-Cox transformation for non-linear regression.

Usage

```
## S3 method for class 'drc'
boxcox(object, lambda = seq(-2, 2, by = 0.25), plotit = TRUE, bcAdd = 0,
method = c("ml", "anova"),  level = 0.95, eps = 1/50,
xlab = expression(lambda), ylab = "log-Likelihood", ...)
```

Arguments

`object`	object of class `drc`.
`lambda`	numeric vector of lambda values; the default is (-2, 2) in steps of 0.25.
`plotit`	logical which controls whether the result should be plotted.
`bcAdd`	numeric value specifying the constant to be added on both sides prior to Box-Cox transformation. The default is 0.
`method`	character string specifying the estimation method for lambda: maximum likelihood or ANOVA-based (optimal lambda inherited from more general ANOVA model fit.
`eps`	numeric value: the tolerance for lambda = 0; defaults to 0.02.
`level`	numeric value: the confidence level required.
`xlab`	character string: the label on the x axis, defaults to "lambda".
`ylab`	character string: the label on the y axis, defaults to "log-likelihood".
`...`	additional graphical parameters.

Details

The optimal lambda value is determined using a profile likelihood approach: For each lambda value the dose-response regression model is fitted and the lambda value (and corresponding model fit) resulting in the largest value of the log likelihood function is chosen.

Value

An object of class "drc" (returned invisibly). If plotit = TRUE a plot of loglik vs lambda is shown indicating a confidence interval (by default 95 the optimal lambda value.

Author(s)

Christian Ritz

References

Carroll, R. J. and Ruppert, D. (1988) *Transformation and Weighting in Regression*, New York: Chapman and Hall (Chapter 4).

See Also

For linear regression the analogue is boxcox.

Examples

```
## Fitting log-logistic model without transformation
ryegrass.m1 <- drm(ryegrass, fct = LL.4())
summary(ryegrass.m1)

## Fitting the same model with the optimal Box-Cox transformation
ryegrass.m2 <- boxcox(ryegrass.m1)
summary(ryegrass.m2)
```

---

braincousens *The Brain-Cousens hormesis models*

Description

'braincousens' provides a very general way of specifying Brain-Cousens' modified log- logistic model for describing hormesis, under various constraints on the parameters.

Usage

```
braincousens(fixed = c(NA, NA, NA, NA, NA),
names = c("b", "c", "d", "e", "f"),
method = c("1", "2", "3", "4"), ssfct = NULL,
fctName, fctText)
```

Arguments

`fixed`	numeric vector. Specifies which parameters are fixed and at what value they are fixed. NAs for parameter that are not fixed.
`names`	a vector of character strings giving the names of the parameters (should not contain ":"). The default is reasonable (see under 'Usage'). The order of the parameters is: b, c, d, e, f (see under 'Details').
`method`	character string indicating the self starter function to use.
`ssfct`	a self starter function to be used.
`fctName`	optional character string used internally by convenience functions.
`fctText`	optional character string used internally by convenience functions.

Details

The Brain-Cousens model is given by the expression

$$f(x) = c + \frac{d - c + fx}{1 + \exp(b(\log(x) - \log(e)))}$$

which is a five-parameter model.

It is a modification of the four-parameter logistic curve to take hormesis into account proposed by Brain and Cousens (1989).

Value

The value returned is a list containing the non-linear function, the self starter function, the parameter names and additional model specific objects.

Note

This function is for use with the function `drm`.

The convenience functions of `braincousens` are `BC.4` and `BC.5`. These functions should be used rather than `braincousens` directly.

Author(s)

Christian Ritz

References

Brain, P. and Cousens, R. (1989) An equation to describe dose responses where there is stimulation of growth at low doses, *Weed Research*, **29**, 93–96.

`bread.drc` *Bread and meat for the sandwich*

Description

Bread and meat for the sandwich estimator of the variance-covariance.

Usage

```
bread.drc(x, ...)

estfun.drc(x, ...)
```

Arguments

`x`	object of class `drc`
`...`	additional arguments. At the moment none are supported

Details

The details are provided by Zeileis (2006).

Value

The unscaled hessian is returned by `bread.drc`, whereas `estfun.drc` returns the estimating function evaluated at the data and the parameter estimates.

By default no clustering is assumed, corresponding to robust standard errors under independence. If a cluster variable is provided the log likelihood contributions provided by `estfun` are summed up for each cluster.

Author(s)

Christian Ritz

References

Zeileis, A. (2006) Object-oriented Computation of Sandwich Estimators, *J. Statist. Software*, **16**, Issue 9.

Examples

```
## The lines below requires that the packages
## 'lmtest' and 'sandwich' are installed
# library(lmtest)
# library(sandwich)

# ryegrass.m1<-drm(rootl ~ conc, data = ryegrass, fct = LL.4())
```

```
# Standard summary output
# coeftest(ryegrass.m1)

# Output with robust standard errors
# coeftest(ryegrass.m1, vcov = sandwich)
```

---

cedergreen *The Cedergreen-Ritz-Streibig model*

Description

'cedergreen' provides a very general way of specifying then Cedergreen-Ritz-Streibig modified log-logistic model for describing hormesis, under various constraints on the parameters.

CRS.6 is the extension of link{cedergreen} with freely varying alpha parameter.

For u-shaped hormesis data 'ucedergreen' provides a very general way of specifying the Cedergreen-Ritz-Streibig modified log-logistic model, under various constraints on the parameters.

Usage

```
cedergreen(fixed = c(NA, NA, NA, NA, NA),
names = c("b", "c", "d", "e", "f"),
method = c("1", "2", "3", "4"), ssfct = NULL,
alpha, fctName, fctText)

CRS.6(fixed = c(NA, NA, NA, NA, NA, NA),
names = c("b","c","d","e","f","g"),
method = c("1", "2", "3", "4"), ssfct = NULL)

ucedergreen(fixed = c(NA, NA, NA, NA, NA),
names = c("b", "c", "d", "e", "f"),
method = c("1", "2", "3", "4"), ssfct = NULL,
alpha)
```

Arguments

fixed	numeric vector. Specifies which parameters are fixed and at what value they are fixed. NAs for parameter that are not fixed.
names	a vector of character strings giving the names of the parameters (should not contain ":"). The order of the parameters is: b, c, d, e, f (see under 'Details').
method	character string indicating the self starter function to use.
ssfct	a self starter function to be used.
alpha	numeric value between 0 and 1, reflecting the steepness of the hormesis peak. This argument needs to be specified.
fctName	optional character string used internally by convenience functions.
fctText	optional character string used internally by convenience functions.

Details

The model is given by the expression

$$f(x) = c + \frac{d - c + fexp(-1/(x^{\alpha}))}{1 + exp(b(log(x) - log(e)))}$$

which is a five-parameter model (alpha is fixed or freely varying). Not all features (eg EC/ED calculation) are available for the model with freely varying alpha.

It is a modification of the four-parameter logistic curve to take hormesis into account.

The u-shaped model is given by the expression

$$f(x) = cd - \frac{d - c + f\exp(-1/x^{\alpha})}{1 + \exp(b(\log(x) - \log(e)))}$$

Value

The value returned is a list containing the non-linear function, the self starter function and the parameter names.

Note

The functions are for use with the functions drm.

Author(s)

Christian Ritz

References

Cedergreen, N. and Ritz, C. and Streibig, J. C. (2005) Improved empirical models describing hormesis, *Environmental Toxicology and Chemistry* **24**, 3166–3172.

See Also

For fixed alpha, several special cases are handled by the following convenience functions CRS.4a, CRS.4b, CRS.4c, CRS.5a, CRS.5b, CRS.5c, UCRS.4a, UCRS.4b, UCRS.4c, UCRS.5a, UCRS.5b, UCRS.5c where a, b and c correspond to the pre-specified alpha values 1, 0.5 and 0.25, respectively.

Examples

```
## Estimating CRS model with alpha unknown
lettuce.crsm1 <- drm(weight~conc, data = lettuce, fct = CRS.6())
summary(lettuce.crsm1)
plot(lettuce.crsm1)  # oops: not increasing until hormesis peak
```

chickweed *Germination of common chickweed (*Stellaria media*)*

Description

Germination data from tests of chickweed seeds from chlorsulfuron resistant and sensitive biotypes

Usage

```
data(chickweed)
```

Format

A data frame with 35 observations on the following 3 variables.

`start` a numeric vector of left endpoints of the monitoring intervals

`end` a numeric vector of right endpoints of the monitoring intervals

`count` a numeric vector of the number of seeds germinated in the interval between start and end

`time` a numeric vector of the non-zero left endpoints of the monitoring intervals (often used for recording in practice)

Details

The germination tests of chickweed seeds from chlorsulfuron resistant and sensitive biotypes in central Zealand were done in petri dishes (diameter: 9.0cm) in a dark growth cabinet at a temperature of 5 degrees Celsius. The seeds were incubated for 24 hours in a 0.3% solution of potassium nitrate in order to imbibe seeds prior to the test. A total of 200 seeds were placed on filter plate. After initialization of the tests, the number of germinated seeds was recorded and removed at 34 consecutive inspection times. Definition of a germinated seed was the breakthrough of the seed testa by the radicle.

Chickweed is known to have dormant seeds and therefore we would not expect 100% germination. It means that the upper limit of the proportion germinated has to be incorporated as a parameter into a model, which adequately reflects the experimental design as well as any expectations about the resulting outcome.

Source

Data are kindly provided by Lisa Borggaard (formerly at the Faculty of Life Sciences, University of Copenhagen).

References

Ritz, C., Pipper, C. B. and Streibig, J. C. (2013) Analysis of germination data from agricultural experiments, *Europ. J. Agronomy*, **45**, 1–6.

Examples

```
## Incorrect analysis using a logistic regression model
## (treating event times as binomial data)
## The argument "type" specifies that binomial data are supplied
chickweed.m0a <- drm(count/200 ~ time, weights = rep(200, 34),
data = chickweed0, fct = LL.3(), type = "binomial")
summary(chickweed.m0a) # showing a summmary of the model fit (including parameter estimates)

## Incorrect analysis based on nonlinear regression
## LL.3() refers to the three-parameter log-logistic model
## As the argument "type" is not specified it is assumed that the data type
##  is continuous and nonlinear regression based on least squares estimation is carried out
chickweed.m0b <- drm(count/200 ~ time, data = chickweed0, fct = LL.3())
summary(chickweed.m0b)  # showing a summmary of the model fit (including parameter estimates)

## How to re-arrange the data for fitting the event-time model
## (only for illustration of the steps needed for converting a dataset,
##  but in this case not needed as both datasets are already provided in "drc")
#chickweed <- data.frame(start = c(0, chickweed0$time), end = c(chickweed0$time, Inf))
#chickweed$count <- c(0, diff(chickweed0$count), 200 - tail(chickweed0$count, 1))
#head(chickweed)  # showing top 6 lines of the dataset
#tail(chickweed)  # showing bottom 6 lines

## Fitting the event-time model (by specifying the argument type explicitly)
chickweed.m1 <- drm(count~start+end, data = chickweed, fct = LL.3(), type = "event")
summary(chickweed.m1)  # showing a summmary of the model fit (including parameter estimates)

## Summary output with robust standard errors
## library(lmtest)
## library(sandwich)
## coeftest(chickweed.m1, vcov = sandwich)

## Calculating t10, t50, t90 for the distribution of viable seeds
ED(chickweed.m1, c(10, 50, 90))

## Plotting data and fitted regression curve
plot(chickweed.m1, xlab = "Time (hours)", ylab = "Proportion germinated",
xlim=c(0, 340), ylim=c(0, 0.25), log="", lwd=2, cex=1.2)
## Adding the fitted curve obtained using nonlinear regression
plot(chickweed.m0b, add = TRUE, lty = 2, xlim=c(0, 340),
ylim=c(0, 0.25), log="", lwd=2, cex=1.2)
# Note: the event-time model has slightly better fit at the upper limit

## Enhancing the plot (to look like in the reference paper)
abline(h = 0.20011, lty = 3, lwd = 2)
text(-15, 0.21, "Upper limit: d", pos = 4, cex = 1.5)

segments(0,0.1,196,0.1, lty = 3, lwd = 2)
segments(196,0.1, 196, -0.1, lty = 3, lwd = 2)
text(200, -0.004, expression(paste("50% germination: ", t[50])), pos = 4, cex = 1.5)
```

```
abline(a = 0.20011/2-0.20011*20.77/4, b = 0.20011*20.77/4/196, lty = 3, lwd = 2)
#text(200, 0.1, expression(paste("Slope: ", b*(-d/(4*t[50])))), pos = 4, cex = 1.5)
text(200, 0.1, expression("Slope: b" %.% "constant"), pos = 4, cex = 1.5)
points(196, 0.1, cex = 2, pch = 0)


## Adding confidence intervals

## Predictions from the event-time model
#coefVec <- coef(chickweed.m1)
#names(coefVec) <- c("b","d","e")
#
#predFct <- function(tival)
#{
#    as.numeric(deltaMethod(coefVec, paste("d/(1+exp(b*(log(",tival,")-log(e))))"),
#    vcov(chickweed.m1)))
#}
#predFctv <- Vectorize(predFct, "tival")
#
#etpred <- t(predFctv(0:340))
#lines(0:340, etpred[,1]-1.96*etpred[,2], lty=1, lwd=2, col="darkgray")
#lines(0:340, etpred[,1]+1.96*etpred[,2], lty=1, lwd=2, col="darkgray")
#
### Predictions from the nonlinear regression model
#nrpred <- predict(chickweed.m0b, data.frame(time=0:340), interval="confidence")
#lines(0:340, nrpred[,2], lty=2, lwd=2, col="darkgray")
#lines(0:340, nrpred[,3], lty=2, lwd=2, col="darkgray")
```

CIcompX *Calculation of combination index for binary mixtures*

Description

For single mixture data combination indices for effective doses as well as effects may be calculated and visualized.

Usage

```
CIcomp(mixProp, modelList, EDvec)

CIcompX(mixProp, modelList, EDvec, EDonly = FALSE)

plotFACI(effList, indAxis = c("ED", "EF"), caRef = TRUE,
showPoints = FALSE, add = FALSE, ylim, ...)
```

Arguments

mixProp	a numeric value between 0 and 1 specifying the mixture proportion/ratio for the single mixture considered.
modelList	a list contained 3 models fits using drm with the model fit for single mixture ratio being the first element, followed by the 2 model fits of the pure substances.
EDvec	a vector of numeric values between 0 and 100 (percentages) coresponding to the effect levels of interest.
EDonly	a logical value indicating whether or not only combination indices for effective doses should be calculated.
effList	a list returned by CIcompX.
indAxis	a character indicating whether effective doses ("ED") or effects ("EF") should be plotted.
caRef	a logical value indicating whether or not a reference line for concentration addition should be drawn.
showPoints	A logical value indicating whether or not estimated combination indices should be plotted.
add	a logical value specifying if the plot should be added to the existing plot.
ylim	a numeric vector of length 2 giving the range for the y axis.
...	additional graphical arguments.

Details

CIcomp calculates the classical combination index for effective doses whereas CIcompX calculates the combination index also for effects as proposed by Martin-Betancor et al. (2015); for details and examples using "drc" see the supplementary material of this paper. The function plotFACI may be used to visualize the calculated combination index as a function of the fraction affected.

Value

CIcomp returns a matrix which one row per ED value. Columns contain estimated combination indices, their standard errors and 95% confidence intervals, p-value for testing CI=1, estimated ED values for the mixture data and assuming concentration addition (CA) with corresponding standard errors.

CIcompX returns similar output both for effective doses and effects (as a list of matrices).

Author(s)

Christian Ritz and Ismael Rodea-Palomares

References

Martin-Betancor, K. and Ritz, C. and Fernandez-Pinas, F. and Leganes, F. and Rodea-Palomares, I. (2015) Defining an additivity framework for mixture research in inducible whole-cell biosensors, *Scientific Reports* **17200**.

See Also

See `mixture` for simultaneous modelling of several mixture ratios, but only at the ED50 level.

See also the help page for `metals`.

Examples

```
## Fitting marginal models for the 2 pure substances
acidiq.0 <- drm(rgr ~ dose, data = subset(acidiq, pct == 999 | pct == 0), fct = LL.4())
acidiq.100 <- drm(rgr ~ dose, data = subset(acidiq, pct == 999 | pct == 100), fct = LL.4())

## Fitting model for single mixture with ratio 17:83
acidiq.17 <- drm(rgr ~ dose, data = subset(acidiq, pct == 17 | pct == 0), fct = LL.4())

## Calculation of combination indices based on ED10, ED20, ED50
CIcomp(0.17, list(acidiq.17, acidiq.0, acidiq.100), c(10, 20, 50))
## CI>1 significantly for ED10 and ED20, but not so for ED50
```

`coef.drc` *Extract Model Coefficients*

Description

Extract parameter estimates.

Usage

```
  ## S3 method for class 'drc'
coef(object, ...)
```

Arguments

`object`	an object of class 'drc'.
`...`	additional arguments.

Value

A vector of parameter coefficients which are extracted from the model object 'object'.

Note

This function may work even in cases where 'summary' does not, because the parameter estimates are retrieved directly from the model fit object without any additional computations of summary statistics and standard errors.

Author(s)

Christian Ritz

See Also

A more comprehensive summary is obtained using `summary.drc`.

Examples

```
## Fitting a four-parameter log-logistic model
ryegrass.m1 <- drm(rootl ~ conc, data = ryegrass, fct = LL.4())
coef(ryegrass.m1)
```

---

comped *Comparison of effective dose values*

Description

Comparison of a pair of effective dose values from independent experiments where only the estimates and their standard errors are reported.

Usage

```
comped(est, se, log = TRUE, interval = TRUE, operator = c("-", "/"),
level = 0.95, df = NULL)
```

Arguments

`est`	a numeric vector of length 2 containing the two estimated ED values
`se`	a numeric vector of length 2 containing the two standard errors
`log`	logical indicating whether or not estimates and standard errors are on log scale
`interval`	logical indicating whether or not a confidence interval should be returned
`operator`	character string taking one of the two values "-" (default) or "/" corresponding to a comparison based on the difference or the ratio.
`level`	numeric value giving the confidence level
`df`	numeric value specifying the degrees of freedom for the percentile used in the confidence interval (optional)

Details

The choice "/" for the argument `operator` and FALSE for `log` will result in estimation of a socalled relative potency (sometimes also called a selectivity index).

The combination TRUE for `log` and "/" for `operator` only influences the confidence interval, that is no ratio is calculated based on logarithm-transformed effective dose values.

By default confidence interval relies on percentiles in the normal distribution.

In case the entire dataset is available the functions `drm` and (subsequently) `EDcomp` should be used instead.

Value

A matrix with the estimated difference or ratio and the associated standard error and the resulting confidence interval (unless not requested).

Note

The development of the function `comped` is a side effect of the project on statistical analysis of toxicity data funded by the Danish EPA ("Statistisk analyse og biologisk tolkning af toksicitetsdata", MST j.nr. 669-00079).

Author(s)

Christian Ritz

References

Wheeler, M. W. and Park, R. M. and Bailer, A. J. (2006) Comparing median lethal concentration values using confidence interval overlap or ratio tests, *Environmental Toxicology and Chemistry*, **25**, 1441–1441.

See Also

The function `ED.drc` calculates arbitrary effective dose values based on a model fit. The function `EDcomp` calculates relative potencies based on arbitrary effective dose values.

Examples

```
## Fitting the model
S.alba.m1 <- boxcox(drm(DryMatter~Dose, Herbicide, data=S.alba, fct = LL.4(),
pmodels=data.frame(Herbicide,1,1,Herbicide)), method = "anova")

## Displaying estimated ED values
ED(S.alba.m1, c(10, 90))

## Making comparisons of ED50 in two ways and for both differences and ratios
compParm(S.alba.m1, "e", "/")

comped(c(28.396147, 65.573335), c(1.874598, 5.618945), log=FALSE, operator = "/")
```

```
# similar result

compParm(S.alba.m1, "e", "-")

comped(c(28.396147, 65.573335), c(1.874598, 5.618945), log=FALSE, operator = "-")
# similar result

## Making comparisons of ED10 and ED90
comped(c(21.173, 44.718), c(11.87, 8.42), log=FALSE, operator = "/")

comped(c(21.173, 44.718), c(11.87, 8.42), log=FALSE, operator = "/", interval = FALSE)

comped(c(21.173, 44.718), c(11.87, 8.42), log=FALSE, operator = "-")
```

`compParm` *Comparison of parameters*

Description

Compare parameters from different assays, either by means of ratios or differences.

Usage

```
compParm(object, strVal, operator = "/", vcov. = vcov, od = FALSE,
pool = TRUE, display = TRUE)
```

Arguments

`object`	an object of class 'drc'.
`strVal`	a name of parameter to compare.
`operator`	a character. If equal to "/" (default) parameter ratios are compared. If equal to "-" parameter differences are compared.
`vcov.`	function providing the variance-covariance matrix. `vcov` is the default, but `sandwich` is also an option (for obtaining robust standard errors).
`od`	logical. If TRUE adjustment for over-dispersion is used.
`pool`	logical. If TRUE curves are pooled. Otherwise they are not. This argument only works for models with independently fitted curves as specified in `drm`.
`display`	logical. If TRUE results are displayed. Otherwise they are not (useful in simulations).

Details

The function compares actual parameter estimates, and therefore the results depend on the parameterisation used. Probably it is most useful in combination with the argument `collapse` in `drm` for specifying parameter constraints in models, either through data frames or lists with formulas without intercept (-1).

Value

A matrix with columns containing the estimates, estimated standard errors, values of t-statistics and p-values for the null hypothesis that the ratio equals 1 or that the difference equals 0 (depending on the `operator` argument).

Author(s)

Christian Ritz

Examples

```
# Fitting a model with names assigned to the parameters!
spinach.m1 <- drm(SLOPE~DOSE, CURVE, data = spinach,
fct = LL.4(names = c("b", "lower", "upper", "ed50")))

## Calculating ratios of parameter estimates for the parameter named "ed50"
compParm(spinach.m1, "ed50")

## Calculating differences between parameter estimates for the parameter named "ed50"
compParm(spinach.m1, "ed50", "-")
```

`confint.drc` *Confidence Intervals for model parameters*

Description

Computes confidence intervals for one or more parameters in a model of class 'drc'.

Usage

```
  ## S3 method for class 'drc'
confint(object, parm, level = 0.95, pool = TRUE, ...)
```

Arguments

`object`	a model object of class 'drc'.
`parm`	a specification of which parameters are to be given confidence intervals, either a vector of numbers or a vector of names. If missing, all parameters are considered.
`level`	the confidence level required.
`pool`	logical. If TRUE curves are pooled. Otherwise they are not. This argument only works for models with independently fitted curves as specified in `drm`.
`...`	additional argument(s) for methods. Not used.

Details

For binomial and Poisson data the confidence intervals are based on the normal distribution, whereas *t* distributions are used of for continuous/quantitative data.

Value

A matrix (or vector) with columns giving lower and upper confidence limits for each parameter. These will be labelled as (1-level)/2 and 1 - (1-level)/2 in

Author(s)

Christian Ritz

Examples

```
## Fitting a four-parameter log-logistic model
ryegrass.m1 <- drm(rootl ~ conc, data = ryegrass, fct = LL.4())

## Confidence intervals for all parameters
confint(ryegrass.m1)

## Confidence interval for a single parameter
confint(ryegrass.m1, "e")
```

CRS.4a *The Cedergreen-Ritz-Streibig model*

Description

'CRS.4a', 'CRS.4b' and 'CRS.4c' provide the Cedergreen-Ritz-Streibig modified log-logistic model for describing hormesis with the lower limit equal to 0.

'UCRS.4a', 'UCRS.4b' and 'UCRS.4c' provide the Cedergreen-Ritz-Streibig modified log-logistic model for describing u-shaped hormesis with the lower limit equal to 0.

Usage

```
CRS.4a(names = c("b", "d", "e", "f"), ...)

UCRS.4a(names = c("b", "d", "e", "f"), ...)
```

Arguments

names	a vector of character strings giving the names of the parameters. The default is reasonable (see above).
...	additional arguments to be passed from the convenience functions.

Details

The model is given by the expression

$$f(x) = 0 + \frac{d - 0 + f\exp(-1/x)}{1 + \exp(b(\log(x) - \log(e)))}$$

which is a five-parameter model.

It is a modification of the four-parameter logistic curve to take hormesis into account.

The u-shaped model is given by the expression

$$f(x) = 0 + d - \frac{d - 0 + f\exp(-1/x^{\alpha})}{1 + \exp(b(\log(x) - \log(e)))}$$

The a,b,c models are obtained by setting alpha equal to 1, 0.5 and 0.25, respectively.

Value

See cedergreen.

Note

This function is for use with the function drm.

Author(s)

Christian Ritz

References

See the reference under cedergreen.

See Also

Similar functions are CRS.5a and UCRS.5a, but with an extra parameter for the lower limit.

Examples

```
## Fitting modified logistic models
lettuce.crsm1 <- drm(lettuce[,c(2,1)], fct=CRS.4a())
summary(lettuce.crsm1)
ED(lettuce.crsm1, c(50))

## Need to explicitly specify that the upper limit
## is the reference in order to get ED10 and ED90 right
ED(lettuce.crsm1, c(10, 50, 90), reference = "upper")

lettuce.crsm2 <- drm(lettuce[,c(2,1)], fct=CRS.4b())
summary(lettuce.crsm2)
ED(lettuce.crsm2, c(50))
```

```
lettuce.crsm3 <- drm(lettuce[,c(2,1)], fct=CRS.4c())
summary(lettuce.crsm3)
ED(lettuce.crsm3, c(50))
```

CRS.5a *Cedergreen-Ritz-Streibig dose-reponse model for describing hormesis*

Description

'CRS.5a', 'CRS.5b' and 'CRS.5c' provide the Cedergreen-Ritz-Streibig modified log-logistic model for describing (inverse u-shaped or j-shaped) hormesis.

'UCRS.5a', 'UCRS.5b' and 'UCRS.5c' provide the Cedergreen-Ritz-Streibig modified log-logistic model for describing u-shaped hormesis.

Usage

```
CRS.5a(names = c("b", "c", "d", "e", "f"), ...)

UCRS.5a(names = c("b", "c", "d", "e", "f"), ...)
```

Arguments

names	a vector of character strings giving the names of the parameters.
...	additional arguments to be passed from the convenience functions.

Details

The model function for inverse u-shaped hormetic patterns is

$$f(x) = c + \frac{d - c + f\exp(-1/x^{\alpha})}{1 + \exp(b(\log(x) - \log(e)))}$$

,

which is a five-parameter model. It is a modification of the four-parameter log-logistic curve to take hormesis into account.

The parameters have the following interpretations

- b: Not direct interpretation
- c: Lower horizontal asymptote
- d: Upper horizontal asymptote
- e: Not direct interpretation
- f: Size of the hormesis effect: the larger the value the larger is the hormesis effect. $f = 0$ corresponds to no hormesis effect and the resulting model is the four-parameter log-logistic model. This parameter should be positive in order for the model to make sense.

The model function for u-shaped hormetic patterns is

$$f(x) = c + d - \frac{d - c + f\exp(-1/x^{\alpha})}{1 + \exp(b(\log(x) - \log(e)))}$$

This model also simplifies to the four-parameter log-logistic model in case $f = 0$ (in a slightly different parameterization as compared to the one used in LL.4).

The models denoted a,b,c are obtained by fixing the alpha parameter at 1, 0.5 and 0.25, respectively.

Value

See cedergreen.

Note

This function is for use with the function drm.

Author(s)

Christian Ritz

References

See the reference under cedergreen.

See Also

Similar functions are CRS.4a and UCRS.4a, but with the lower limit (the parameter c) fixed at 0 (one parameter less to be estimated).

Examples

```
## Modified logistic model
lettuce.m1 <- drm(lettuce[,c(2,1)], fct=CRS.5a())
summary(lettuce.m1)
ED(lettuce.m1, c(50))

lettuce.m2 <- drm(lettuce[,c(2,1)], fct=CRS.5b())
summary(lettuce.m2)
ED(lettuce.m2, c(50))

lettuce.m3 <- drm(lettuce[,c(2,1)], fct=CRS.5c())
summary(lettuce.m3)
ED(lettuce.m3, c(50))
```

daphnids *Daphnia test*

Description

The number of immobile daphnids –in contrast to mobile daphnids– out of a total of 20 daphnids was counted for several concentrations of a toxic substance.

Usage

```
data(daphnids)
```

Format

A data frame with 16 observations on the following 4 variables.

`dose` a numeric vector

`no` a numeric vector

`total` a numeric vector

`time` a factor with levels `24h` `48h`

Details

The same daphnids were counted at 24h and later again at 48h.

Source

Nina Cedergreen, Faculty of Life Sciences, University of Copenhagen, Denmark.

Examples

```
## Fitting a model with different parameters
## for different curves
daphnids.m1 <- drm(no/total~dose, time, weights = total,
data = daphnids, fct = LL.2(), type = "binomial")

## Goodness-of-fit test
modelFit(daphnids.m1)

## Summary of the data
summary(daphnids.m1)

## Fitting a model with a common intercept parameter
daphnids.m2 <- drm(no/total~dose, time, weights = total,
data = daphnids, fct = LL.2(), type = "binomial",
pmodels = list(~1, ~time))
```

`decontaminants` *Performance of decontaminants used in the culturing of a micro-organism*

Description

The two decontaminants 1-hexadecylpyridium chloride and oxalic acid were used. Additionally there was a control group (coded as concentration 0 and only included under oxalic acid).

Usage

```
data("decontaminants")
```

Format

A data frame with 128 observations on the following 3 variables.

`conc` a numeric vector of percentage weight per volume

`count` a numeric vector of numbers of M. bovis colonies at stationarity

`group` a factor with levels `hpc` and `oxalic` of the decontaminants used

Details

These data examplify Wadley's problem: counts where the maximum number is not known. The data were analyzed by Trajstman (1989) using a three-parameter logistic model and then re-analyzed by Morgan and Smith (1992) using a three-parameter Weibull type II model. In both cases the authors adjusted for overdispersion (in different ways).

It seems that Morgan and Smith (1992) fitted separate models for the two decontaminants and using the control group for both model fits. In the example below a joint model is fitted where the control group is used once to determine a shared upper limit at concentration 0.

Source

Trajstman, A. C. (1989) Indices for Comparing Decontaminants when Data Come from Dose-Response Survival and Contamination Experiments, *Applied Statistics*, **38**, 481–494.

References

Morgan, B. J. T. and Smith, D. M. (1992) A Note on Wadley's Problem with Overdispersion, *Applied Statistics*, **41**, 349–354.

Examples

```
## Wadley's problem using a three-parameter log-logistic model
decon.LL.3.1 <- drm(count~conc, group, data = decontaminants, fct = LL.3(),
type = "Poisson", pmodels = list(~group, ~1, ~group))

summary(decon.LL.3.1)

plot(decon.LL.3.1)


## Same model fit in another parameterization (no intercepts)
decon.LL.3.2 <- drm(count~conc, group, data = decontaminants, fct=LL.3(),
type = "Poisson", pmodels = list(~group-1, ~1, ~group-1))

summary(decon.LL.3.2)
```

`deguelin` *Deguelin applied to chrysanthemum aphis*

Description

Quantal assay data from an experiment where the insectide deguelin was applied to *Macrosiphoniella sanborni*.

Usage

```
data(deguelin)
```

Format

A data frame with 6 observations on the following 4 variables.

`dose` a numeric vector of doses applied

`log10dose` a numeric vector of logarithm-transformed doses

`r` a numeric vector contained number of dead insects

`n` a numeric vector contained the total number of insects

Details

The log-logistic model provides an inadequate fit.

The dataset is used in Nottingham and Birch (2000) to illustrate a semiparametric approach to dose-response modelling.

Source

Morgan, B. J. T. (1992) *Analysis of Quantal Response Data*, London: Chapman \& Hall/CRC (Table 3.9, p. 117).

References

Notttingham, Q. J. and Birch, J. B. (2000) A semiparametric approach to analysing dose-response data, *Statist. Med.*, **19**, 389–404.

Examples

```
## Log-logistic fit
deguelin.m1 <- drm(r/n~dose, weights=n, data=deguelin, fct=LL.2(), type="binomial")
modelFit(deguelin.m1)
summary(deguelin.m1)

## Loess fit
deguelin.m2 <- loess(r/n~dose, data=deguelin, degree=1)

## Plot of data with fits superimposed
plot(deguelin.m1, ylim=c(0.2,1))
lines(1:60, predict(deguelin.m2, newdata=data.frame(dose=1:60)), col = 2, lty = 2)

lines(1:60, 0.95*predict(deguelin.m2,
newdata=data.frame(dose=1:60))+0.05*predict(deguelin.m1, newdata=data.frame(dose=1:60), se = FALSE),
col = 3, lty=3)
```

`drm` *Fitting dose-response models*

Description

A general model fitting function for analysis of concentration/dose/time-effect/response data.

Usage

```
drm(formula, curveid, pmodels, weights, data = NULL, subset, fct,
type = c("continuous", "binomial", "Poisson", "quantal", "event"),
bcVal = NULL, bcAdd = 0,
start, na.action = na.omit, robust = "mean", logDose = NULL,
control = drmc(), lowerl = NULL, upperl = NULL, separate = FALSE, pshifts = NULL)
```

Arguments

formula	a symbolic description of the model to be fit. Either of the form 'response dose' or as a data frame with response values in first column and dose values in second column.
curveid	a numeric vector or factor containing the grouping of the data.
pmodels	a data frame with a many columns as there are parameters in the non-linear function. Or a list containing a formula for each parameter in the nonlinear function.
weights	a numeric vector containing weights. For continuous/quantitative responses weights are multiplied inside the squared errors (see the details below). For binomial reponses weights provide information about the total number of binary observations used to obtain the response (which is a proportion): 1/2 and 10/20 lead to different analyses due to the different totals (2 vs. 20) even though the proportion in both cases is 0.5.
data	an optional data frame containing the variables in the model.
subset	an optional vector specifying a subset of observations to be used in the fitting process.
fct	a list with three or more elements specifying the non-linear function, the accompanying self starter function, the names of the parameter in the non-linear function and, optionally, the first and second derivatives as well as information used for calculation of ED values. Currently available functions include, among others, the four- and five-parameter log-logistic models LL.4, LL.5 and the Weibull model W1.4. Use getMeanFunctions for a full list.
type	a character string specifying the data type (parameter estimation will depend on the data type as different log likelihood function will be used).
bcVal	a numeric value specifying the lambda parameter to be used in the Box-Cox transformation.
bcAdd	a numeric value specifying the constant to be added on both sides prior to Box-Cox transformation. The default is 0.
start	an optional numeric vector containing starting values for all mean parameters in the model. Overrules any self starter function.
na.action	a function for treating mising values ('NA's). Default is na.omit.
robust	a character string specifying the rho function for robust estimation. Default is non-robust least squares estimation ("mean"). Available robust methods are: median estimation ("median"), least median of squares ("lms"), least trimmed squares ("lts"), metric trimming ("trimmed"), metric winsorizing ("winsor") and Tukey's biweight ("tukey").
logDose	a numeric value or NULL. If log doses value are provided the base of the logarithm should be specified (exp(1) for the natural logarithm and 10 for 10-logarithm).
control	a list of arguments controlling constrained optimisation (zero as boundary), maximum number of iteration in the optimisation, relative tolerance in the optimisation, warnings issued during the optimisation.

lowerl	a numeric vector of lower limits for all parameters in the model (the default corresponds to minus infinity for all parameters).
upperl	a numeric vector of upper limits for all parameters in the model (the default corresponds to plus infinity for all parameters).
separate	logical value indicating whether curves should be fit separately (independent of each other).
pshifts	a matrix of constants to be added to the matrix of parameters. Default is no shift for all parameters.

Details

This function relies on the general optimiser function optim for the minimisation of negative log likelihood function. For a continuous response this reduces to least squares estimation, which is carried out by minimising the following sums of squares

$$\sum_{i=1}^{N}[w_i(y_i - f_i)]^2$$

where y_i, f_i, and w_i correspond to the observed value, expected value, and the weight respectively, for the ith observation (from 1 to N).

The control arguments are specified using the function drmc.

Setting lowerl and/or upperl automatically invokes constrained optimisation.

The columns of a data frame argument to pmodels are automatically converted into factors. This does not happen if a list is specified.

Value

An object of class 'drc'.

Note

For robust estimation MAD (median abslolute deviance) is used to estimate the residual variance.

Author(s)

Christian Ritz and Jens C. Streibig

See Also

Examples using drm found in the help pages of ryegrass (continuous data), secalonic (continuous data), and selenium (binomial data), as well as for a number of other datasets and functions in drc.

drmc *Sets control arguments*

Description

Set control arguments in the control argument in the function 'drm'.

Usage

```
drmc(constr = FALSE, errorm = TRUE, maxIt = 500, method="BFGS",
noMessage = FALSE, relTol = 1e-07, rmNA=FALSE, useD = FALSE,
trace = FALSE, otrace = FALSE, warnVal = -1, dscaleThres = 1e-15, rscaleThres = 1e-15)
```

Arguments

constr	logical. If TRUE optimisation is constrained, only yielding non-negative parameters.
errorm	logical specifying whether failed convergence in drm should result in an error or only a warning.
maxIt	numeric. The maximum number of iterations in the optimisation procedure.
method	character string. The method used in the optimisation procedure. See optim for available methods.
noMessage	logical, specifying whether or not messages should be displayed.
relTol	numeric. The relative tolerance in the optimisation procedure.
rmNA	logical. Should NAs be removed from sum of squares used for estimation? Default is FALSE (not removed).
useD	logical. If TRUE derivatives are used for estimation (if available).
trace	logical. If TRUE the trace from optim is displayed.
otrace	logical. If TRUE the output from optim is displayed.
warnVal	numeric. If equal to 0 then the warnings are stored and displayed at the end. See under 'warn' in options. The default results in suppression of warnings.
dscaleThres	numeric value specifying the threshold for dose scaling.
rscaleThres	numeric value specifying the threshold for response scaling.

Value

A list with 8 components, one for each of the above arguments.

Note

The use of a non-zero constant bcAdd may in some cases make it more difficult to obtain convergence of the estimation procedure.

Author(s)

Christian Ritz

Examples

```
### Displaying the default settings
drmc()

### Using 'method' argument
model1 <- drm(ryegrass, fct = LL.4())

model2 <- drm(ryegrass, fct = LL.4(),
control = drmc(method = "Nelder-Mead"))
```

`earthworms` *Earthworm toxicity test*

Description

The dataset was obtained from a toxicity test using earthworms, and it contains the number of earthworms remaining in a container that was contaminated with a toxic substance (not disclosed) at various doses; so the number of earthworms not migrating to the neighbouring uncontaminated container.

Usage

```
data(earthworms)
```

Format

A data frame with 35 observations on the following 3 variables.

`dose` a numeric vector of dose values

`number` a numeric vector containing counts of remaining earthworms in the container

`total` a numeric vector containing total number of earthworms put in the containers

Details

At dose 0 around half of the earthworms is expected be in each of the two containers. Thus it is not appropriate to fit an ordinary logistic regression with log(dose) as explanatory variable to these data as it implies an upper limit of 1 at dose 0 and in fact this model does not utilise the observations at dose 0 (see the example section below).

Source

The dataset is kindly provided by Nina Cedergreen, Faculty of Life Sciences, University of Copenhagen, Denmark.

Examples

```
## Fitting a logistic regression model
earthworms.m1 <- drm(number/total~dose, weights = total, data = earthworms,
fct = LL.2(), type = "binomial")
modelFit(earthworms.m1)  # a crude goodness-of-fit test

## Fitting an extended logistic regression model
##  where the upper limit is estimated
earthworms.m2 <- drm(number/total~dose, weights = total, data = earthworms,
fct = LL.3(), type = "binomial")
modelFit(earthworms.m2)  # goodness-of-fit test
# improvement not visible in test!!!

## Comparing model1 and model2
## (Can the first model be reduced to the second model?)
anova(earthworms.m1, earthworms.m2)
```

`ED.drc` *Estimating effective doses*

Description

`ED` estimates effective doses (ECp/EDp/ICp) for given reponse levels.

Usage

```
  ## S3 method for class 'drc'
ED(object, respLev, interval = c("none", "delta", "fls", "tfls"),
  clevel = NULL, level = ifelse(!(interval == "none"), 0.95, NULL),
  reference = c("control", "upper"), type = c("relative", "absolute"), lref, uref,
  bound = TRUE, od = FALSE, vcov. = vcov, display = TRUE, pool = TRUE, logBase = NULL,
  multcomp = FALSE, ...)
```

Arguments

`object`	an object of class 'drc'.
`respLev`	a numeric vector containing the response levels.
`interval`	character string specifying the type of confidence intervals to be supplied. The default is "none". Use "delta" for asymptotics-based confidence intervals (using the delta method and the t-distribution). Use "fls" for from logarithm scale based confidence intervals (in case the parameter in the model is log(ED50) as for the `llogistic2`) models. The only alternative for model-robust fits is using inverse regression.
`clevel`	character string specifying the curve id in case on estimates for a specific curve or compound is requested. By default estimates are shown for all curves.

level	numeric. The level for the confidence intervals. The default is 0.95.
reference	character string. Is the upper limit or the control level the reference?
type	character string. Whether the specified response levels are absolute or relative (default).
lref	numeric value specifying the lower limit to serve as reference.
uref	numeric value specifying the upper limit to serve as reference (e.g., 100%).
bound	logical. If TRUE only ED values between 0 and 100% are allowed. FALSE is useful for hormesis models.
od	logical. If TRUE adjustment for over-dispersion is used.
vcov.	function providing the variance-covariance matrix. vcov is the default, but sandwich is also an option (for obtaining robust standard errors).
display	logical. If TRUE results are displayed. Otherwise they are not (useful in simulations).
pool	logical. If TRUE curves are pooled. Otherwise they are not. This argument only works for models with independently fitted curves as specified in drm.
logBase	numeric. The base of the logarithm in case logarithm transformed dose values are used.
multcomp	logical to switch on output for use with the package multcomp (which needs to be activated first). Default is FALSE (corresponding to the original output).
...	see the details section below.

Details

For hormesis models (braincousens and cedergreen), the additional arguments lower and upper may be supplied. These arguments specify the lower and upper limits of the bisection method used to find the ED values. The lower and upper limits need to be smaller/larger than the EDx level to be calculated. The default limits are 0.001 and 1000 for braincousens and 0.0001 and 10000 for cedergreen and ucedergreen, but this may need to be modified (for cedergreen the upper limit may need to be increased and for ucedergreen the lower limit may need to be increased). Note that the lower limit should not be set to 0 (use instead something like 1e-3, 1e-6, ...).

Value

An invisible matrix containing the shown matrix with two or more columns, containing the estimates and the corresponding estimated standard errors and possibly lower and upper confidence limits. Or, alternatively, a list with elements that may be plugged directly into parm in the package *multcomp* (in case the argument multcomp is TRUE).

Author(s)

Christian Ritz

See Also

backfit, isobole, and maED use ED for specific calculations involving estimated ED values.

The related function EDcomp may be used for estimating differences and ratios of ED values, whereas compParm may be used to compare other model parameters.

Examples

```
## Fitting 4-parameter log-logistic model
ryegrass.m1 <- drm(ryegrass, fct = LL.4())

## Calculating EC/ED values
ED(ryegrass.m1, c(10, 50, 90))
## first column: the estimates of ED10, ED50 and ED90
## second column: the corresponding estimated standard errors

### How to use the argument 'ci'

## Also displaying 95% confidence intervals
ED(ryegrass.m1, c(10, 50, 90), interval = "delta")

## Comparing delta method and back-transformed
##  confidence intervals for ED values

## Fitting 4-parameter log-logistic
##  in different parameterisation (using LL2.4)
ryegrass.m2 <- drm(ryegrass, fct = LL2.4())

ED(ryegrass.m1, c(10, 50, 90), interval = "fls")
ED(ryegrass.m2, c(10, 50, 90), interval = "delta")


### How to use the argument 'bound'

## Fitting the Brain-Cousens model
lettuce.m1 <- drm(weight ~ conc,
data = lettuce, fct = BC.4())

### Calculating ED[-10]

# This does not work
#ED(lettuce.m1, -10)

## Now it does work
ED(lettuce.m1, -10, bound = FALSE)  # works
ED(lettuce.m1, -20, bound = FALSE)  # works

## The following does not work for another reason: ED[-30] does not exist
#ED(lettuce.m1, -30, bound = FALSE)
```

EDcomp *Comparison of relative potencies between dose-response curves*

Description

Relative potencies (also called selectivity indices) for arbitrary doses are compared between fitted dose-response curves.

Usage

```
EDcomp(object, percVec, percMat = NULL, compMatch = NULL, od = FALSE, vcov. = vcov,
reverse = FALSE,
interval = c("none", "delta", "fieller", "fls"),
level = ifelse(!(interval == "none"), 0.95, NULL),
reference = c("control", "upper"),
type = c("relative", "absolute"),
display = TRUE, pool = TRUE, logBase = NULL,
multcomp = FALSE, ...)

relpot(object, plotit = TRUE, compMatch = NULL, percVec = NULL, interval = "none",
type = c("relative", "absolute"),
scale = c("original", "percent", "unconstrained"), ...)
```

Arguments

object	an object of class 'drc'.
percVec	a numeric vector of dosage values.
percMat	a matrix with 2 columns providing the pairs of indices percVec to be compared. By default all pairs are compared.
compMatch	an optional character vector of names of assays to be compared. If not specified all comparisons are supplied.
od	logical. If TRUE adjustment for over-dispersion is used. This argument only makes a difference for binomial data.
vcov.	function providing the variance-covariance matrix. vcov is the default, but sandwich is also an option (for obtaining robust standard errors).
reverse	logical. If TRUE the order of comparison of two curves is reversed.
interval	character string specifying the type of confidence intervals to be supplied. The default is "none". Use "delta" for asymptotics-based confidence intervals (using the delta method and the t-distribution). Use "fieller" for confidence intervals based on Fieller's theorem (with help from the delta method). Use "fls" for confidence interval back-transformed from logarithm scale (in case the parameter in the model fit is log(ED50) as is the case for the logistic or llogistic2 models); currently the argument logBase then also needs to be specified.
level	numeric. The level for the confidence intervals. Default is 0.95.
reference	character string. Is the upper limit or the control level the reference?
type	character string specifying whether absolute or relative response levels are supplied.
logBase	numeric. The base of the logarithm in case logarithm transformed dose values are used.

display	logical. If TRUE results are displayed. Otherwise they are not (useful in simulations).
pool	logical. If TRUE curves are pooled. Otherwise they are not. This argument only works for models with independently fitted curves as specified in drm.
multcomp	logical to switch on output for use with the package multcomp (which needs to be activated first). Default is FALSE (corresponding to the original output).
...	In SI: additional arguments to the function doing the calculations. For instance the upper limit for the bisection method needs to be larger than the ED values used in the required relative pontency. In relpot: additional graphical parameters.
plotit	logical. If TRUE the relative potencies are plotted as a function of the response level.
scale	character string indicating the scale to be used on the x axis: original or percent response level (only having an effect for type="relative").

Details

The function relpot is a convenience function, which is useful for assessing how the relative potency changes as a function of the response level (e.g., for plotting as outlined by Ritz *et al* (2006)).

Fieller's theorem is incorporated using the formulas provided by Kotz and Johnson (1983) and Finney (1978).

For objects of class 'braincousens' or 'mlogistic' the additional argument may be the 'upper' argument or the 'interval' argument. The 'upper' argument specifies the upper limit of the bisection method. The upper limits needs to be larger than the EDx level to be calculated. The default limit is 1000. The 'interval' argument should specify a rough interval in which the dose yielding the maximum hormetical response lies. The default interval is 'c(0.001, 1000)'. Notice that the lower limit should not be set to 0 (use something like 1e-3, 1e-6, ...).

Value

An invisible matrix containing the shown matrix with two or more columns, containing the estimates and the corresponding estimated standard errors and possibly lower and upper confidence limits. Or, alternatively, a list with elements that may be plugged directly into parm in the package *multcomp* (in case the argument multcomp is TRUE).

Note

This function only works for the following built-in functions available in the package *drc*: braincousens, cedergreen, ucedergreen, llogistic, and weibull1.

Author(s)

Christian Ritz

References

Finney, D. J. (1978) *Statistical method in Biological Assay*, London: Charles Griffin House, 3rd edition (pp. 80–82).

Kotz, S. and Johnson, N. L. (1983) *Encyclopedia of Statistical Sciences Volume 3*, New York: Wiley \& Sons (pp. 86–87).

Ritz, C. and Cedergreen, N. and Jensen, J. E. and Streibig, J. C. (2006) Relative potency in nonsimilar dose-response curves, *Weed Science*, **54**, 407–412.

See Also

A related function is ED.drc (used for calculating effective doses).

Examples

```
spinach.LL.4 <- drm(SLOPE~DOSE, CURVE, data = spinach, fct = LL.4())

EDcomp(spinach.LL.4, c(50,50))
EDcomp(spinach.LL.4, c(10,50))
EDcomp(spinach.LL.4, c(10,50), reverse = TRUE)

## Using the package multcomp
#sires <- SI(spinach.LL.4, c(25, 50, 75))
#library(multcomp)
#summary(glht(parm(sires[[2]][[1]], sires[[2]][[2]]), rhs = 1))

## Comparing specific ratios: 25/25, 50/50, 75/75
#sires2 <- SI(spinach.LL.4, c(25, 50, 75), matrix(c(1, 1, 2, 2, 3, 3), 3, 2, byrow = TRUE))
#library(multcomp)
#summary(glht(parm(sires2[[2]][[1]], sires2[[2]][[2]]), rhs = 1))


## Relative potency of two herbicides
m2 <- drm(DryMatter~Dose, Herbicide,
data = S.alba, fct = LL.3())

EDcomp(m2, c(50, 50))
EDcomp(m2, c(50, 50), interval = "delta")
EDcomp(m2, c(50, 50), interval = "fieller")

## Comparison based on an absolute
##  response level

m3 <- drm(SLOPE~DOSE, CURVE,
data = spinach, fct = LL.4())

EDcomp(m3, c(0.5,0.5), compMatch = c(2,4), type = "absolute", interval = "fieller")

EDcomp(m3, c(55,80), compMatch = c(2,4))
# same comparison using a relative response level
```

```
## Relative potency transformed from log scale
m4 <- drm(drymatter~log(dose), treatment, data=G.aparine[-c(1:40), ],
pmodels = data.frame(treatment,treatment,1,treatment), fct = LL2.4())

EDcomp(m4, c(50,50), interval = "fls", logBase = exp(1))
```

`etmotc` *Effect of erythromycin on mixed sewage microorganisms*

Description

Relative growth rate in biomass of mixed sewage microorganisms (per hour) as a function of increasing concentrations of the antibiotic erythromycin (mg/l).

Usage

```
data(etmotc)
```

Format

A data frame with 57 observations on the following 4 variables.

`cell` a numeric vector

`dose1` a numeric vector

`pct1` a numeric vector

`rgr1` a numeric vector

Details

Data stem from an experiment investigating the effect of pharmaceuticals, that are used in human and veterinary medicine and that are being released into the aquatic environment through waste water or through manure used for fertilising agricultural land. The experiment constitutes a typical dose-response situation. The dose is concentration of the antibiotic erythromycin (mg/l), which is an antibiotic that can be used by persons or animals showing allergy to penicillin, and the measured response is the relative growth rate in biomass of mixed sewage microorganisms (per hour), measured as turbidity two hours after exposure by means of a spectrophotometer. The experiment was designed in such a way that eight replicates were assigned to the control (dose 0), but no replicates were assigned to the 7 non-zero doses. Further details are found in Christensen et al (2006).

Source

Christensen, A. M. and Ingerslev, F. and Baun, A. 2006 Ecotoxicity of mixtures of antibiotics used in aquacultures, *Environmental Toxicology and Chemistry*, **25**, 2208–2215.

Examples

```
etmotc.m1<-drm(rgr1~dose1, data=etmotc[1:15,], fct=LL.4())
plot(etmotc.m1)
modelFit(etmotc.m1)
summary(etmotc.m1)

etmotc.m2<-drm(rgr1~dose1, data=etmotc[1:15,], fct=W2.4())
plot(etmotc.m2, add = TRUE)
modelFit(etmotc.m2)
summary(etmotc.m2)

etmotc.m3<-drm(rgr1~dose1, data=etmotc[1:15,], fct=W2.3())
plot(etmotc.m3, add = TRUE)
modelFit(etmotc.m3)
summary(etmotc.m3)
```

EXD *Exponential decay model*

Description

Exponential decay model with or without a nonzero lower limit.

Usage

```
EXD.2(fixed = c(NA, NA), names = c("d", "e"), ...)

EXD.3(fixed = c(NA, NA, NA), names = c("c", "d", "e"), ...)
```

Arguments

fixed	numeric vector. Specifies which parameters are fixed and at what value they are fixed. NAs for parameter that are not fixed.
names	vector of character strings giving the names of the parameters (should not contain ":"). The default parameter names are: init, plateau, k.
...	additional arguments to be passed from the convenience functions.

Details

The exponential decay model is a three-parameter model with mean function:

$$f(x) = c + (d - c)(\exp(-x/e))$$

The parameter init is the upper limit (attained at $x = 0$), the parameter plateau is the lower limit reached for x going to infinity and the parameter $e > 0$ is determining the steepness of the decay. The curve is monotonously decreasing in x.

Value

A list of class drcMean, containing the mean function, the self starter function, the parameter names and other components such as derivatives and a function for calculating ED values.

Author(s)

Christian Ritz

References

Organisation for Economic Co-operation and Development (OECD) (2006) *Current approaches in the statistical analysis of ecotoxicity data: A guidance to application - annexes*, Paris: OECD (p. 80).

See Also

Similar models giving exponential increasing curves are AR.2 and AR.3.

Examples

```
## Fitting an exponential decay model
ryegrass.m1<-drm(rootl~conc, data=ryegrass, fct=EXD.3())

plot(ryegrass.m1)

summary(ryegrass.m1)
```

---

finney71 *Example from Finney (1971)*

Description

For each of six concentration of an insecticid the number of insects affected (out of the number of insects) was recorded.

Usage

```
data(finney71)
```

Format

A data frame with 6 observations on the following 3 variables.

`dose` a numeric vector

`total` a numeric vector

`affected` a numeric vector

Source

Finney, D. J. (1971) *Probit Analysis*, Cambridge: Cambridge University Press.

Examples

```
## Model with ED50 as a parameter
finney71.m1 <- drm(affected/total ~ dose, weights = total,
data = finney71, fct = LL.2(), type = "binomial")

summary(finney71.m1)
plot(finney71.m1, broken = TRUE, bp = 0.1, lwd = 2)

ED(finney71.m1, c(10, 20, 50), interval = "delta", reference = "control")

## Model fitted with 'glm'
#fitl.glm <- glm(cbind(affected, total-affected) ~ log(dose),
#family=binomial(link = logit), data=finney71[finney71$dose != 0, ])
#summary(fitl.glm)  # p-value almost agree for the b parameter
#
#xp <- dose.p(fitl.glm, p=c(0.50, 0.90, 0.95))  # from MASS
#xp.ci <- xp + attr(xp, "SE") %*% matrix(qnorm(1 - 0.05/2)*c(-1,1), nrow=1)
#zp.est <- exp(cbind(xp.ci[,1],xp,xp.ci[,2]))
#dimnames(zp.est)[[2]] <- c("zp.lcl","zp","zp.ucl")
#zp.est  # not far from above results with 'ED'

## Model with log(ED50) as a parameter
finney71.m2 <- drm(affected/total ~ dose, weights = total,
data = finney71, fct = LL2.2(), type = "binomial")

## Confidence intervals based on back-transformation
##  complete agreement with results based on 'glm'
ED(finney71.m2, c(10, 20, 50), interval = "fls", reference = "control")
```

`fitted.drc` *Extract fitted values from model*

Description

Extracts fitted values from an object of class 'drc'.

Usage

```
  ## S3 method for class 'drc'
fitted(object, ...)
```

Arguments

object	an object of class 'drc'.
...	additional arguments.

Value

Fitted values extracted from 'object'.

Author(s)

Christian Ritz

Examples

```
ryegrass.m1 <- drm(rootl ~ conc, data = ryegrass, fct = LL.4())
plot(fitted(ryegrass.m1), residuals(ryegrass.m1))  # a residual plot
```

---

fplogistic *Fractional polynomial-logistic dose-response models*

Description

Model function for specifying dose-response models that are a combination of a logistic model and an appropriate class of fractional polynomials.

Usage

```
fplogistic(p1, p2, fixed = c(NA, NA, NA, NA), names = c("b", "c", "d", "e"),
method = c("1", "2", "3", "4"), ssfct = NULL, fctName, fctText)

FPL.4(p1, p2, fixed = c(NA, NA, NA, NA), names = c("b", "c", "d", "e"), ...)
```

Arguments

p1	numeric denoting the negative power of log(dose+1) in the fractional polynomial.
p2	numeric denoting the positive power of log(dose+1) in the fractional polynomial.
fixed	numeric vector. Specifies which parameters are fixed and at what value they are fixed. NAs for parameter that are not fixed.
names	a vector of character strings giving the names of the parameters (should not contain ":"). The default is reasonable (see under 'Usage'). The order of the parameters is: b, c, d, e, f (see under 'Details').
method	character string indicating the self starter function to use.

ssfct	a self starter function to be used.
fctName	optional character string used internally by convenience functions.
fctText	optional character string used internally by convenience functions.
...	Additional arguments (see fplogistic).

Details

The fractional polynomial dose-response models introduced by Namata *et al.* (2008) are implemented using the logistic model as base.

Value

The value returned is a list containing the nonlinear function, the self starter function and the parameter names.

Author(s)

Christian Ritz

References

Namata, Harriet and Aerts, Marc and Faes, Christel and Teunis, Peter (2008) Model Averaging in Microbial Risk Assessment Using Fractional Polynomials, *Risk Analysis* **28**, 891–905.

See Also

Examples are found maED.

G.aparine *Herbicide applied to Galium aparine*

Description

Small plants of *Galium aparine*, growing in pots in a green house, were sprayed with the technical grade phenmidipham herbicide either alone or in mixture with an ester of oleic acid. The plants were allowed to grow in the green house for 14 days after herbicide treatment. Then the dry matter was measured per pot.

Usage

```
data(G.aparine)
```

Format

A data frame with 240 observations on the following 3 variables.

`dose` a numeric vector of dose value (g/ha)

`drymatter` a numeric vector of dry matter weights (mg/pot)

`treatment` a numeric vector giving the grouping: 0: control, 1,2: herbicide formulations

Source

Cabanne, F., Gaudry, J. C. and Streibig, J. C. (1999) Influence of alkyl oleates on efficacy of phenmedipham applied as an acetone:water solution on Galium aparine, *Weed Research*, **39**, 57–67.

Examples

```
## Fitting a model with a common control (so a single upper limit: "1")
G.aparine.m1 <- drm(drymatter ~ dose, treatment, data = G.aparine,
pmodels = data.frame(treatment, treatment, 1, treatment), fct = LL.4())

## Visual inspection of fit
plot(G.aparine.m1, broken = TRUE)

## Lack of fit test
modelFit(G.aparine.m1)

## Summary output
summary(G.aparine.m1)

## Predicted values with se and confidence intervals
#predict(G.aparine.m1, interval = "confidence")
# long output

## Calculating the relative potency
EDcomp(G.aparine.m1, c(50,50))

## Showing the relative potency as a
## function of the response level
relpot(G.aparine.m1)
relpot(G.aparine.m1, interval = "delta")
# appears constant!

## Response level in percent
relpot(G.aparine.m1, scale = "percent")

## Fitting a reduced model (with a common slope parameter)
G.aparine.m2 <- drm(drymatter ~ dose, treatment, data = G.aparine,
pmodels = data.frame(1, treatment, 1, treatment), fct = LL.4())

anova(G.aparine.m2, G.aparine.m1)

## Showing the relative potency
relpot(G.aparine.m2)

## Fitting the same model in a different parameterisation
G.aparine.m3 <- drm(drymatter ~ dose, treatment, data = G.aparine,
pmodels = data.frame(treatment, treatment, 1, treatment), fct = LL2.4())

EDcomp(G.aparine.m3, c(50, 50), logBase = exp(1))
```

gammadr *Gamma dose-response model*

Description

The gamma dose-response model is a four-parameter model derived from the cumulative distribution function of the gamma distribution.

Usage

```
gammadr(fixed = c(NA, NA, NA, NA),
names = c("b", "c", "d", "e"), fctName, fctText)
```

Arguments

fixed	numeric vector specifying which parameters are fixed and at what value they are fixed. NAs are used for parameters that are not fixed.
names	a vector of character strings giving the names of the parameters (should not contain ":"). The default is reasonable (see under 'Usage').
fctName	optional character string used internally by convenience functions.
fctText	optional character string used internally by convenience functions.

Details

Following Wheeler and Bailer (2009) the model function is defined as follows:

$$f(x) = c + (d - c) * pgamma(b * x, e, 1)$$

This model is only suitable for increasing dose-response data.

Value

The value returned is a list containing the nonlinear function, the self starter function and the parameter names.

Author(s)

Christian Ritz

References

Wheeler, M. W., Bailer, A. J. (2009) Comparing model averaging with other model selection strategies for benchmark dose estimation, *Environmental and Ecological Statistics*, **16**, 37–51.

gaussian *Normal and log-normal biphasic dose-response models*

Description

Model functions for fitting symmetric or skewed bell-shaped/biphasic dose-response patterns.

Usage

```
gaussian(fixed = c(NA, NA, NA, NA, NA), names = c("b", "c", "d", "e", "f"),
method = c("1", "2", "3", "4"), ssfct = NULL, fctName, fctText, loge = FALSE)

lgaussian(fixed = c(NA, NA, NA, NA, NA), names = c("b", "c",
"d", "e", "f"), method = c("1", "2", "3", "4"), ssfct = NULL,
fctName, fctText, loge = FALSE)
```

Arguments

fixed	numeric vector. Specifies which parameters are fixed and at what value they are fixed. NAs for parameter that are not fixed.
names	a vector of character strings giving the names of the parameters (should not contain ":"). The default is reasonable (see under 'Usage'). The order of the parameters is: b, c, d, e, f (see under 'Details').
method	character string indicating the self starter function to use.
ssfct	a self starter function to be used.
fctName	optional character string used internally by convenience functions.
fctText	optional character string used internally by convenience functions.
loge	logical indicating whether or not e or log(e) should be a parameter in the model. By default e is a model parameter.

Details

Details yet to be provided.

Value

The value returned is a list containing the nonlinear function, the self starter function and the parameter names.

Note

The functions are for use with the function drm.

Author(s)

Christian Ritz

`germination` *Germination of three crops*

Description

Germination data were obtained from experiments involving the three species mungbean, rice, and wheat, which were opposed to different temperatures between 10 and 40 degrees Celsius. Experiments lasted at most 18 days.

Usage

```
data(germination)
```

Format

A data frame with 192 observations on the following 5 variables.

`temp` a numeric vector of temperatures that seeds were exposed to

`species` a factor with levels `mungbean` `rice` `wheat`

`start` a numeric vector of left endpoints of the monitoring intervals

`end` a numeric vector of right endpoints of the monitoring intervals

`germinated` a numeric vector giving the numbers of seeds germinated

Details

For each of the three species mungbean, rice, and wheat, a total of 20 seeds were uniformly distributed on filter paper in a petri dish (diameter: 9.0cm) and then placed in dark climate cabinets with different temperatures (10, 16, 22, 28, 34, 40 degrees Celsius). Not all of the temperatures were applied to all species. The germinated seeds were counted and removed from the petri dish on a daily basis up to 18 days (or until all seeds had germinated). I

n this experiment we also assume that the upper limit of the proportion germinated is a parameter that has to be estimated from the data. Moreover, we assume that different combinations of species and temperature may lead to different germination curves with respect to slope, time required for 50% germination, and upper limit.

References

Ritz, C., Pipper, C. B. and Streibig, J. C. (2013) Analysis of germination data from agricultural experiments, *Europ. J. Agronomy*, **45**, 1–6.

See Also

Analysis of a single germination curve is shown for `chickweed`.

Examples

```
## Fitting two-parameter log-logistic curves to each combination of species and temperature
##  (upper limit fixed at 1)
## Note: Rows 24 and 62 are omitted from the dataset (all mungbean seeds germinated
##  and thus no right-censoring in this case)

## germLL.2 <- drm(germinated ~ start + end, species:factor(temp),
## data = germination[c(1:23, 25:61, 63:192), ], fct = LL.2(), type = "event")
## plot(germLL.2, ylim=c(0, 1.5), legendPos=c(2.5,1.5))  # plotting the fitted curves and the data
## summary(germLL.2)  # showing the parameter estimates

## Fitting two-parameter log-logistic curves to each combination of species and temperature
## Note: the argument "start" may be used for providing sensible initial
##  parameter values for estimation procedure (is needed occasionally)
##  (initial values were obtained from the model fit germLL.2)
## Note also: the argument "upper" ensures that the upper limit cannot exceed 1
## (however, no restrictions are imposed on the two remaining parameters
## (as indicated by an infinite value)

## germLL.3 <- drm(germinated~start+end, species:factor(temp),
## data = germination[c(1:23, 25:61, 63:192), ], fct = LL.3(), type = "event",
## start = c(coef(germLL.2)[1:13], rep(0.7,13), coef(germLL.2)[14:26]),
## upper = c(rep(Inf, 13), rep(1, 13), rep(Inf, 13)))

## Plotting the fitted curves and the data
## plot(germLL.3, ylim = c(0, 1.5), legendPos = c(2.5,1.5))

## Showing the parameter estimates
## summary(germLL.3)

## Showing the parameter estimates with robust standard errors
## library(lmtest)
## coeftest(germLL.3, vcov = sandwich)

## Calculating t50 with associated standard errors
## ED(germLL.3, 50)

## Calculating t10, t20, t50 with 95% confidence intervals
## ED(germLL.3, c(10, 20, 50), interval = "delta")

## Comparing t50 between combinations by means of approximate t-tests
## compParm(germLL.3, "e", "-")

## Making plots of fitted regression curves for each species

## Plot for mungbean
#plot(germLL.3, log="", ylim=c(0, 1), xlim=c(0, 20),
#level=c("mungbean:10", "mungbean:16"),
#lty=2:3, lwd = 1.5,
#xlab="Time (days)",
#ylab="Proportion germinated",
```

```
#main="Mungbean",
#legendPos=c(3, 1.05), legendText=c(expression(10*degree), expression(16*degree)))

## Plot for rice
#plot(germLL.3, log="", ylim=c(0, 1), xlim=c(0, 20),
#level=c("rice:16", "rice:22", "rice:28", "rice:34", "rice:40"),
#lty=2:6, lwd = 1.5,
#xlab="Time (days)",
#ylab="Proportion germinated",
#main="Rice",
#pch=2:6,
#legendPos=c(3, 1.05), legendText=c(expression(16*degree), expression(22*degree),
#expression(28*degree), expression(34*degree), expression(40*degree)))

## Plot for wheat
#plot(germLL.3, log="", ylim=c(0, 1), xlim=c(0, 20),
#level=c("wheat:10", "wheat:16", "wheat:22", "wheat:28", "wheat:34", "wheat:40"),
#lty=c("dashed","dotted","dotdash","longdash","twodash","232A"), lwd = 1.5,
#xlab="Time (days)",
#ylab="Proportion germinated",
#main="Wheat",
#legendPos=c(3, 1.05),
#legendText=c(expression(10*degree), expression(16*degree), expression(22*degree),
#expression(28*degree), expression(34*degree), expression(40*degree)))
```

`getInitial` *Showing starting values used*

Description

Function for showing the starting values of the model parameters used when fitting a dose-response model

Usage

```
getInitial(object)
```

Arguments

`object`	object of class 'drc'

Value

A vector of starting values for the model parameters used to initialize the estimation procedure.

Note

This function is masking the standard function in the stats package.

Author(s)

Christian Ritz

`getMeanFunctions` *Display available dose-response models*

Description

Display information about available, built-in dose-response models.

Usage

```
getMeanFunctions(noParm = NA, fname = NULL, flist = NULL, display =TRUE)
```

Arguments

`noParm`	numeric specifying the number of parameters of the models to be displayed. The default (NA) results in display of all models, regardless of number of parameters.
`fname`	character string or vector of character strings specifying the short name(s) of the models to be displayed (need to match exactly).
`flist`	list of built-in functions to be displayed.
`display`	logical indicating whether or not the requested models should be displayed on the R console.

Details

The arguments `noParm` and `fname` can be combined.

Value

An invisible list of functions or a list of strings with brief function descriptions is returned.

Author(s)

Christian Ritz

Examples

```
## Listing all functions
getMeanFunctions()

## Listing all functions with 4 parameters
getMeanFunctions(4)

## Listing all (log-)logistic functions
getMeanFunctions(fname = "L")
```

```
## Listing all three-parameter (log-)logistic or Weibull functions
getMeanFunctions(3, fname = c("LL", "W"))

## Listing all four-parameter (log-)logistic or Weibull functions
getMeanFunctions(4, fname = c("LL", "W"))
```

---

glymet *Glyphosate and metsulfuron-methyl tested on algae.*

Description

The dataset has 7 mixtures, 8 dilutions, two replicates and 5 common control controls. Four observations are missing, giving a total of 113 observations.

Usage

```
data(glymet)
```

Format

A data frame with 113 observations on the following 3 variables.

`dose` a numeric vector of dose values

`pct` a numeric vector denoting the grouping according to the mixtures percentages

`rgr` a numeric vector of response values (relative growth rates)

Details

The dataset is analysed in Soerensen et al (2007). The concentration addition model can be entertained for this dataset.

Source

The dataset is kindly provided by Nina Cedergreen, Department of Agricultural Sciences, Royal Veterinary and Agricultural University, Denmark.

References

Soerensen, H. and Cedergreen, N. and Skovgaard, I. M. and Streibig, J. C. (2007) An isobole-based statistical model and test for synergism/antagonism in binary mixture toxicity experiments, *Environmental and Ecological Statistics*, **14**, 383–397.

Examples

```
## Fitting the model with freely varying ED50 values
glymet.free <- drm(rgr~dose, pct, data = glymet,
fct = LL.3(), pmodels = list(~factor(pct) , ~1, ~factor(pct)))

## Lack-of-fit test
modelFit(glymet.free)  # acceptable
summary(glymet.free)

## Plotting isobole structure
isobole(glymet.free, exchange=0.01)

## Fitting the concentration addition model
glymet.ca <- mixture(glymet.free, model = "CA")

## Comparing to model with freely varying e parameter
anova(glymet.ca, glymet.free)  # borderline accepted

## Plotting isobole based on concentration addition
isobole(glymet.free, glymet.ca, exchange = 0.01)  # acceptable fit

## Fitting the Hewlett model
glymet.hew <- mixture(glymet.free, model = "Hewlett")

### Comparing to model with freely varying e parameter
anova(glymet.ca, glymet.hew)
# borderline accepted
# the Hewlett model offers no improvement over concentration addition

## Plotting isobole based on the Hewlett model
isobole(glymet.free, glymet.hew, exchange = 0.01)
# no improvement over concentration addition
```

`gompertz` *Mean function for the Gompertz dose-response or growth curve*

Description

This function provides a very general way of specifying the mean function of the decreasing or incresing Gompertz dose-response or growth curve models.

Usage

```
gompertz(fixed = c(NA, NA, NA, NA), names = c("b", "c", "d", "e"),
method = c("1", "2", "3", "4"), ssfct = NULL,
fctName, fctText)
```

Arguments

fixed	numeric vector. Specifies which parameters are fixed and at what value they are fixed. NAs for parameter that are not fixed.
names	vector of character strings giving the names of the parameters (should not contain ":"). The order of the parameters is: b, c, d, e (see under 'Details' for the precise meaning of each parameter).
method	character string indicating the self starter function to use.
ssfct	a self starter function to be used.
fctName	character string used internally by convenience functions (optional).
fctText	character string used internally by convenience functions (optional).

Details

The Gompertz model is given by the mean function

$$f(x) = c + (d - c)(\exp(-\exp(b(x - e))))$$

and it is a dose-response/growth curve on the entire real axis, that is it is not limited to non-negative values even though this is the range for most dose-response and growth data. One consequence is that the curve needs not reach the lower asymptote at dose 0.

If

$$b < 0$$

the mean function is increasing and it is decreasing for

$$b > 0$$

. The decreasing Gompertz model is not a well-defined dose-response model and other dose-response models such as the Weibull models should be used instead.

Various re-parameterisations of the model are used in practice.

Value

The value returned is a list containing the non-linear function, the self starter function and the parameter names.

Note

The function is for use with the function drm, but typically the convenience functions G.2, G.3, G.3u, and G.4 should be used.

Author(s)

Christian Ritz

References

Seber, G. A. F. and Wild, C. J. (1989) *Nonlinear Regression*, New York: Wiley \& Sons (p. 331).

See Also

The Weibull model weibull2 is closely related to the Gompertz model.

---

gompertzd *The derivative of the Gompertz function*

Description

'gompertzd' provides a way of specifying the derivative of the Gompertz function as a dose-response model.

Usage

```
gompertzd(fixed = c(NA, NA), names = c("a", "b"))
```

Arguments

fixed	numeric vector. Specifies which parameters are fixed and at what value they are fixed. NAs for parameter that are not fixed.
names	a vector of character strings giving the names of the parameters (should not contain ":"). The default is (notice the order): a, b.

Details

The derivative of the Gompertz function is defined as

$$f(x) = a \exp(bx - a/b(exp(bx) - 1))$$

For $a > 0$ and b not 0, the function is decreasing, equaling a at $x = 0$ and approaching 0 at plus infinity.

Value

The value returned is a list containing the model function, the self starter function and the parameter names.

Note

This function is for use with the function drm.

Author(s)

Christian Ritz

H.virescens *Mortality of tobacco budworms*

Description

For three days, moths of the tobacco budworm (*Heliothis virescens*) were exposed to doses of the pyrethroid trans-cypermethrin.

Usage

```
data(H.virescens)
```

Format

A data frame with 12 observations on the following 4 variables.

`dose` a numeric vector of dose values (μg)

`numdead` a numeric vector of dead or knocked-down moths

`total` a numeric vector of total number of moths

`sex` a factor with levels `F` `M` denoting a grouping according to sex

Details

In Venables and Ripley (2002), these data are analysed using a logistic regression with base-2 logarithm of dose as explanatory variable.

Source

Venables, W. N. and Ripley, B. D (2002) *Modern Applied Statistics with S*, New York: Springer (fourth edition).

Examples

```
## Fitting dose-response model (log-logistic with common slope)
Hv.m1 <- drm(numdead/total~dose, sex, weights = total, data = H.virescens, fct = LL.2(),
pmodels = list(~ 1, ~ sex - 1), type = "binomial")
summary(Hv.m1)

## Fitting the same model as in Venables and Riply (2002)
Hv.m2 <- glm(cbind(numdead, total-numdead) ~ sex + I(log2(dose)) - 1, data = H.virescens,
family = binomial)

## Comparing the fits
logLik(Hv.m1)
logLik(Hv.m2)

## Estimated ED values (matching those given in MASS)
```

```
ED(Hv.m1, c(25, 50, 75))
```

`hatvalues.drc` *Model diagnostics for nonlinear dose-response models*

Description

Hat values (leverage values) and Cook's distance are provided for nonlinear dose-response model fits using the same formulas as in linear regression but based on the corresponding but approximate quantities available for nonlinear models.

Usage

```
  ## S3 method for class 'drc'
cooks.distance(model, ...)

  ## S3 method for class 'drc'
hatvalues(model, ...)
```

Arguments

`model`	an object of class 'drc'.
`...`	additional arguments (not used).

Details

Hat values and Cook's distance are calculated using the formula given by Cook et al. (1986) and McCullagh and Nelder (1989).

The output values can be assessed in the same way as in linear regression.

Value

A vector of leverage values (hat values) or values of Cook's distance (one value per observation).

Author(s)

Christian Ritz

References

Cook, R. D. and Tsai, C.-L. and Wei, B. C. (1986) Bias in Nonlinear Regression, *Biometrika* **73**, 615–623.

McCullagh, P. and Nelder, J. A. (1989) emphGeneralized Linear Models, Second edition, Chapman \& Hall/CRC.

Examples

```
ryegrass.LL.4 <- drm(rootl ~ conc, data = ryegrass, fct = LL.4())

hatvalues(ryegrass.LL.4)

cooks.distance(ryegrass.LL.4)
```

`heartrate` *Heart rate baroreflexes for rabbits*

Description

The dataset contains measurements of mean arterial pressure (mmHG) and heart rate (b/min) for a baroreflex curve.

Usage

```
data(heartrate)
```

Format

A data frame with 18 observations on the following 2 variables.

`pressure` a numeric vector containing measurements of arterial pressure.

`rate` a numeric vector containing measurements of heart rate.

Details

The dataset is an example of an asymmetric dose-response curve, that is not easily handled using the log-logistic or Weibull models (`LL.4`, `LL.5`, `W1.4` and `W2.4`), whereas the `baro5` model provides a nice fit.

Source

Ricketts, J. H. and Head, G. A. (1999) A five-parameter logistic equation for investigating asymmetry of curvature in baroreflex studies, *Am. J. Physiol. (Regulatory Integrative Comp. Physiol. 46)*, **277**, 441–454.

Examples

```
## Fitting the baro5 model
heartrate.m1 <- drm(rate~pressure, data=heartrate, fct=baro5())
plot(heartrate.m1)

coef(heartrate.m1)
```

```
#Output:
#b1:(Intercept) b2:(Intercept)  c:(Intercept)  d:(Intercept)  e:(Intercept)
#      11.07984       46.67492      150.33588      351.29613       75.59392

## Inserting the estimated baro5 model function in deriv()
baro5Derivative <- deriv(~ 150.33588 + ((351.29613 - 150.33588)/
(1 + (1/(1 + exp((2 * 11.07984 * 46.67492/(11.07984 + 46.67492)) *
(log(x) - log(75.59392 ))))) * (exp(11.07984 * (log(x) - log(75.59392)))) +
(1 - (1/(1 + exp((2 * 11.07984 * 46.67492/(11.07984 + 46.67492)) *
(log(x) - log(75.59392 )))))) * (exp(46.67492 * (log(x) - log(75.59392 ))))), "x", function(x){})

## Plotting the derivative
#pressureVector <- 50:100
pressureVector <- seq(50, 100, length.out=300)
derivativeVector <- attr(baro5Derivative(pressureVector), "gradient")
plot(pressureVector, derivativeVector, type = "l")

## Finding the minimum
pressureVector[which.min(derivativeVector)]
```

`isobole` *Creating isobolograms*

Description

'isobole' displays isobole based on EC/ED50 estimates from a log-logistic model. Additionally isoboles determined by the concentration addition model, Hewlett's model and Voelund's model can be added to the plot.

Usage

```
isobole(object1, object2, exchange = 1, cifactor = 2, ename = "e",
xaxis = "100", xlab, ylab, xlim, ylim, ...)
```

Arguments

`object1`	object of class 'drc' where EC/ED50 parameters vary freely.
`object2`	object of class 'drc' where EC/ED50 parameters vary according to Hewlett's model.
`ename`	character string. The name of the EC/ED50 variable.
`xaxis`	character string. Is the mixture "0:100" or "100:0" on the x axis?
`exchange`	numeric. The exchange rate between the two substances.
`cifactor`	numeric. The factor to be used in the confidence intervals. Default is 2, but 1 has been used in publications.

xlab	an optional label for the x axis.
ylab	an optional label for the y axis.
xlim	a numeric vector of length two, containing the lower and upper limit for the x axis.
ylim	a numeric vector of length two, containing the lower and upper limit for the y axis.
...	Additional graphical parameters.

Details

The model fits to be supplied as first and optionally second argument are obtained using mixture and drm.

Value

No value is returned. Only used for the side effect: the isobologram shown.

Author(s)

Christian Ritz

References

Ritz, C. and Streibig, J. C. (2014) From additivity to synergism - A modelling perspective *Synergy*, **1**, 22–29.

See Also

The examples in acidiq, glymet and mecter.

---

leaflength *Leaf length of barley*

Description

In an experiment barley was grown in a hydroponic solution with a herbicide.

Usage

```
data(leaflength)
```

Format

A data frame with 42 observations on the following 2 variables.

Dose a numeric vector

DW a numeric vector

Details

The dataset exhibits a large hormetical effect.

Source

Nina Cedergreen, Royal Veterinary and Agricultural University, Denmark.

Examples

```
## Fitting a hormesis model
leaflength.crs4c1 <- drm(DW ~ Dose, data = leaflength, fct = CRS.4c())
plot(fitted(leaflength.crs4c1), residuals(leaflength.crs4c1))

leaflength.crs4c2 <- boxcox(drm(DW ~ Dose, data = leaflength, fct = CRS.4c()),
method = "anova", plotit = FALSE)
summary(leaflength.crs4c2)

## Plottinf fitted curve and original data
plot(leaflength.crs4c2, broken = TRUE, conLevel = 0.001, type = "all", legend = FALSE,
ylab = "Produced leaf length (cm)", xlab = "Metsulfuron-methyl (mg/l)",
main = "Hormesis: leaf length of barley")
```

---

lepidium *Dose-response profile of degradation of agrochemical using lepidium*

Description

Estimation of the degradation profile of an agrochemical based on soil samples at depth 0-10cm from a calibration experiment.

Usage

```
data(lepidium)
```

Format

A data frame with 42 observations on the following 2 variables.

`conc` a numeric vector of concentrations (g/ha)

`weight` a numeric vector of plant weight (g) after 3 weeks' growth

Details

It is an experiment with seven concentrations and six replicates per concentration. *Lepidium* is rather robust as it only responds to high concentrations.

Source

Racine-Poon, A. (1988) A Bayesian Approach to Nonlinear Calibration Problems, *J. Am. Statist. Ass.*, **83**, 650–656.

Examples

```
lepidium.m1 <- drm(weight~conc, data=lepidium, fct = LL.4())

modelFit(lepidium.m1)

plot(lepidium.m1, type = "all", log = "")
```

---

lettuce *Hormesis in lettuce plants*

Description

Data are from an experiment where isobutylalcohol was dissolved in a nutrient solution in which lettuce (*Lactuca sativa*) plants were grown. The plant biomass of the shoot was determined af 21 days.

Usage

```
data(lettuce)
```

Format

A data frame with 14 observations on the following 2 variables.

conc a numeric vector of concentrations of isobutylalcohol (mg/l)

weight a numeric vector of biomass of shoot (g)

Details

The data set illustrates hormesis, presence of a subtoxic stimulus at low concentrations.

Source

van Ewijk, P. H. and Hoekstra, J. A. (1993) Calculation of the EC50 and its Confidence Interval When Subtoxic Stimulus Is Present, *ECOTOXICOLOGY AND ENVIRONMENTAL SAFETY*, **25**, 25–32.

References

van Ewijk, P. H. and Hoekstra, J. A. (1994) Curvature Measures and Confidence Intervals for the Linear Logistic Model, *Appl. Statist.*, **43**, 477–487.

Examples

```
## Look at data
lettuce

## Monotonous dose-response model
lettuce.m1 <- drm(weight~conc, data=lettuce, fct=LL.3())

plot(lettuce.m1, broken = TRUE)

## Model fit in van Ewijk and Hoekstra (1994)
lettuce.m2 <- drm(weight~conc, data=lettuce, fct=BC.4())
modelFit(lettuce.m2)

plot(lettuce.m2, add = TRUE, broken = TRUE, type = "none", lty = 2)

## Hormesis effect only slightly significant
summary(lettuce.m2)

## Hormesis effect highly significant
##  compare with t-test for the "f" parameter in the summary output)
anova(lettuce.m1, lettuce.m2)
```

---

lin.test *Lack-of-fit test for the mean structure based on cumulated residuals*

Description

The function provides a lack-of-fit test for the mean structure based on cumulated residuals from the model fit.

Usage

```
lin.test(object, noksSim = 20, seed = 20070325, plotit = TRUE,
log = "", bp = 0.01, xlab, ylab, ylim, ...)
```

Arguments

object	object of class 'drc'.
noksSim	numeric specifying the number of simulations used to obtain the p-value.
seed	numeric specifying the seed value for the random number generator.
plotit	logical indicating whether or not the observed cumulated residual process should be plotted. Default is to plot the process.
log	character string which should contains '"x"' if the x axis is to be logarithmic, '"y"' if the y axis is to be logarithmic and '"xy"' or '"yx"' if both axes are to be logarithmic. The default is "x". The empty string "" yields the original axes.

bp	numeric value specifying the break point below which the dose is zero (the amount of stretching on the dose axis above zero in order to create the visual illusion of a logarithmic scale *including* 0).
xlab	string character specifying an optional label for the x axis.
ylab	character string specifying an optional label for the y axis.
ylim	numeric vector of length two, containing the lower and upper limit for the y axis.
...	additional arguments to be passed further to the basic plot method.

Details

The function provides a graphical model checking of the mean structure in a dose-response model. The graphical display is supplemented by a p-value based on a supremum-type test.

The test is applicable even in cases where data are non-normal or exhibit variance heterogeneity.

Value

A p-value for test of the null hypothesis that the mean structure is appropriate. Ritz and Martinussen (2009) provide the details.

Author(s)

Christian Ritz

References

Ritz, C and Martinussen, T. (2009) Lack-of-fit tests for assessing mean structures for continuous dose-response data, *Submitted manuscript*

See Also

Other available lack-of-fit tests are the Neill test (neill.test) and ANOVA-based test (modelFit).

Examples

```
## Fitting a log-logistic model to the dataset 'etmotc'
etmotc.m1<-drm(rgr1~dose1, data=etmotc[1:15,], fct=LL.4())

## Test based on umulated residuals
lin.test(etmotc.m1, 1000)
#lin.test(etmotc.m1, 10000, plotit = FALSE)  # more precise

## Fitting an exponential model to the dataset 'O.mykiss'
O.mykiss.m1<-drm(weight~conc, data=O.mykiss, fct=EXD.2(), na.action=na.omit)

## ANOVA-based test
modelFit(O.mykiss.m1)

## Test based on umulated residuals
lin.test(O.mykiss.m1, log = "", cl = 0.2, xlab = "Dose (mg/l)", main = "B", ylim = c(-0.6, 0.6))
```

```
#lin.test(O.mykiss.m1, noksSim = 10000, plotit = FALSE)  # more precise
```

LL.2 *The two-parameter log-logistic function*

Description

'LL.2' and 'LL2.2' provide the two-parameter log-logistic function where the lower limit is fixed at 0 and the upper limit is fixed at 1, mostly suitable for binomial/quantal responses.

Usage

```
LL.2(upper = 1, fixed = c(NA, NA), names = c("b", "e"), ...)

l2(upper = 1, fixed = c(NA, NA), names = c("b", "e"), ...)

LL2.2(upper = 1, fixed = c(NA, NA), names = c("b", "e"), ...)
```

Arguments

upper	numeric value. The fixed, upper limit in the model. Default is 1.
fixed	numeric vector. Specifies which parameters are fixed and at what value they are fixed. NAs for parameter that are not fixed.
names	a vector of character strings giving the names of the parameters. The default is reasonable.
...	Additional arguments (see llogistic).

Details

The two-parameter log-logistic function is given by the expression

$$f(x) = \frac{1}{1 + \exp(b(\log(x) - \log(e)))}$$

or in another parameterisation

$$f(x) = \frac{1}{1 + \exp(b(\log(x) - e))}$$

The model function is symmetric about the inflection point (e).

Value

See llogistic.

Note

This function is for use with the function drm.

Author(s)

Christian Ritz

See Also

Related functions are LL.3, LL.4, LL.5 and the more general llogistic.

Examples

```
## Fitting a two-parameter logistic model
##  to binomial responses (a logit model)
earthworms.m1 <- drm(number/total~dose, weights=total,
data = earthworms, fct = LL.2(), type = "binomial")

plot(earthworms.m1)  # not fitting at the upper limit!
```

LL.3 *The three-parameter log-logistic function*

Description

'LL.3' and 'LL2.3' provide the three-parameter log-logistic function where the lower limit is equal to 0.

'LL.3u' and 'LL2.3u' provide three-parameter logistic function where the upper limit is equal to 1, mainly for use with binomial/quantal response.

Usage

```
LL.3(fixed = c(NA, NA, NA), names = c("b", "d", "e"), ...)

LL.3u(upper = 1, fixed = c(NA, NA, NA), names = c("b", "c", "e"), ...)

l3(fixed = c(NA, NA, NA), names = c("b", "d", "e"), ...)

l3u(upper = 1, fixed = c(NA, NA, NA), names = c("b", "c", "e"), ...)

LL2.3(fixed = c(NA, NA, NA), names = c("b", "d", "e"), ...)

LL2.3u(upper = 1, fixed = c(NA, NA, NA), names = c("b", "c", "e"), ...)
```

Arguments

`upper`	numeric value. The fixed, upper limit in the model. Default is 1.
`fixed`	numeric vector. Specifies which parameters are fixed and at what value they are fixed. NAs for parameter that are not fixed.
`names`	a vector of character strings giving the names of the parameters. The default is reasonable.
`...`	Additional arguments (see `llogistic`).

Details

The three-parameter log-logistic function with lower limit 0 is

$$f(x) = 0 + \frac{d - 0}{1 + \exp(b(\log(x) - \log(e)))}$$

or in another parameterisation

$$f(x) = 0 + \frac{d - 0}{1 + \exp(b(\log(x) - e))}$$

The three-parameter log-logistic function with upper limit 1 is

$$f(x) = c + \frac{1 - c}{1 + \exp(b(\log(x) - \log(e)))}$$

or in another parameterisation

$$f(x) = c + \frac{1 - c}{1 + \exp(b(\log(x) - e))}$$

Both functions are symmetric about the inflection point (e).

Value

See `llogistic`.

Note

This function is for use with the function `drm`.

Author(s)

Christian Ritz

References

Finney, D. J. (1971) *Probit Analysis*, Cambridge: Cambridge University Press.

See Also

Related functions are LL.2, LL.4, LL.5 and the more general llogistic.

Examples

```
## Fitting model with lower limit equal 0
ryegrass.model1 <- drm(rootl ~ conc, data = ryegrass, fct = LL.3())
summary(ryegrass.model1)

## Fitting binomial response
##  with non-zero control response

## Example dataset from Finney (1971) - example 19
logdose <- c(2.17, 2,1.68,1.08,-Inf,1.79,1.66,1.49,1.17,0.57)
n <- c(142,127,128,126,129,125,117,127,51,132)
r <- c(142,126,115,58,21,125,115,114,40,37)
treatment <- factor(c("w213","w213","w213","w213",
"w214","w214","w214","w214","w214","w214"))
# Note that the control is included in one of the two treatment groups
finney.ex19 <- data.frame(logdose, n, r, treatment)

## Fitting model where the lower limit is estimated
fe19.model1 <- drm(r/n~logdose, treatment, weights = n, data = finney.ex19,
logDose = 10, fct = LL.3u(), type="binomial",
pmodels = data.frame(treatment, 1, treatment))

summary(fe19.model1)
modelFit(fe19.model1)
plot(fe19.model1, ylim = c(0, 1.1), bp = -1, broken = TRUE, legendPos = c(0, 1))
abline(h = 1, lty = 2)
```

LL.4 *The four-parameter log-logistic function*

Description

'LL.4' and 'LL2.4' provide the four-parameter log-logistic function, self starter function, names of the parameters and, optionally, first and second derivatives for a faster estimation.

Usage

```
LL.4(fixed = c(NA, NA, NA, NA), names = c("b", "c", "d", "e"), ...)

l4(fixed = c(NA, NA, NA, NA), names = c("b", "c", "d", "e"), ...)

LL2.4(fixed = c(NA, NA, NA, NA), names = c("b", "c", "d", "e"), ...)
```

Arguments

fixed	numeric vector. Specifies which parameters are fixed and at what value they are fixed. NAs for parameter that are not fixed.
names	a vector of character strings giving the names of the parameters. The default is reasonable.
...	Additional arguments (see llogistic).

Details

The four-parameter log-logistic function is given by the expression

$$f(x) = c + \frac{d - c}{1 + \exp(b(\log(x) - \log(e)))}$$

or in another parameterisation (converting the term $\log(e)$ into a parameter)

$$f(x) = c + \frac{d - c}{1 + \exp(b(\log(x) - \tilde{e}))}$$

The function is symmetric about the inflection point (e).

Value

See llogistic.

Note

This function is for use with the function drm.

Author(s)

Christian Ritz and Jens C. Streibig

References

Seber, G. A. F. and Wild, C. J (1989) *Nonlinear Regression*, New York: Wiley \& Sons (p. 330).

See Also

Setting $c = 0$ yields LL.3. See also LL.5.

Examples

```
spinach.m1 <- drm(SLOPE~DOSE, CURVE, data = spinach, fct = LL.4())
spinach.m1
```

LL.5 *The five-parameter log-logistic function*

Description

'LL.5' and 'LL2.5' provide the five-parameter log-logistic function, self starter function and names of the parameters.

Usage

```
LL.5(fixed = c(NA, NA, NA, NA, NA), names = c("b", "c", "d", "e", "f"), ...)

l5(fixed = c(NA, NA, NA, NA, NA), names = c("b", "c", "d", "e", "f"), ...)

LL2.5(fixed = c(NA, NA, NA, NA, NA), names = c("b", "c", "d", "e", "f"), ...)
```

Arguments

`fixed`	numeric vector. Specifies which parameters are fixed and at what value they are fixed. NAs for parameter that are not fixed.
`names`	a vector of character strings giving the names of the parameters. The default is reasonable.
`...`	Additional arguments (see `llogistic`).

Details

The five-parameter logistic function is given by the expression

$$f(x) = c + \frac{d-c}{(1+\exp(b(\log(x)-\log(e))))^f}$$

or in another parameterisation

$$f(x) = c + \frac{d-c}{(1+\exp(b(\log(x)-e)))^f}$$

The function is asymmetric for f different from 1.

Value

See `llogistic`.

Note

This function is for use with the function `drm`.

Author(s)

Christian Ritz

References

Finney, D. J. (1979) Bioassay and the Practise of Statistical Inference, *Int. Statist. Rev.*, **47**, 1–12.

See Also

Related functions are `LL.4` and `LL.3`.

Examples

```
ryegrass.m1 <- drm(rootl ~ conc, data = ryegrass, fct = LL.5())
summary(ryegrass.m1)
```

`llogistic` *The log-logistic function*

Description

'llogistic' provides a very general way of specifying log-logistic models, under various constraints on the parameters.

Usage

```
llogistic(fixed = c(NA, NA, NA, NA, NA),
names = c("b", "c", "d", "e", "f"),
method = c("1", "2", "3", "4"), ssfct = NULL,
fctName, fctText)

llogistic2(fixed = c(NA, NA, NA, NA, NA),
names = c("b", "c", "d", "e", "f"),
ss = c("1", "2", "3"), ssfct = NULL,
fctName, fctText)
```

Arguments

`fixed`	numeric vector. Specifies which parameters are fixed and at what value they are fixed. NAs for parameter that are not fixed.
`names`	a vector of character strings giving the names of the parameters (should not contain ":"). The default is reasonable (see under 'Usage'). The order of the parameters is: b, c, d, e, f (see under 'Details').
`method`	character string indicating the self starter function to use.
`ss`	character string indicating the self starter function to use.
`ssfct`	a self starter function to be used.
`fctName`	optional character string used internally by convenience functions.
`fctText`	optional character string used internally by convenience functions.

Details

The default arguments yields the five-parameter log-logistic function given by the expression

$$f(x) = c + \frac{d - c}{(1 + \exp(b(\log(x) - \log(e))))^f}$$

If the parameter f differs from 1 then the function is asymmetric; otherwise it is symmetric (on log scale). This function is fitted using `llogistic`.

The log-logistic function with log(e) rather than e as a parameter, that is using the parameterisation

$$f(x) = c + \frac{d - c}{(1 + \exp(b(\log(x) - e)))^f}$$

is fitted using `llogistic2`.

Sometimes the log-logistic models are also called Hill models.

Value

The value returned is a list containing the nonlinear function, the self starter function and the parameter names.

Note

The functions are for use with the function `drm`.

Author(s)

Christian Ritz

References

Finney, D. J. (1979) Bioassay and the Practise of Statistical Inference, *Int. Statist. Rev.*, **47**, 1–12.

Seber, G. A. F. and Wild, C. J. (1989) *Nonlinear Regression*, New York: Wiley \& Sons (p. 330).

See Also

For convenience several special cases are available: `LL.2`, `LL.3`, `LL.4` and `LL.5`. Examples are provided in the help pages for these functions.

`lnormal` *Log-normal dose-response model*

Description

`lnormal` and the accompanying convenience functions provide a general framework for specifying the mean function of the decreasing or incresing log-normal dose-response model.

Usage

```
lnormal(fixed = c(NA, NA, NA, NA), names = c("b", "c", "d", "e"),
method = c("1", "2", "3", "4"), ssfct = NULL,
fctName, fctText, loge = FALSE)

LN.2(upper = 1, fixed = c(NA, NA), names = c("b", "e"), ...)

LN.3(fixed = c(NA, NA, NA), names = c("b", "d", "e"), ...)

LN.3u(upper = 1, fixed = c(NA, NA, NA), names = c("b", "c", "e"), ...)

LN.4(fixed = c(NA, NA, NA, NA), names = c("b", "c", "d", "e"), ...)
```

Arguments

`fixed`	numeric vector. Specifies which parameters are fixed and at what value they are fixed. NAs for parameter that are not fixed.
`names`	vector of character strings giving the names of the parameters (should not contain ":"). The default is reasonable (see under 'Usage'). The order of the parameters is: b, c, d, e, f (see under 'Details' for the precise meaning of each parameter).
`method`	character string indicating the self starter function to use.
`ssfct`	a self starter function to be used.
`fctName`	character string used internally by convenience functions (optional).
`fctText`	character string used internally by convenience functions (optional).
`loge`	logical indicating whether or not ED50 or log(ED50) should be a parameter in the model. By default ED50 is a model parameter.
`upper`	numeric specifying the upper horizontal asymptote in the convenience function. The default is 1.
`...`	additional arguments to be passed from the convenience functions to `lnormal`.

Details

For the case where log(ED50), denoted e in the equation below, is a parameter in the model, the mean function is:

$$f(x) = c + (d - c)(\Phi(b(\log(x) - e)))$$

and the mean function is:

$$f(x) = c + (d - c)(\Phi(b(\log(x) - \log(e))))$$

in case ED50, which is also denoted e, is a parameter in the model. If the former model is fitted any estimated ED values will need to be back-transformed subsequently in order to obtain effective doses on the original scale.

The mean functions above yield the same models as those described by Bruce and Versteeg (1992), but in a different parameterisations (among other things the natural logarithm is used).

For the case $c = 0$ and $d = 1$, the log-normal model reduces the classic probit model (Finney, 1971) with log dose as explanatory variable (mostly used for quantal data). This special case is available through the convenience function `LN.2`.

The case $c = 0$ is available as the function `LN.3`, whereas the `LN.3u` corresponds to the special case where the upper horizontal asymptote is fixed (default is 1). The full four-parameter model is available through `LN.4`.

Value

The value returned is a list containing the non-linear function, the self starter function and the parameter names.

Note

The function is for use with the function `drm`, but typically the convenience functions `link{LN.2}`, `link{LN.3}`, `link{LN.3u}`, and `link{LN.4}` should be used.

Author(s)

Christian Ritz

References

Finney, D. J. (1971) *Probit analysis*, London: Cambridge University Press.

Bruce, R. D. and Versteeg, D. J. (1992) A statistical procedure for modeling continuous toxicity data, *Environ. Toxicol. Chem.*, **11**, 1485–1494.

See Also

The log-logistic model (`llogistic`) is very similar to the log-normal model at least in the middle, but they may differ in the tails and thus provide different estimates of low effect concentrations EC/ED.

Examples are provided in the help pages of the datasets `S.capricornutum`, `P.promelas`, and `M.bahia`.

`logistic` *The logistic model*

Description

The general asymmetric five-parameter logistic model for describing dose-response relationships.

Usage

```
logistic(fixed = c(NA, NA, NA, NA, NA), names = c("b", "c", "d", "e", "f"),
method = c("1", "2", "3", "4"), ssfct = NULL,
fctName, fctText)

L.3(fixed = c(NA, NA, NA), names = c("b", "d", "e"), ...)
L.4(fixed = c(NA, NA, NA, NA), names = c("b", "c", "d", "e"), ...)
L.5(fixed = c(NA, NA, NA, NA, NA), names = c("b", "c", "d", "e", "f"), ...)
```

Arguments

`fixed`	numeric vector. Specifies which parameters are fixed and at what value they are fixed. NAs for parameter that are not fixed.
`names`	a vector of character strings giving the names of the parameters (should not contain ":"). The order of the parameters is: b, c, d, e, f (see under 'Details').
`method`	character string indicating the self starter function to use.
`ssfct`	a self starter function to be used.
`fctName`	optional character string used internally by convenience functions.
`fctText`	optional character string used internally by convenience functions.
`...`	Additional arguments (see `llogistic`).

Details

The default arguments yields the five-parameter logistic mean function given by the expression

$$f(x) = c + \frac{d - c}{(1 + \exp(b(x - e)))^f}$$

The model is different from the log-logistic models `llogistic` and `llogistic2` where the term

$$log(x)$$

is used instead of

$$x$$

.

The model is sometimes referred to as the Boltzmann model.

Value

The value returned is a list containing the nonlinear function, the self starter function and the parameter names.

Author(s)

Christian Ritz

Examples

```
## Fitting the four-parameter logistic model
ryegrass.m1 <- drm(rootl ~ conc, data = ryegrass, fct = L.4())
summary(ryegrass.m1)

## Fitting an asymmetric logistic model
##  requires installing the package 'NISTnls'
# Ratkowsky3.m1 <- drm(y~x, data = Ratkowsky3,
# fct = L.5(fixed = c(NA, 0, NA, NA, NA)))
# plot(Ratkowsky3.m1)
# summary(Ratkowsky3.m1)
## okay agreement with NIST values
##  for the two parameters that are the same
```

---

logLik.drc *Extracting the log likelihood*

Description

`loglik` extracts the value of the log likelihood function evaluated at the parameter estimates.

Usage

```
  ## S3 method for class 'drc'
logLik(object, ...)
```

Arguments

`object`	an object of class 'drc'.
`...`	additional arguments.

Value

The evaluated log likelihood as a numeric value and the corresponding degrees of freedom as well as the number of observations as attributes.

Note

The value of the log likelihood could be used to compare model fits of the same data based on different dose-response models or based on the same model but fitted different algorithms, software programmes, or starting values. For comparisons: Larger is better.

Author(s)

Christian Ritz

Examples

```
## Fitting a four-parameter log-logistic model
ryegrass.m1 <- drm(rootl ~conc, data = ryegrass, fct = LL.4())
logLik(ryegrass.m1)
```

M.bahia *Effect of an effluent on the growth of mysid shrimp*

Description

Juvenile mysid shrimp (*Mysidopsis bahia*) were exposed to up to 32% effluent in a 7-day survival and growth test. The average weight per treatment replicate of surviving organisms was measured.

Usage

```
data(M.bahia)
```

Format

A data frame with 40 observations on the following 2 variables.

`conc` a numeric vector of effluent concentrations (%)

`dryweight` a numeric vector of average dry weights (mg)

Details

The data are analysed in Bruce and Versteeg (1992) using a log-normal dose-response model (using the logarithm with base 10).

At 32% there was complete mortality, and this justifies using a model where a lower asymptote of 0 is assumed.

Source

Bruce, R. D. and Versteeg, D. J. (1992) A statistical procedure for modeling continuous toxicity data, *Environ. Toxicol. Chem.*, **11**, 1485–1494.

Examples

```
M.bahia.m1 <- drm(dryweight~conc, data=M.bahia, fct=LN.3())

## Variation increasing
plot(fitted(M.bahia.m1), residuals(M.bahia.m1))

## Using transform-both-sides approach
M.bahia.m2 <- boxcox(M.bahia.m1, method = "anova")
summary(M.bahia.m2)  # logarithm transformation

## Variation roughly constant, but still not a great fit
plot(fitted(M.bahia.m2), residuals(M.bahia.m2))

## Visual comparison of fits
plot(M.bahia.m1, type="all", broken=TRUE)
plot(M.bahia.m2, add=TRUE, type="none", broken=TRUE, lty=2)

ED(M.bahia.m2, c(10,20,50), ci="fls")

## A better fit
M.bahia.m3 <- boxcox(update(M.bahia.m1, fct = LN.4()), method = "anova")
#plot(fitted(M.bahia.m3), residuals(M.bahia.m3))
plot(M.bahia.m3, add=TRUE, type="none", broken=TRUE, lty=3, col=2)
ED(M.bahia.m3, c(10,20,50), ci="fls")
```

---

maED *Estimation of ED values using model-averaging*

Description

Estimates and confidence intervals for ED values are estimated using model-averaging.

Usage

```
maED(object, fctList = NULL, respLev, interval = c("none", "buckland", "kang"),
linreg = FALSE, clevel = NULL, level = 0.95, type = c("relative", "absolute"),
display = TRUE, na.rm = FALSE, extended = FALSE)
```

Arguments

object	an object of class 'drc'.
fctList	a list of non-linear functions to be compared.
respLev	a numeric vector containing the response levels.
interval	character string specifying the type of confidence intervals to be supplied. The default is "none". The choices "buckland" and "kang" are explained in the Details section.
linreg	logical indicating whether or not additionally a simple linear regression model should be fitted.
clevel	character string specifying the curve id in case on estimates for a specific curve or compound is requested. By default estimates are shown for all curves.
level	numeric. The level for the confidence intervals. The default is 0.95.
type	character string. Whether the specified response levels are absolute or relative (default).
display	logical. If TRUE results are displayed. Otherwise they are not (useful in simulations).
na.rm	logical indicating whether or not NA occurring during model fitting should be left out of subsequent calculations.
extended	logical specifying whether or not an extended output (including fit summaries) should be returned.

Details

Model-averaging of individual estimates is carried out as described by Buckland *et al.* (1997) and Kang *et al.* (2000) using AIC-based weights. The two approaches differ w.r.t. the calculation of confidence intervals: Buckland *et al.* (1997) provide an approximate variance formula under the assumption of perfectly correlated estimates (so, confidence intervals will tend to be too wide). Kang *et al.* (2000) use the model weights to calculate confidence limits as weighted means of the confidence limits for the individual fits; this procedure corresponds to using the standard error in Equation (3) given by Buckland *et al.* (1997) (assuming symmetric confidence intervals based on the same percentile).

Value

A matrix with two or more columns, containing the estimates and the corresponding estimated standard errors and possibly lower and upper confidence limits.

Author(s)

Christian Ritz

References

Buckland, S. T. and Burnham, K. P. and Augustin, N. H. (1997) Model Selection: An Integral Part of Inference, *Biometrics* **53**, 603–618.

Kang, Seung-Ho and Kodell, Ralph L. and Chen, James J. (2000) Incorporating Model Uncertainties along with Data Uncertainties in Microbial Risk Assessment, *Regulatory Toxicology and Pharmacology* **32**, 68–72.

See Also

The function `mselect` provides a summary of fit statistics for several models fitted to the same data.

Examples

```
## Fitting an example dose-response model
ryegrass.m1 <- drm(rootl~conc, data = ryegrass, fct = LL.4())

## Comparing models (showing the AIC values)
mselect(ryegrass.m1,
list(LL.5(), LN.4(), W1.4(), W2.4(), FPL.4(-1,1), FPL.4(-2,3), FPL.4(-0.5,0.5)))

## Doing the actual model-averaging
maED(ryegrass.m1,
list(LL.5(), LN.4(), W1.4(), W2.4(), FPL.4(-1,1), FPL.4(-2,3), FPL.4(-0.5,0.5)),
c(10, 50, 90))

## With confidence intervals according to Buckland et al. (1997)
maED(ryegrass.m1,
list(LL.5(), LN.4(), W1.4(), W2.4(), FPL.4(-1,1), FPL.4(-2,3), FPL.4(-0.5,0.5)),
c(10, 50, 90), "buckland")

## With confidence intervals according to Kang et al. (2000)
maED(ryegrass.m1,
list(LL.5(), LN.4(), W1.4(), W2.4(), FPL.4(-1,1), FPL.4(-2,3), FPL.4(-0.5,0.5)),
c(10, 50, 90), "kang")

## Comparing to model-averaged ED values with simple linear regression included
maED(ryegrass.m1,
list(LL.5(), LN.4(), W1.4(), W2.4(), FPL.4(-1,1), FPL.4(-2,3), FPL.4(-0.5,0.5)),
c(10, 50, 90), interval = "buckland", linreg = TRUE)



## Example with a model fit involving two compounds/curves
S.alba.m1 <- drm(DryMatter~Dose, Herbicide, data=S.alba, fct = LL.4(),
pmodels=data.frame(Herbicide,1,1,Herbicide))

## Model-averaged ED50 for both compounds
maED(S.alba.m1, list(LL.3(), LN.4()), 50)

## Model-averaged ED50 only for one compound (glyphosate)
maED(S.alba.m1, list(LL.3(), LN.4()), 50, clevel="Glyphosate")

## With confidence intervals
maED(S.alba.m1, list(LL.3(), LN.4()), 50, interval="buckland")
```

```
## For comparison model-specific confidence intervals
ED(S.alba.m1, 50, interval="delta")  # wider!
```

MAX *Maximum mean response*

Description

MAX estimates the maximum mean response and the dose at which it occurs.

Usage

```
MAX(object, lower = 1e-3, upper = 1000, pool = TRUE)
```

Arguments

object	an object of class 'drc'.
lower	numeric. Lower limit for bisection method. Need to be smaller than EDx level to be calculated.
upper	numeric. Upper limit for bisection method. Need to be larger than EDx level to be calculated.
pool	logical. If TRUE curves are pooled. Otherwise they are not. This argument only works for models with independently fitted curves as specified in drm.

Details

This function is only implemented for the built-in functions of class braincousens and cedergreen.

This function was used for obtaining the results on hormesis effect size reported in Cedergreen et al. (2005).

Value

A matrix with one row per curve in the data set and two columns: one containing the dose at which the maximum occurs and one containing the corresponding maximum response.

Author(s)

Christian Ritz

References

Cedergreen, N. and Ritz, C. and Streibig, J. C. (2005) Improved empirical models describing hormesis, *Environmental Toxicology and Chemistry* **24**, 3166–3172.

Examples

```
## Fitting a Cedergreen-Ritz-Streibig model
lettuce.m1 <- drm(weight~conc, data = lettuce, fct = CRS.4c())

## Finding maximum average response and the corrresponding dose
MAX(lettuce.m1)
```

`mecter` *Mechlorprop and terbythylazine tested on Lemna minor*

Description

Data consist of 5 mixture, 6 dilutions, three replicates, and 12 common controls; in total 102 onservations.

Usage

```
data(mecter)
```

Format

A data frame with 102 observations on the following 3 variables.

`dose` a numeric vector of dose values

`pct` a numeric vector denoting the grouping according to the mixtures percentages

`rgr` a numeric vector of response values (relative growth rates)

Details

The dataset is analysed in Soerensen et al (2007). The asymmetric Voelund model is appropriate, whereas the symmetric Hewlett model is not.

Source

The dataset is kindly provided by Nina Cedergreen, Department of Agricultural Sciences, Royal Veterinary and Agricultural University, Denmark.

References

Soerensen, H. and Cedergreen, N. and Skovgaard, I. M. and Streibig, J. C. (2007) An isobole-based statistical model and test for synergism/antagonism in binary mixture toxicity experiments, *Environmental and Ecological Statistics*, **14**, 383–397.

Examples

```
## Fitting the model with freely varying ED50 values
mecter.free <- drm(rgr ~ dose, pct, data = mecter,
fct = LL.4(), pmodels = list(~1, ~1, ~1, ~factor(pct) - 1))

## Lack-of-fit test
modelFit(mecter.free)  # not really acceptable
summary(mecter.free)

## Plotting isobole structure
isobole(mecter.free, exchange = 0.02)

## Fitting the concentration addition model
mecter.ca <- mixture(mecter.free, model = "CA")

## Comparing to model with freely varying e parameter
anova(mecter.ca, mecter.free)  # rejected

## Plotting isobole based on concentration addition
isobole(mecter.free, mecter.ca, exchange = 0.02)  # poor fit

## Fitting the Hewlett model
mecter.hew <- mixture(mecter.free, model = "Hewlett")

## Comparing to model with freely varying e parameter
anova(mecter.hew, mecter.free)  # rejected

## Plotting isobole based on the Hewlett model
isobole(mecter.free, mecter.hew, exchange = 0.02)  # poor fit

## Fitting the Voelund model
mecter.voe<-mixture(mecter.free, model = "Voelund")

## Comparing to model with freely varying e parameter
anova(mecter.voe, mecter.free)  # accepted

## Plotting isobole based on the Voelund model
isobole(mecter.free, mecter.voe, exchange = 0.02)  # good fit
```

`metals` *Data from heavy metal mixture experiments*

Description

Data are from a study of the response of the cyanobacterial self-luminescent metallothionein-based whole-cell biosensor Synechoccocus elongatus PCC 7942 pBG2120 to binary mixtures of 6 heavy metals (Zn, Cu, Cd, Ag, Co and Hg).

Usage

```
data("metals")
```

Format

A data frame with 543 observations on the following 3 variables.

`metal` a factor with levels `Ag` `AgCd` `Cd` `Co` `CoAg` `CoCd` `Cu` `CuAg` `CuCd` `CuCo` `CuHg` `CuZn` `Hg` `HgCd` `HgCo` `Zn` `ZnAg` `ZnCd` `ZnCo` `ZnHg`

`conc` a numeric vector of concentrations

`BIF` a numeric vector of luminescence induction factors

Details

Data are from the study described by Martin-Betancor et al. (2015).

Source

Martin-Betancor, K. and Ritz, C. and Fernandez-Pinas, F. and Leganes, F. and Rodea-Palomares, I. (2015) Defining an additivity framework for mixture research in inducible whole-cell biosensors, *Scientific Reports* **17200**.

Examples

```
## One example from the paper by Martin-Betancor et al (2015)

## Figure 2

## Fitting a model for "Zn"
Zn.lgau <- drm(BIF ~ conc, data = subset(metals, metal == "Zn"),
fct = lgaussian(), bcVal = 0, bcAdd = 10)

## Plotting data and fitted curve
plot(Zn.lgau, log = "", type = "all",
xlab = expression(paste(plain("Zn")^plain("2+"), " ", mu, "", plain("M"))))

## Calculating effective doses
ED(Zn.lgau, 50, interval = "delta")
ED(Zn.lgau, -50, interval = "delta", bound = FALSE)
ED(Zn.lgau, 99.999,interval = "delta")  # approx. for ED0

## Fitting a model for "Cu"
Cu.lgau <- drm(BIF ~ conc, data = subset(metals, metal == "Cu"),
fct = lgaussian())

## Fitting a model for the mixture Cu-Zn
CuZn.lgau <- drm(BIF ~ conc, data = subset(metals, metal == "CuZn"),
fct = lgaussian())

## Calculating effects needed for the FA-CI plot
CuZn.effects <- CIcompX(0.015, list(CuZn.lgau, Cu.lgau, Zn.lgau),
```

```
c(-5, -10, -20, -30, -40, -50, -60, -70, -80, -90, -99, 99, 90, 80, 70, 60, 50, 40, 30, 20, 10))

## Reproducing the FA-cI plot shown in Figure 5d
plotFACI(CuZn.effects, "ED", ylim = c(0.8, 1.6), showPoints = TRUE)
```

`methionine` *Weight gain for different methionine sources*

Description

Data consist of average body weight gain of chickens being treated with one of the two methionine sources DLM and HMTBA.

Usage

```
data(methionine)
```

Format

A data frame with 9 observations on the following 3 variables:

`product` a factor with levels `control`, `DLM` and `MHA` denoting the treatments

`dose` a numeric vector of methionine dose

`gain` a numeric vector of average body weight gain

Details

The dataset contains a common control measurement for the two treatments. More examples using this dataset are found under `AR.2` and `MM.2`.

Source

Kratzer. D. D. and Littell, R. C. (2006) Appropriate Statistical Methods to Compare Dose Responses of Methionine Sources, *Poultry Science*, **85**, 947–954.

Examples

```
## Fitting model with constraint on one parameter
met.ar.m1 <- drm(gain~dose, product, data = methionine,
fct = AR.3(), pmodels = list(~1, ~factor(product), ~factor(product)),
upperl = c(Inf, Inf, 1700, Inf, Inf))

plot(met.ar.m1, xlim=c(0,0.3), ylim=c(1450, 1800))
abline(h=1700, lty=1)

summary(met.ar.m1)
```

`mixture` *Fitting binary mixture models*

Description

'mixture' fits a concentration addition, Hewlett or Voelund model to data from binary mixture toxicity experiments.

Usage

```
mixture(object, model = c("CA", "Hewlett", "Voelund"), start, startm, control = drmc())
```

Arguments

`object`	object of class 'drc' corresponding to the model with freely varying EC50 values.
`model`	character string. It can be "CA", "Hewlett" or "Voelund".
`start`	optional numeric vector supplying starting values for all parameters in the mixture model.
`startm`	optional numeric vector supplying the lambda parameter in the Hewlett model or the eta parameters (two parameters) in the Voelund model.
`control`	list of arguments controlling constrained optimisation (zero as boundary), maximum number of iteration in the optimisation, relative tolerance in the optimisation, warnings issued during the optimisation.

Details

The function is a wrapper to `drm`, implementing the models described in Soerensen et al. (2007). See the paper for a discussion of the merits of the different models.

Currently only the log-logistic models are available. Application of Box-Cox transformation is not yet available.

Value

An object of class 'drc' with a few additional components.

Author(s)

Christian Ritz

References

Ritz, C. and Streibig, J. C. (2014) From additivity to synergism - A modelling perspective *Synergy*, **1**, 22–29.

See Also

The examples in acidiq (the Hewlett model), glymet (dose/concentration addition) and mecter (the Voelund model).

MM *Michaelis-Menten model*

Description

The functions can be used to fit (shifted) Michaelis-Menten models that are used for modeling enzyme kinetics, weed densities etc.

Usage

```
MM.2(fixed = c(NA, NA), names = c("d", "e"), ...)

MM.3(fixed = c(NA, NA, NA), names = c("c", "d", "e"), ...)
```

Arguments

fixed	numeric vector. Specifies which parameters are fixed and at what value they are fixed. NAs for parameter that are not fixed.
names	a vector of character strings giving the names of the parameters (should not contain ":").
...	additional arguments from convenience functions to llogistic.

Details

The model is defined by the three-parameter model function

$$f(x, (c, d, e)) = c + \frac{d - c}{1 + (e/x)}$$

It is an increasing as a function of the dose x, attaining the lower limit c at dose 0 ($x = 0$) and the upper limit d for infinitely large doses. The parameter e corresponds to the dose yielding a response halfway between c and d.

The common two-parameter Michaelis-Menten model (MM.2) is obtained by setting c equal to 0.

Value

A list of class drcMean, containing the mean function, the self starter function, the parameter names and other components such as derivatives and a function for calculating ED values.

Note

At the moment the implementation cannot deal with infinite concentrations.

Author(s)

Christian Ritz

See Also

Related models are the asymptotic regression models AR.2 and AR.3.

Examples

```
## Fitting Michaelis-Menten model
met.mm.m1 <- drm(gain~dose, product, data=methionine, fct=MM.3(),
pmodels = list(~1, ~factor(product), ~factor(product)))
plot(met.mm.m1, log = "", ylim=c(1450, 1800))
summary(met.mm.m1)
ED(met.mm.m1, c(10, 50))

## Calculating bioefficacy: approach 1
coef(met.mm.m1)[4] / coef(met.mm.m1)[5] * 100

## Calculating bioefficacy: approach 2
EDcomp(met.mm.m1, c(50,50))

## Simplified models
met.mm.m2a <- drm(gain~dose, product, data=methionine, fct=MM.3(),
pmodels = list(~1, ~factor(product), ~1))
anova(met.mm.m2a, met.mm.m1)  # model reduction not possible

met.mm.m2b <- drm(gain~dose, product, data=methionine, fct=MM.3(),
pmodels = list(~1, ~1, ~factor(product)))
anova(met.mm.m2b, met.mm.m1)  # model reduction not possible
```

---

modelFit *Assessing the model fit*

Description

Checking the fit of dose-response model by means of formal significance tests or graphical procedures.

Usage

```
modelFit(object, test = NULL, method = c("gof", "cum"))
```

Arguments

object	object of class 'drc'
test	character string defining the test method to apply
method	character string specifying the method to be used for assessing the model fit

Details

Currently two methods are available. For continuous data the clasical lack-of-fit test is applied (Bates and Watts, 1988). The test compares the dose-response model to a more general ANOVA model using an approximate F-test. For quantal data the crude goodness-of-fit test based on Pearson's statistic is used.

None of these tests are very powerful. A significant test result is more alarming than a non-significant one.

Value

An object of class 'anova' which will be displayed in much the same way as an ordinary ANOVA table.

Author(s)

Christian Ritz

References

Bates, D. M. and Watts, D. G. (1988) *Nonlinear Regression Analysis and Its Applications*, New York: Wiley \& Sons (pp. 103–104).

Examples

```
## Comparing the four-parameter log-logistic model
##  to a one-way ANOVA model using an approximate F test
## in other words applying a lack-of-fit test
ryegrass.m1 <- drm(rootl ~ conc, data = ryegrass, fct = W1.4())
modelFit(ryegrass.m1)
```

---

mr.test *Mizon-Richard test for dose-response models*

Description

The function provides a lack-of-fit test for the mean structure based on the Mizon-Richard test as compared to a specific alternative model.

Usage

```
mr.test(object1, object2, object, x, var.equal = TRUE, component = 1)
```

Arguments

object1	object of class 'drc' (null model).
object2	object of class 'drc' (alternative model).
object	object of class 'drc' (fitted model under alternative).
x	numeric vector of dose values.
var.equal	logical indicating whether or not equal variances can be assumed across doses.
component	numeric vector specifying the component(s) in the parameter vector to use in the test.

Details

The function provides a p-value indicating whether or not the mean structure is appropriate.

The test is applicable even in cases where data are non-normal or exhibit variance heterogeneity.

Value

A p-value for test of the null hypothesis that the chosen mean structure is appropriate as compared to the alternative mean structure provided (see Ritz and Martinussen (2011) for a detailed explanation).

Note

This functionality is still experimental: Currently, the null and alternative models are hardcoded! In the future the function will be working for null and alternative models specified by the user.

Author(s)

Christian Ritz

References

Ritz, C and Martinussen, T. (2011) Lack-of-fit tests for assessing mean structures for continuous dose-response data, *Environmental and Ecological Statistics*, **18**, 349–366

See Also

See also modelFit for details on the related lack-of-fit test against an ANOVA model.

Examples

```
## Fitting log-logistic and Weibull models
## The Weibull model is the alternative
etmotc.m1<-drm(rgr1~dose1, data=etmotc[1:15,], fct=LL.4())
etmotc.m2 <- update(etmotc.m1, fct=W1.4())

## Fitting the fitted model (using the alternative model)
etmotc.m3 <- drm(fitted(etmotc.m1)~dose1, data=etmotc[1:15,], fct=W1.4())
```

```
## Handling missing values
xVec <- etmotc[1:15,]$dose1
xVec[1:8] <- 1e-10  # avoiding 0's

## Obtaining the Mizon-Richard test
mr.test(etmotc.m1, etmotc.m2, etmotc.m3, xVec, var.equal = FALSE)
```

`mselect` *Dose-response model selection*

Description

Model selection by comparison of different models using the following criteria: the log likelihood value, Akaike's information criterion (AIC), the estimated residual standard error or the p-value from a lack-of-fit test.

Usage

```
mselect(object, fctList = NULL, nested = FALSE,
sorted = c("IC", "Res var", "Lack of fit", "no"), linreg = FALSE, icfct = AIC)
```

Arguments

`object`	an object of class 'drc'.
`fctList`	a list of dose-response functions to be compared.
`nested`	logical. TRUE results in F tests between adjacent models (in 'fctList'). Only sensible for nested models.
`sorted`	character string determining according to which criterion the model fits are ranked.
`linreg`	logical indicating whether or not additionally polynomial regression models (linear, quadratic, and cubic models) should be fitted (they could be useful for a kind of informal lack-of-test consideration for the models specified, capturing unexpected departures).
`icfct`	function for supplying the information criterion to be used. `AIC` and `BIC` are two options.

Details

For Akaike's information criterion and the residual standard error: the smaller the better and for lack-of-fit test (against a one-way ANOVA model): the larger (the p-value) the better. Note that the residual standard error is only available for continuous dose-response data.

Log likelihood values cannot be used for comparison unless the models are nested.

Value

A matrix with one row for each model and one column for each criterion.

Author(s)

Christian Ritz

Examples

```
### Example with continuous/quantitative data
## Fitting initial four-parameter log-logistic model
ryegrass.m1 <- drm(rootl ~ conc, data = ryegrass, fct = LL.4())

## Model selection
mselect(ryegrass.m1, list(LL.3(), LL.5(), W1.3(), W1.4(), W2.4(), baro5()))

## Model selection including linear, quadratic, and cubic regression models
mselect(ryegrass.m1, list(LL.3(), LL.5(), W1.3(), W1.4(), W2.4(), baro5()), linreg = TRUE)

## Comparing nested models
mselect(ryegrass.m1, list(LL.5()), nested = TRUE)

### Example with quantal data
## Fitting initial two-parameter log-logistic model
earthworms.m1 <- drm(number/total~dose, weights=total,
data = earthworms, fct = LL.2(), type = "binomial")

## Comparing 4 models
mselect(earthworms.m1, list(W1.2(), W2.2(), LL.3()))
```

`multi2` *Multistage dose-response model with quadratic terms*

Description

The multistage dose-response model is a combination of log-logistic models that should be useful for describing more complex dose-response patterns.

Usage

```
multi2(
fixed = c(NA, NA, NA, NA, NA),
names = c("b1", "b2", "b3", "c", "d"),
ssfct = NULL,
fctName,
fctText)
```

Arguments

fixed	numeric vector specifying which parameters are fixed and at what value they are fixed. NAs are used for parameters that are not fixed.
names	a vector of character strings giving the names of the parameters (should not contain ":"). The default is reasonable (see under 'Usage').
ssfct	a self starter function to be used.
fctName	optional character string used internally by convenience functions.
fctText	optional character string used internally by convenience functions.

Details

The multistage model function with quadratic terms is defined as follows

$$f(x) = c + (d - c)\exp(-b1 - b2x - b3x^2)$$

where x denotes the dose or the logarithm-transformed dose.

Value

The value returned is a list containing the nonlinear function, the self starter function and the parameter names.

Author(s)

Christian Ritz

References

Wheeler, M. W., Bailer, A. J. (2009) Comparing model averaging with other model selection strategies for benchmark dose estimation, *Environmental and Ecological Statistics*, **16**, 37–51.

---

nasturtium — *Dose-response profile of degradation of agrochemical using nasturtium*

Description

Estimation of the degradation profile of an agrochemical based on soil samples at depth 0-10cm from a calibration experiment.

Usage

```
data(nasturtium)
```

Format

A data frame with 42 observations on the following 2 variables.

`conc` a numeric vector of concentrations (g/ha)

`weight` a numeric vector of plant weight (mg) after 3 weeks' growth

Details

It is an experiment with seven concentrations and six replicates per concentration. *Nasturtium* is sensitive and its weight reduces noticeable at low concentrations.

Racine-Poon (1988) suggests using a three-parameter log-logistic model.

Source

Racine-Poon, A. (1988) A Bayesian Approach to Nonlinear Calibration Problems, *J. Am. Statist. Ass.*, **83**, 650–656.

Examples

```
nasturtium.m1 <- drm(weight~conc, data=nasturtium, fct = LL.3())

modelFit(nasturtium.m1)

plot(nasturtium.m1, type = "all", log = "", xlab = "Concentration (g/ha)", ylab = "Weight (mg)")
```

NEC *Dose-response model for estimation of no effect concentration (NEC).*

Description

The no effect concentration has been proposed as an alternative to both the classical no observed effect concentration (NOEC) and the regression-based EC/ED approach. The NEC model is a dose-response model with a threshold below which the response is assumed constant and equal to the control response.

Usage

```
NEC(fixed = c(NA, NA, NA, NA), names = c("b", "c", "d", "e"), fctName, fctText)

NEC.2(upper = 1, fixed = c(NA, NA), names = c("b", "e"), ...)

NEC.3(fixed = c(NA, NA, NA), names = c("b", "d", "e"), ...)

NEC.4(fixed = c(NA, NA, NA, NA), names = c("b", "c", "d", "e"), ...)
```

Arguments

fixed	numeric vector specifying which parameters are fixed and at what value they are fixed. NAs are used for parameters that are not fixed.
names	a vector of character strings giving the names of the parameters (should not contain ":"). The default is reasonable (see under 'Usage').
fctName	optional character string used internally by convenience functions.
fctText	optional character string used internally by convenience functions.
upper	numeric value. The fixed, upper limit in the model. Default is 1.
...	additional arguments in NEC

Details

The NEC model function proposed by Pires *et al* (2002) is defined as follows

$$f(x) = c + (d - c)\exp(-b(x - e)I(x - e)) + \frac{d2}{1 + \exp(b2(\log(x) - \log(e2)))}$$

where $I(x - e)$ is the indicator function. It is equal to 0 for $x <= e$ and equal 1 for $x > e$.

In other words: The parameter e in NEC in "drc" corresponds to the parameter c' in Pires *et al* (2002), the parameter b in NEC in "drc" corresponds to the parameter m' in Pires *et al* (2002), the parameter d in NEC in "drc" corresponds to the parameter l' in Pires *et al* (2002), and finally the parameter c in NEC in "drc" (the lower horizontal limit) is (implictly) fixed at 0 in Pires *et al* (2002)

Value

The value returned is a list containing the nonlinear function, the self starter function and the parameter names.

Author(s)

Christian Ritz

References

Pires, A. M., Branco, J. A., Picado, A., Mendonca, E. (2002) Models for the estimation of a 'no effect concentration', *Environmetrics*, **13**, 15–27.

Examples

```
nec.m1 <- drm(rootl~conc, data=ryegrass, fct=NEC.4())

summary(nec.m1)

plot(nec.m1)

abline(v=coef(nec.m1)[4], lty=2)  # showing the estimated threshold
```

`neill.test` *Neill's lack-of-fit test for dose-response models*

Description

'neill.test' provides a lack-of-fit test for non-linear regression models. It is applicable both in cases where there are replicates (in which case it reduces to the standard lack-of-fit test against an ANOVA model) and in cases where there are no replicates, though then a grouping has to be provided.

Usage

```
neill.test(object, grouping, method = c("c-finest", "finest", "percentiles"),
breakp = NULL, display = TRUE)
```

Arguments

`object`	object of class 'drc' or 'nls'.
`grouping`	character or numeric vector that provides the grouping of the dose values.
`method`	character string specifying the method to be used to generate a grouping of the dose values.
`breakp`	numeric vector of break points for generating dose intervals that form a grouping.
`display`	logical. If TRUE results are displayed. Otherwise they are not (useful in simulations).

Details

The functions used the methods `df.residual` and `residuals` and the 'data' component of `object` (only for determining the number of observations).

Value

The function returns an object of class anova which is displayed using `print.anova`.

Note

A clustering technique could be employed to determine the grouping to be used in cases where there are no replicates. There should at most be ceiling(n/2) clusters as otherwise some observations will not be used in the test. At the other end there need to be more clusters than parameters in the model.

Author(s)

Christian Ritz

References

Neill, J. W. (1988) Testing for lack of fit in nonlinear regression, *Ann. Statist.*, **16**, 733–740

See Also

See also modelFit for details on the lack-of-fit test against an ANOVA model.

Examples

```
### Example with 'drc' object

## Lack-of-fit test against ANOVA
ryegrass.m1 <-drm(rootl~conc, data = ryegrass, fct = LL.4())
modelFit(ryegrass.m1)

## The same test using 'neill.test'
neill.test(ryegrass.m1, ryegrass$conc)

## Generating a grouping
neill.test(ryegrass.m1, method="c-finest")
neill.test(ryegrass.m1, method="finest")
neill.test(ryegrass.m1, method="perc")
```

---

noEffect *Testing if there is a dose effect at all*

Description

A significance test is provided for the comparison of the dose-response model considered and the simple linear regression model with slope 0 (a horizontal regression line corresponding to no dose effect)

Usage

```
noEffect(object)
```

Arguments

object	an object of class 'drc'.

Details

Perhaps useful for screening purposes.

Value

The likelihood ratio test statistic and the corresponding degrees of freedom and p-value are reported.

Author(s)

Christian Ritz

Examples

```
ryegrass.LL.4 <- drm(rootl ~ conc, data = ryegrass, fct = LL.4())

noEffect(ryegrass.LL.4)
# p-value < 0.0001: there is a highly significant dose effect!
```

O.mykiss *Test data from a 21 day fish test*

Description

Test data from a 21 day fish test following the guidelines OECD GL204, using the test organism Rainbow trout *Oncorhynchus mykiss*.

Usage

```
data(O.mykiss)
```

Format

A data frame with 70 observations on the following 2 variables.

`conc` a numeric vector of concentrations (mg/l)

`weight` a numeric vector of wet weights (g)

Details

Weights are measured after 28 days.

Source

Organisation for Economic Co-operation and Development (OECD) (2006) *CURRENT APPROACHES IN THE STATISTICAL ANALYSIS OF ECOTOXICITY DATA: A GUIDANCE TO APPLICATION - ANNEXES*, Paris (p. 65).

References

Organisation for Economic Co-operation and Development (OECD) (2006) *CURRENT APPROACHES IN THE STATISTICAL ANALYSIS OF ECOTOXICITY DATA: A GUIDANCE TO APPLICATION - ANNEXES*, Paris (pp. 80–85).

Examples

```
head(O.mykiss)

## Fitting exponential model
O.mykiss.m1 <- drm(weight ~ conc, data = O.mykiss, fct = EXD.2(), na.action = na.omit)
modelFit(O.mykiss.m1)
summary(O.mykiss.m1)

## Fitting same model with transform-both-sides approach
O.mykiss.m2 <- boxcox(O.mykiss.m1 , method = "anova")
summary(O.mykiss.m2)
# no need for a transformation

## Plotting the fit
plot(O.mykiss.m1, type = "all", xlim = c(0, 500), ylim = c(0,4),
xlab = "Concentration (mg/l)", ylab = "Weight (g)", broken = TRUE)
```

P.promelas *Effect of sodium pentachlorophenate on growth of fathead minnow*

Description

Fathead minnows (*Pimephales promelas*) were exposed to sodium pentachlorophenate concentrations ranging from 32 to 512 micro g/L in a 7-day larval survival and growth test. The average dry weight was measured.

Usage

```
data(P.promelas)
```

Format

A data frame with 24 observations on the following 2 variables.

`conc` a numeric vector of sodium pentachlorophenate concentrations (micro g/L).

`dryweight` a numeric vector dry weights (mg)

Details

The data are analysed in Bruce and Versteeg (1992) using a log-normal dose-response model (using the logarithm with base 10).

Source

Bruce, R. D. and Versteeg, D. J. (1992) A statistical procedure for modeling continuous toxicity data, *Environ. Toxicol. Chem.*, **11**, 1485–1494.

Examples

```
## Model with ED50 on log scale as parameter
p.prom.m1<-drm(dryweight~conc, data=P.promelas, fct=LN.3())

plot(fitted(p.prom.m1), residuals(p.prom.m1))

plot(p.prom.m1, type="all", broken=TRUE, xlim=c(0,1000))
summary(p.prom.m1)
ED(p.prom.m1, c(10,20,50), interval="delta")

## Model with ED50 as parameter
p.prom.m2<-drm(dryweight~conc, data=P.promelas, fct=LN.3(loge=TRUE))
summary(p.prom.m2)
ED(p.prom.m2, c(10,20,50), interval="fls")
```

`plot.drc` *Plotting fitted dose-response curves*

Description

`plot` displays fitted curves and observations in the same plot window, distinguishing between curves by different plot symbols and line types.

Usage

```
  ## S3 method for class 'drc'
plot(x, ..., add = FALSE, level = NULL,
  type = c("average", "all", "bars", "none", "obs", "confidence"),
  broken = FALSE, bp, bcontrol = NULL, conName = NULL, axes = TRUE,
  gridsize = 100, log = "x", xtsty, xttrim = TRUE,
  xt = NULL, xtlab = NULL, xlab, xlim,
  yt = NULL, ytlab = NULL, ylab, ylim,
  cex, cex.axis = 1, col = FALSE, lty, pch,
  legend, legendText, legendPos, cex.legend = 1,
  normal = FALSE, normRef = 1, confidence.level = 0.95)
```

Arguments

`x`	an object of class 'drc'.
`...`	additional graphical arguments. For instance, use `lwd=2` or `lwd=3` to increase the width of plot symbols.
`add`	logical. If TRUE then add to already existing plot.
`level`	vector of character strings. To plot only the curves specified by their names.

`type` a character string specifying how to plot the data. There are currently 5 options: "average" (averages and fitted curve(s); default), "none" (only the fitted curve(s)), "obs" (only the data points), "all" (all data points and fitted curve(s)), "bars" (averages and fitted curve(s) with model-based standard errors (see Details)), and "confidence" (confidence bands for fitted curve(s)).

`broken` logical. If TRUE the x axis is broken provided this axis is logarithmic (using functionality in the CRAN package 'plotrix').

`bp` numeric value specifying the break point below which the dose is zero (the amount of stretching on the dose axis above zero in order to create the visual illusion of a logarithmic scale *including* 0). The default is the base-10 value corresponding to the rounded value of the minimum of the log10 values of all positive dose values. This argument is only working for logarithmic dose axes.

`bcontrol` a list with components `factor`, `style` and `width`. Controlling the appearance of the break (in case `broken` is `TRUE`). The component `factor` is the distance from the control to the break as a multiple of the value of `bp` (default is 2). The component `style` can take the values: `gap`, `slash` or `zigzag`. The component `width` is the width of the break symbol (default is 0.02).

`conName` character string. Name on x axis for dose zero. Default is '"0"'.

`axes` logical indicating whether both axes should be drawn on the plot.

`gridsize` numeric. Number of points in the grid used for plotting the fitted curves.

`log` a character string which contains '"x"' if the x axis is to be logarithmic, '"y"' if the y axis is to be logarithmic and '"xy"' or '"yx"' if both axes are to be logarithmic. The default is "x". The empty string "" yields the original axes.

`xtsty` a character string specifying the dose axis style for arrangement of tick marks. By default ("base10") For a logarithmic axis by default only base 10 tick marks are shown ("base10"). Otherwise sensible equidistantly located tick marks are shown ("standard"), relying on `axTicks`.

`xttrim` logical specifying if the number of tick marks should be trimmed in case too many tick marks are initially determined.

`xt` a numeric vector containing the positions of the tick marks on the x axis.

`xtlab` a vector containing the tick marks on the x axis.

`xlab` an optional label for the x axis.

`xlim` a numeric vector of length two, containing the lower and upper limit for the x axis.

`yt` a numeric vector, containing the positions of the tick marks on the y axis.

`ytlab` a vector containing the tick marks on the y axis.

`ylab` an optional label for the y axis.

`ylim` a numeric vector of length two, containing the lower and upper limit for the y axis.

`cex` numeric or numeric vector specifying the size of plotting symbols and text (see `par` for details).

`cex.axis` numeric value specifying the magnification to be used for axis annotation relative to the current setting of cex.

col	either logical or a vector of colours. If TRUE default colours are used. If FALSE (default) no colours are used.
legend	logical. If TRUE a legend is displayed.
legendText	a character string or vector of character strings specifying the legend text (the position of the upper right corner of the legend box).
legendPos	numeric vector of length 2 giving the position of the legend.
cex.legend	numeric specifying the legend text size.
lty	a numeric vector specifying the line types.
pch	a vector of plotting characters or symbols (see points).
normal	logical. If TRUE the plot of the normalized data and fitted curves are shown (for details see Weimer et al. (2012) for details).
normRef	numeric specifying the reference for the normalization (default is 1).
confidence.level	confidence level for error bars. Defaults to 0.95.

Details

The use of xlim allows changing the range of the x axis, extrapolating the fitted dose-response curves. Note that changing the range on the x axis may also entail a change of the range on the y axis. Sometimes it may be useful to extend the upper limit on the y axis (using ylim) in order to fit a legend into the plot.

See colors for the available colours.

Suitable labels are automatically provided.

The arguments broken and bcontrol rely on the function link{axis.break} with arguments style and brw in the package plotrix.

The model-based standard errors used for the error bars are calculated as the fitted value plus/minus the estimated error times the 1-(alpha/2) quantile in the t distribution with degrees of freedom equal to the residual degrees of freedom for the model (or using a standard normal distribution in case of binomial and poisson data), where alpha=1-confidence.level. The standard errors are obtained using the predict method with the arguments interval = "confidence" and level=confidence.level.

Value

An invisible data frame with the values used for plotting the fitted curves. The first column contains the dose values, and the following columns (one for each curve) contain the fitted response values.

Author(s)

Christian Ritz and Jens C. Streibig. Contributions from Xiaoyan Wang and Greg Warnes.

References

Weimer, M., Jiang, X., Ponta, O., Stanzel, S., Freyberger, A., Kopp-Schneider, A. (2012) The impact of data transformations on concentration-response modeling. *Toxicology Letters*, **213**, 292–298.

Examples

```
## Fitting models to be plotted below
ryegrass.m1 <- drm(rootl~conc, data = ryegrass, fct = LL.4())
ryegrass.m2 <- drm(rootl~conc, data = ryegrass, fct = LL.3())  # lower limit fixed at 0

## Plotting observations and fitted curve for the first model
plot(ryegrass.m1, broken = TRUE)

## Adding fitted curve for the second model (not much difference)
plot(ryegrass.m2, broken = TRUE, add = TRUE, type = "none", col = 2, lty = 2)

## Add confidence region for the first model.
plot(ryegrass.m1, broken = TRUE, type="confidence", add=TRUE)

## Finetuning the axis break
plot(ryegrass.m1, broken = TRUE, bcontrol = list(style = "gap"))
plot(ryegrass.m1, broken = TRUE, bcontrol = list(style = "slash"))
plot(ryegrass.m1, broken = TRUE, bcontrol = list(style = "zigzag"))

## Plot without axes
plot(ryegrass.m1, axes = FALSE)

## Fitting model to be plotted below
spinach.m1 <- drm(SLOPE~DOSE, CURVE, data = spinach, fct = LL.4())

## Plot with no colours
plot(spinach.m1, main = "Different line types (default)")

## Plot with default colours
plot(spinach.m1, col = TRUE, main = "Default colours")

## Plot with specified colours
plot(spinach.m1, col = c(2,6,3,23,56), main = "User-specified colours")

## Plot of curves 1 and 2 only
plot(spinach.m1, level = c(1,2), main = "User-specified curves")

## Plot with symbol of different sizes
plot(spinach.m1, cex = c(1,2,3,4,5), main = "User-specified symbil sizes")

## Plot with confidence regions
plot(spinach.m1, col = TRUE, main = "Confidence Regions", type = "confidence")

## Add points
plot(spinach.m1, col = TRUE, add=TRUE)

## Fitting another model to be plotted below
lettuce.m1 <- drm(weight~conc, data = lettuce, fct = LL.4())

## Using the argument 'bp'. Compare the plots!
par(mfrow = c(2, 2))
```

```
plot(lettuce.m1, main = "bp = default")  # using the default
plot(lettuce.m1, bp = 1e-4, main = "bp = 1e-4")
plot(lettuce.m1, bp = 1e-6, main = "bp = 1e-6")
plot(lettuce.m1, bp = 1e-8, main = "bp = 1e-8")
par(mfrow = c(1,1))

## User-specified position of legend
S.alba.m1 <- drm(DryMatter~Dose, Herbicide, data = S.alba, fct = LL.4())

plot(S.alba.m1)
plot(S.alba.m1, legendPos = c(0.3, 4.8))
```

PR *Expected or predicted response*

Description

The function returns the expected or predicted response for specified dose values.

Usage

```
PR(object, xVec, ...)
```

Arguments

`object`	object of class `drc` obtaining fitting a dose-response model.
`xVec`	numeric vector of dose values.
`...`	additional arguments to be supplied to `predict.drc`. No effect at the moment.

Details

This function is a convenience function for easy access to predicted values.

Value

A numeric vector of predicted values or possibly a matrix of predicted values and corresponding standard errors.

Author(s)

Christian Ritz after a suggestion from Andrew Kniss.

See Also

Predictions can also be obtained using `predict.drc`.

Examples

```
ryegrass.m1 <- drm(ryegrass, fct = LL.4())
PR(ryegrass.m1, c(5, 10))

ryegrass.m2 <- drm(ryegrass, fct = LL2.4())
PR(ryegrass.m2, c(5, 10))

spinach.m1 <- drm(SLOPE~DOSE, CURVE, data=spinach, fct = LL.4())
PR(spinach.m1, c(5, 10))
```

`predict.drc` *Prediction*

Description

Predicted values for models of class 'drc' or class 'mrdrc'.

Usage

```
  ## S3 method for class 'drc'
predict(object, newdata, se.fit = FALSE,
  interval = c("none", "confidence", "prediction"),
  level = 0.95, na.action = na.pass, od = FALSE, vcov. = vcov, ...)

  ## S3 method for class 'mrdrc'
predict(object, newdata, se.fit = FALSE,
  interval = c("none", "confidence", "prediction"),
  level = 0.95, pava = FALSE, ...)
```

Arguments

`object`	an object of class 'drc'.
`newdata`	An optional data frame in which to look for variables with which to predict. If omitted, the fitted values are used.
`se.fit`	logical. If TRUE standard errors are required.
`interval`	character string. Type of interval calculation: "none", "confidence" or "prediction".
`level`	Tolerance/confidence level.
`na.action`	function determining what should be done with missing values in 'newdata'. The default is to predict 'NA'.
`od`	logical. If TRUE adjustment for over-dispersion is used.
`vcov.`	function providing the variance-covariance matrix. `vcov` is the default, but `sandwich` is also an option (for obtaining robust standard errors).

pava	logical. If TRUE the fit is monotoniosed using pool adjacent violators algorithm.
...	further arguments passed to or from other methods.

Details

For the built-in log-logistics and Weibull-type models standard errors and confidence/prediction intervals can be calculated. At the moment it only works for the situations where all observations are assumed to have a common variance.

Value

A matrix with as many rows as there are dose values provided in 'newdata' or in the original dataset (in case 'newdata' is not specified) and columns with fitted, standard errors, lower and upper limits of confidence intervals.

Author(s)

Christian Ritz

See Also

For details are found in the help page for predict.lm.

Examples

```
## Fitting a model
spinach.model1 <- drm(SLOPE~DOSE, CURVE, data = spinach, fct = LL.4())

## Predicting values a dose=2 (with standard errors)
predict(spinach.model1, data.frame(dose=2, CURVE=c("1", "2", "3")), se.fit = TRUE)

## Getting confidence intervals
predict(spinach.model1, data.frame(dose=2, CURVE=c("1", "2", "3")),
interval = "confidence")

## Getting prediction intervals
predict(spinach.model1, data.frame(dose=2, CURVE=c("1", "2", "3")),
interval = "prediction")
```

---

print.drc *Printing key features*

Description

'print' displays brief information on an object of class 'drc'.

Usage

```
  ## S3 method for class 'drc'
print(x, ..., digits = max(3, getOption("digits") - 3))
```

Arguments

x	an object of class 'drc'.
...	additional arguments.
digits	an integer giving the number of digits of the parameter coefficients. Default is 3.

Author(s)

Christian Ritz

Examples

```
## Fitting a four-parameter log-logistic model
ryegrass.m1 <- drm(rootl ~conc, data = ryegrass, fct = LL.4())

## Displaying the model fit
print(ryegrass.m1)
ryegrass.m1  # gives the same output as the previous line
```

---

print.summary.drc *Printing summary of non-linear model fits*

Description

This method produces formatted output of the summary statistics: parameter estimates, estimated standard errors, z-test statistics and corresponding p-values.

Usage

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

Arguments

x	an object of class 'drc'.
...	additional arguments.

Value

The object (argument x) is returned invisibly.

Author(s)

Christian Ritz

Examples

```
ryegrass.m1 <- drm(rootl~conc, data=ryegrass, fct= LL.4())

summary(ryegrass.m1)
```

---

rdrm *Simulating a dose-response curve*

Description

Simulation of a dose-response curve with user-specified dose values and error distribution.

Usage

```
rdrm(nosim, fct, mpar, xerror, xpar = 1, yerror = "rnorm", ypar = c(0, 1),
onlyY = FALSE)
```

Arguments

nosim	numeric. The number of simulated curves to be returned.
fct	list. Any built-in function in the package *drc* or a list with similar components.
mpar	numeric. The model parameters to be supplied to fct.
xerror	numeric or character. The distribution for the dose values.
xpar	numeric vector supplying the parameter values defining the distribution for the dose values. If xerror is a distribution then remember that the number of dose values also is part of this argument (the first argument).
yerror	numeric or character. The error distribution for the response values.
ypar	numeric vector supplying the parameter values defining the error distribution for the response values.
onlyY	logical. If TRUE then only the response values are returned (useful in simulations). Otherwise both dose values and response values (and for binomial data also the weights) are returned.

Details

The distribution for the dose values can either be a fixed set of dose values (a numeric vector) used repeatedly for creating all curves or be a distribution specified as a character string resulting in varying dose values from curve to curve.

The error distribution for the response values can be any continuous distribution like `rnorm` or `rgamma`. Alternatively it can be the binomial distribution `rbinom`.

Value

A list with up to 3 components (depending on the value of the `onlyY` argument).

Author(s)

Christian Ritz

References

~put references to the literature/web site here ~

Examples

```
## Simulating normally distributed dose-response data

## Model fit to simulate from
ryegrass.m1 <- drm(rootl~conc, data = ryegrass, fct = LL.4())

## 10 random dose-response curves based on the model fit
sim10a <- rdrm(10, LL.4(), coef(ryegrass.m1), xerror = ryegrass$conc)
sim10a


## Simulating binomial dose-response data

## Model fit to simulate from
deguelin.m1 <- drm(r/n~dose, weights=n, data=deguelin, fct=LL.2(), type="binomial")

## 10 random dose-response curves
sim10b <- rdrm(10, LL.2(), coef(deguelin.m1), deguelin$dose, yerror="rbinom", ypar=deguelin$n)
sim10b
```

`residuals.drc` *Extracting residuals from the fitted dose-response model*

Description

'residuals' extracts different types of residuals from an object of class 'drc'.

Usage

```
  ## S3 method for class 'drc'
residuals(object, typeRes = c("working", "standardised", "studentised"),
  trScale = TRUE, ...)
```

Arguments

object	an object of class 'drc'.
typeRes	character string specifying the type of residual to be returned: raw/working residuals, residuals standardised using the estimated residual standard error, or studentised residuals based on the H matrix of partial derivatives of the model function.
trScale	logical value indicating whether or not to return residuals on the transformed scale (in case a Box-Cox transformation was applied).
...	additional arguments.

Details

Standardised residuals are the raw residuals divided by a scale estimate (if available).

Studentised residuals are obtained by dividing by a scale estimate and in addition a correction factor (square root of 1 minus h with h is a diagonal element in the hat matrix).

Value

The raw (also called working) residuals or some kind of scaled residuals extracted from 'object'.

Note

The 'standardised' residuals are available for least squares estimation with or without Box-Cox transformation or variance as a power of the mean.

Author(s)

Christian Ritz

Examples

```
## Fitting a four-parameter log-logistic model
ryegrass.m1 <- drm(rootl ~conc, data = ryegrass, fct = LL.4())

## Displaying the residual plot (raw residuals)
plot(fitted(ryegrass.m1), residuals(ryegrass.m1))

## Using the standardised residuals
plot(fitted(ryegrass.m1), residuals(ryegrass.m1, typeRes = "standard"))
```

```
## Overlayering the studentised residuals ... not much of a difference
points(fitted(ryegrass.m1), residuals(ryegrass.m1, typeRes = "student"), col = 2)
```

RScompetition *Competition between two biotypes*

Description

To assess the competitive ability between two biotypes of *Lolium rigidum*, one resistant to glyphosate and the other a sensitive wild type, the density of resistant and sensitive biotypes was counted after germination.

Usage

```
data(RScompetition)
```

Format

A data frame with 49 observations on the following 3 variables.

`z` a numeric vector with densities of the resistant biotype (plants/m2)

`x` a numeric vector with densities of the sensitive biotype (plants/m2)

`biomass` a numeric vector of biomass weight (g/plant)

Details

A hyperbolic model (Jensen, 1993) is describing the data reasonably well.

Source

The dataset is from Pedersen et al (2007).

References

Jensen, J. E. (1993) Fitness of herbicide-resistant weed biotypes described by competition models, *Proceedings of the 8th EWRS Symposium, 14-16 June, Braunschweig, Germany*, **1**, 25–32.

Pedersen, B. P. and Neve, P. and Andreasen, C. and Powles, S. (2007) Ecological fitness of a glyphosate resistant *Lolium rigidum* population: Growth and seed production along a competition gradient, *Basic and Applied Ecology*, **8**, 258–268.

ryegrass *Effect of ferulic acid on growth of ryegrass*

Description

A single dose-response curve.

Usage

```
data(ryegrass)
```

Format

A data frame with 24 observations on the following 2 variables.

rootl a numeric vector of root lengths

conc a numeric vector of concentrations of ferulic acid

Details

The data are part of a study to investigate the joint action of phenolic acids on root growth inhibition of perennial ryegrass (*Lolium perenne L*).

`conc` is the concentration of ferulic acid is in mM, and `rootl` is the root length of perennial ryegrass measured in cm.

Source

Inderjit and J. C. Streibig, and M. Olofsdotter (2002) Joint action of phenolic acid mixtures and its significance in allelopathy research, *Physiologia Plantarum*, **114**, 422–428, 2002.

Examples

```
## Displaying the data set
ryegrass

## Fitting a four-parameter Weibull model (type 2)
ryegrass.m1 <- drm(rootl ~ conc, data = ryegrass, fct = W2.4())

## Displaying a summary of the model fit
summary(ryegrass.m1)

## Plotting the fitted curve together with the original data
plot(ryegrass.m1)

## Fitting a four-parameter Weibull model (type 1)
ryegrass.m2 <- drm(rootl ~ conc, data = ryegrass, fct = W1.4())
plot(ryegrass.m2)
```

```
## Fitting a four-parameter log-logistic model
## with user-defined parameter names
ryegrass.m3 <- drm(rootl ~ conc, data = ryegrass,
fct = LL.4(names = c("Slope", "Lower Limit", "Upper Limit", "ED50")))
summary(ryegrass.m3)

## Comparing log-logistic and Weibull models
## (Figure 2 in Ritz (2009))
ryegrass.m0 <- drm(rootl ~ conc, data = ryegrass, fct = LL.4())
ryegrass.m1 <- drm(rootl ~ conc, data = ryegrass, fct = W1.4())
ryegrass.m2 <- drm(rootl ~ conc, data = ryegrass, fct = W2.4())

plot(ryegrass.m0, broken=TRUE, xlab="Dose (mM)", ylab="Root length (cm)", lwd=2,
cex=1.2, cex.axis=1.2, cex.lab=1.2)
plot(ryegrass.m1, add=TRUE, broken=TRUE, lty=2, lwd=2)
plot(ryegrass.m2, add=TRUE, broken=TRUE, lty=3, lwd=2)

arrows(3, 7.5, 1.4, 7.5, 0.15, lwd=2)
text(3,7.5, "Weibull-2", pos=4, cex=1.2)

arrows(2.5, 0.9, 5.7, 0.9, 0.15, lwd=2)
text(3,0.9, "Weibull-1", pos=2, cex=1.2)
```

`S.alba` *Potency of two herbicides*

Description

Data are from an experiment, comparing the potency of the two herbicides glyphosate and bentazone in white mustard *Sinapis alba*.

Usage

```
data(S.alba)
```

Format

A data frame with 68 observations on the following 3 variables.

`Dose` a numeric vector containing the dose in g/ha.

`Herbicide` a factor with levels `Bentazone` `Glyphosate` (the two herbicides applied).

`DryMatter` a numeric vector containing the response (dry matter in g/pot).

Details

The lower and upper limits for the two herbicides can be assumed identical, whereas slopes and ED50 values are different (in the log-logistic model).

Source

Christensen, M. G. and Teicher, H. B., and Streibig, J. C. (2003) Linking fluorescence induction curve and biomass in herbicide screening, *Pest Management Science*, **59**, 1303–1310.

See Also

See the examples sections for drm and EDcomp.

Examples

```
## Fitting a log-logistic model with
##  common lower and upper limits
S.alba.LL.4.1 <- drm(DryMatter~Dose, Herbicide, data=S.alba, fct = LL.4(),
pmodels=data.frame(Herbicide,1,1,Herbicide))
summary(S.alba.LL.4.1)

## Applying the optimal transform-both-sides Box-Cox transformation
## (using the initial model fit)
S.alba.LL.4.2 <- boxcox(S.alba.LL.4.1, method = "anova")
summary(S.alba.LL.4.2)

## Plotting fitted regression curves together with the data
plot(S.alba.LL.4.2)
```

S.capricornutum *Effect of cadmium on growth of green alga*

Description

Green alga (*Selenastrum capricornutum*) was exposed to cadmium chloride concentrations ranging from 5 to 80 micro g/L in geometric progression in 4-day population growth test.

Usage

```
data(S.capricornutum)
```

Format

A data frame with 18 observations on the following 2 variables.

`conc` a numeric vector of cadmium chloride concentrations (micro g/L)

`count` a numeric vector of algal counts (10000 x cells /ml)

Details

The data are analysed in Bruce and Versteeg (1992) using a log-normal dose-response model (using the logarithm with base 10).

Source

Bruce, R. D. and Versteeg, D. J. (1992) A statistical procedure for modeling continuous toxicity data, *Environ. Toxicol. Chem.*, **11**, 1485–1494.

Examples

```
## Fitting 3-parameter log-normal model
s.cap.m1 <- drm(count ~ conc, data = S.capricornutum, fct = LN.3())

## Residual plot
plot(fitted(s.cap.m1), residuals(s.cap.m1))

## Fitting model with transform-both-sides approach
s.cap.m2 <- boxcox(s.cap.m1, method = "anova")
summary(s.cap.m2)

## Residual plot after transformation (looks better)
plot(fitted(s.cap.m2), residuals(s.cap.m2))

## Calculating ED values on log scale
ED(s.cap.m2, c(10, 20, 50), interval="delta")

## Fitting model with ED50 as parameter
## (for comparison)
s.cap.m3 <- drm(count ~ conc, data = S.capricornutum, fct = LN.3(loge=TRUE))
s.cap.m4 <- boxcox(s.cap.m3, method = "anova")
summary(s.cap.m4)
ED(s.cap.m4, c(10, 20, 50), interval = "fls")
```

---

searchdrc *Searching through a range of initial parameter values to obtain convergence*

Description

'searchdrc' provides a facility for searching through a range of parameter values (one-dimensional) in order to obtain convergence of the estimation procedure.

Usage

```
searchdrc(object, which, range, len = 50)
```

Arguments

object	an object of class 'drc'. The object can be from a model that could not fitted.
which	a character string containing the parameter name
range	a numeric vector of length 2 specifying the interval endpoints for the range.
len	numeric. The number of points in the interval.

Details

The function goes through the range with increments such that in total at most `len` sets of parameter values are used as initial values for the estimation procedure. You would need to identify the parameter which is most likely to cause problems for the estimation procedure.

Value

An object of class 'drc'.

Author(s)

Christian Ritz

Examples

```
## No example yet
```

`secalonic` *Root length measurements*

Description

Data stem from an experiment assessing the inhibitory effect of secalonic acids on plant growth.

Usage

```
data(secalonic)
```

Format

A data frame with 7 observations on the following 2 variables:

`dose` a numeric vector containing dose values (mM)

`rootl` a numeric vector containing root lengths (cm)

Details

For each dose the root length is an average three measurements.

Source

Gong, X. and Zeng, R. and Luo, S. and Yong, C. and Zheng, Q. (2004) Two new secalonic acids from *Aspergillus Japonicus* and their allelopathic effects on higher plants, *Proceedings of International Symposium on Allelopathy Research and Application, 27-29 April, Shanshui, Guangdong, China (Editors: R. Zeng and S. Luo)*, 209–217.

Ritz, C (2009) Towards a unified approach to dose-response modeling in ecotoxicology *To appear in Environ Toxicol Chem*.

Examples

```
## Fitting a four-parameter log-logistic model
secalonic.m1 <- drm(rootl ~ dose, data = secalonic, fct = LL.4())
summary(secalonic.m1)

## Fitting a three-parameter log-logistic model
##  lower limit fixed at 0
secalonic.m2 <- drm(rootl ~ dose, data = secalonic, fct = LL.3())
summary(secalonic.m1)

## Comparing logistic and log-logistic models
## (Figure 1 in Ritz (2009))
secalonic.LL4 <- drm(rootl ~ dose, data = secalonic, fct = LL.4())
secalonic.L4 <- drm(rootl ~ dose, data = secalonic, fct = L.4())

plot(secalonic.LL4, broken=TRUE, ylim=c(0,7), xlab="Dose (mM)", ylab="Root length (cm)",
cex=1.2, cex.axis=1.2, cex.lab=1.2, lwd=2)

plot(secalonic.L4, broken=TRUE, ylim=c(0,7), add=TRUE, type="none", lty=2, lwd=2)

abline(h=coef(secalonic.L4)[3], lty=3, lwd=2)
```

---

selenium *Data from toxicology experiments with selenium*

Description

Comparison of toxicity of four types of selenium by means of dose-response analysis

Usage

```
data(selenium)
```

Format

A data frame with 25 observations on the following 4 variables.

`type` a numeric vector indicating the form of selenium applied

`conc` a numeric vector of (total) selenium concentrations

`total` a numeric vector containing the total number of flies

`dead` a numeric vector containing the number of dead flies

Details

The experiment is described in more details by Jeske et al. (2009).

Source

Jeske, D. R., Xu, H. K., Blessinger, T., Jensen, P. and Trumble, J. (2009) Testing for the Equality of EC50 Values in the Presence of Unequal Slopes With Application to Toxicity of Selenium Types, *Journal of Agricultural, Biological, and Environmental Statistics*, **14**, 469–483

Examples

```
## Analysis similar to what is proposed in Jeske et al (2009)
##  but simply using existing functionality in "drc"

## Fitting the two-parameter log-logistic model with unequal ED50 and slope
sel.m1 <- drm(dead/total~conc, type, weights=total, data=selenium, fct=LL.2(),
type="binomial")
#sel.m1b <- drm(dead/total~conc, type, weights=total, data=selenium, fct=LN.2(),
# type="binomial", start=c(1,1,1,1,50,50,50,50))
plot(sel.m1, ylim = c(0, 1.3))
summary(sel.m1)

## Testing for equality of slopes
sel.m2 <- drm(dead/total~conc, type, weights=total, data=selenium, fct=LL.2(),
type="binomial", pmodels=list(~1, ~factor(type)-1))
sel.m2b <- drm(dead/total~conc, type, weights=total, data=selenium, fct=LN.2(),
type="binomial", pmodels=list(~1, ~factor(type)-1))
plot(sel.m2, ylim = c(0, 1.3))
summary(sel.m2)
anova(sel.m2, sel.m1)  # 48.654
#anova(sel.m2b, sel.m1b)
# close to the value 48.46 reported in the paper

## Testing for equality of ED50
sel.m3<-drm(dead/total~conc, type, weights=total, data=selenium, fct=LL.2(),
type="binomial", pmodels=list(~factor(type)-1, ~1))
#sel.m3b<-drm(dead/total~conc, type, weights=total, data=selenium, fct=LN.2(),
# type="binomial", pmodels=list(~factor(type)-1, ~1), start=c(1,1,1,1,50))
plot(sel.m3, ylim = c(0, 1.3))
summary(sel.m3)

anova(sel.m3, sel.m1)  # 123.56
#anova(sel.m3b, sel.m1b)
# not too far from the value 138.45 reported in the paper
# (note that the estimation procedure is not exactly the same)
# (and we use the log-logistic model instead of the log-normal model)
```

---

simDR *Simulating ED values under various scenarios*

Description

Simulating ED values for a given model and given dose values.

Usage

```
simDR(mpar, sigma, fct, noSim = 1000, conc, edVec = c(10, 50), seedVal = 20070723)
```

Arguments

mpar	numeric vector of model parameters
sigma	numeric specifying the residual standard deviation
fct	list supplying the chosen mean function
conc	numeric vector of concentration/dose values
edVec	numeric vector of ED values to estimate in each simulation
noSim	numeric giving the number of simulations
seedVal	numeric giving the seed used to initiate the random number generator

Details

The arguments mpar and sigma are typically obtained from a previous model fit.

Only dose-response models assuming normally distributed errors can be used.

Value

A list of matrices with as many components as there are chosen ED values. The entries in the matrices are empirical standard deviations of the estimated ED values. Row-wise from top to bottom more and more concentration/dose values are included in the simulations; top row starting with 5 concentrations. The number of replicates increases column by column from left to right.

The list is returned invisbly as the matrices also are displayed.

Author(s)

Christian Ritz

Examples

```
ryegrass.m1 <- drm(ryegrass, fct=LL.4())

simDR(coef(ryegrass.m1), sqrt(summary(ryegrass.m1)$resVar), LL.4(), 2,
c(1.88, 3.75, 7.50, 0.94, 15, 0.47, 30, 0.23, 60), seedVal = 200710291)
```

spinach *Inhibition of photosynthesis*

Description

Data from an experiment investigating the inhibition of photosynthesis in response to two synthetic photosystem II inhibitors, the herbicides diuron and bentazon. More specifically, the effect of oxygen consumption of thylakoid membranes (chloroplasts) from spinach was measured after incubation with the synthetic inhibitors in 5 assays, resulting in 5 dose-response curves.

Usage

```
data(spinach)
```

Format

A data frame with 105 observations on the following four variables:

CURVE a numeric vector specifying the assay or curve (a total of 5 independent assays where used in this experiment).

HERBICIDE a character vector specifying the herbicide applied: bentazon or diuron.

DOSE a numeric vector giving the herbicide concentration in muMol.

SLOPE a numeric vector with the measured response: oxygen consumption of thylakoid membranes.

Details

The experiment is described in more details by Streibig (1998).

Source

Streibig, J. C. (1998) Joint action of natural and synthetic photosystem II inhibitors, *Pesticide Science*, **55**, 137–146.

Examples

```
## Displaying the first rows in the dataset
head(spinach)  # displaying first 6 rows in the data set
```

summary.drc *Summarising non-linear model fits*

Description

'summary' compiles a comprehensive summary for objects of class 'drc'.

Usage

```
  ## S3 method for class 'drc'
summary(object, od = FALSE, pool = TRUE, ...)
```

Arguments

object	an object of class 'drc'.
od	logical. If TRUE adjustment for over-dispersion is used.
pool	logical. If TRUE curves are pooled. Otherwise they are not. This argument only works for models with independently fitted curves as specified in drm.
...	additional arguments.

Value

A list of summary statistics that includes parameter estimates and estimated standard errors.

Note

Examples on usage are for instance found in the help pages of ryegrass and secalonic.

Author(s)

Christian Ritz

terbuthylazin *The effect of terbuthylazin on growth rate*

Description

Test on the effect of terbuthylazin on *Lemna minor*, performed on an aseptic culture according to the OECD-guidelines.

Usage

```
data(terbuthylazin)
```

Format

A data frame with 30 observations on the following 2 variables.

dose a numeric vector of dose values.

rgr a numeric vector of relative growth rates.

Details

Dose is

$$\mu l^{-1}$$

and rgr is the relative growth rate of *Lemna*.

Source

Cedergreen N. (2004). Unpublished bioassay data.

Examples

```
## displaying first 6 rows of the data set
head(terbuthylazin)

## Fitting log-logistic model
terbuthylazin.m1 <- drm(rgr~dose, data = terbuthylazin, fct = LL.4())
summary(terbuthylazin.m1)

## Fitting log-logistic model
##  with Box-Cox transformation
terbuthylazin.m2 <- boxcox(terbuthylazin.m1, method = "anova")
summary(terbuthylazin.m2)
```

---

twophase *Two-phase dose-response model*

Description

The two-phase dose-response model is a combination of log-logistic models that should be useful for describing more complex dose-response patterns.

Usage

```
twophase(fixed = c(NA, NA, NA, NA, NA, NA, NA),
names = c("b1", "c1", "d1", "e1", "b2", "d2", "e2"), fctName, fctText)
```

Arguments

fixed	numeric vector specifying which parameters are fixed and at what value they are fixed. NAs are used for parameters that are not fixed.
names	a vector of character strings giving the names of the parameters (should not contain ":"). The default is reasonable (see under 'Usage').
fctName	optional character string used internally by convenience functions.
fctText	optional character string used internally by convenience functions.

Details

Following Groot *et al* (1996) the two-phase model function is defined as follows

$$f(x) = c + \frac{d1 - c}{1 + \exp(b1(\log(x) - \log(e1)))} + \frac{d2}{1 + \exp(b2(\log(x) - \log(e2)))}$$

For each of the two phases, the parameters have the same interpretation as in the ordinary log-logistic model.

Value

The value returned is a list containing the nonlinear function, the self starter function and the parameter names.

Author(s)

Christian Ritz

References

Groot, J. C. J., Cone, J. W., Williams, B. A., Debersaques, F. M. A., Lantinga, E. A. (1996) Multiphasic analysis of gas production kinetics for in vitro fermentation of ruminant feeds, *Animal Feed Science Technology*, **64**, 77–89.

See Also

The basic component in the two-phase model is the log-logistic model `llogistic`.

---

update.drc *Updating and re-fitting a model*

Description

'update' updates and re-fits a model on the basis of an object of class 'drc'.

Usage

```
  ## S3 method for class 'drc'
update(object, ..., evaluate = TRUE)
```

Arguments

object	an object of class 'drc'.
...	arguments to alter in object.
evaluate	logical. If TRUE model is re-fit; otherwise an unevaluated call is returned.

Value

An object of class 'drc'.

Author(s)

Christian Ritz

Examples

```
## Fitting a four-parameter Weibull model
model1 <- drm(ryegrass, fct = W1.4())

## Updating 'model1' by fitting a three-parameter Weibull model instead
model2 <- update(model1, fct = W1.3())
anova(model2, model1)
```

---

ursa

Model function for the universal response surface approach (URSA) for the quantitative assessment of drug interaction

---

Description

URSA provides a parametric approach for modelling the joint action of several agents. The model allows quantification of synergistic effects through a single parameter.

Usage

```
ursa(fixed = rep(NA, 7), names = c("b1", "b2", "c", "d", "e1", "e1", "f"),
ssfct = NULL)
```

Arguments

fixed	numeric vector. Specifies which parameters are fixed and at what value they are fixed. NAs for parameter that are not fixed.
names	a vector of character strings giving the names of the parameters. The default is reasonable.
ssfct	a self starter function to be used (optional).

Details

The model function is defined implicitly through an appropriate equation. More details are provided by Greco et al (1990, 1995).

Value

A list containing the nonlinear function, the self starter function, and the parameter names.

Author(s)

Christian Ritz after an idea by Hugo Ceulemans.

References

Greco, W. R. and Park H. S. and Rustum, Y. M. (1990) Application of a New Approach for the Quantitation of Drug Synergism to the Combination of cis-Diamminedichloroplatinum and 1-beta-D-Arabinofuranosylcytosine, *Cancer Research*, **50**, 5318–5327.

Greco, W. R. Bravo, G. and Parsons, J. C. (1995) The Search for Synergy: A Critical Review from a Response Surface Perspective, *Pharmacological Reviews*, **47**, Issue 2, 331–385.

See Also

Other models for fitting mixture data are the Hewlett and Voelund models `mixture`.

Examples

```
## Here is the complete statistical analysis of the data
##  from Greco et al. (1995) by means of the URSA model
if (FALSE)
{
d1 <- c(0, 0, 0, 0, 0, 0, 0, 0, 2, 5, 10, 20, 50, 2, 2, 2,
2, 2, 5, 5, 5, 5, 5, 10, 10, 10, 10, 10, 20, 20, 20, 20,
20, 50, 50, 50, 50, 50)

d2 <- c(0, 0, 0, 0.2, 0.5, 1, 2, 5, 0, 0, 0, 0, 0, 0.2,
0.5, 1, 2, 5, 0.2, 0.5,  1, 2, 5, 0.2, 0.5, 1, 2, 5, 0.2,
0.5, 1, 2, 5, 0.2, 0.5, 1, 2, 5)

effect <- c(106.00, 99.20, 115.00, 79.20, 70.10, 49.00,
21.00, 3.83, 74.20, 71.50,48.10, 30.90, 16.30, 76.30,
48.80, 44.50, 15.50, 3.21, 56.70, 47.50, 26.80, 16.90,
```

```
3.25, 46.70, 35.60, 21.50, 11.10, 2.94, 24.80, 21.60,
17.30, 7.78, 1.84, 13.60, 11.10, 6.43, 3.34, 0.89)

greco <- data.frame(d1, d2, effect)

greco.m1 <- drm(effect ~ d1 + d2, data = greco, fct = ursa(fixed = c(NA, NA, 0, NA, NA, NA, NA)))

plot(fitted(greco.m1), residuals(greco.m1)) # wedge-shaped

summary(greco.m1)

## Transform-both-sides approach using a logarithm transformation
greco.m2 <- drm(effect ~ d1 + d2, data = greco, fct = ursa(fixed = c(NA, NA, 0, NA, NA, NA, NA)),
bcVal = 0, control = drmc(relTol = 1e-12))

plot(fitted(greco.m2), residuals(greco.m2))  # looks okay

summary(greco.m2)
# close to the estimates reported by Greco et al. (1995)
}
```

---

vcov.drc *Calculating variance-covariance matrix for objects of class 'drc'*

Description

'vcov' returns the estimated variance-covariance matrix for the parameters in the non-linear function.

Usage

```
  ## S3 method for class 'drc'
vcov(object, ..., corr = FALSE, od = FALSE, pool = TRUE, unscaled = FALSE)
```

Arguments

object	an object of class 'drc'.
...	additional arguments.
corr	logical. If TRUE a correlation matrix is returned.
od	logical. If TRUE adjustment for over-dispersion is used. This argument only makes a difference for binomial data.
pool	logical. If TRUE curves are pooled. Otherwise they are not. This argument only works for models with independently fitted curves as specified in drm.
unscaled	logical. If TRUE the unscaled variance-covariance is returned. This argument only makes a difference for continuous data.

Value

A matrix of estimated variances and covariances.

Author(s)

Christian Ritz

Examples

```
## Fitting a four-parameter log-logistic model
ryegrass.m1 <- drm(rootl ~ conc, data = ryegrass, fct = LL.4())
vcov(ryegrass.m1)
vcov(ryegrass.m1, corr = TRUE)
```

`vinclozolin` *Vinclozolin from AR in vitro assay*

Description

Dose-response experiment with vinclozolin in an AR reporter gene assay

Usage

```
data(vinclozolin)
```

Format

A data frame with 53 observations on the following 3 variables.

`exper` a factor with levels `10509` `10821` `10828` `10904` `11023` `11106`

`conc` a numeric vector of concentrations of vinclozolin

`effect` a numeric vector of luminescense effects

Details

The basic dose-response experiment was repeated 6 times on different days. Chinese Hamster Ovary cells were exposed to various concentrations of vinclozolin for 22 hours and the resulting luminescense effects were recorded.

Data are part of mixture experiment reported in Nellemann *et al* (2003).

Source

Nellemann C., Dalgaard M., Lam H.R. and Vinggaard A.M. (2003) The combined effects of vinclozolin and procymidone do not deviate from expected additivity *in vitro* and *in vivo*, *Toxicological Sciences*, **71**, 251–262.

Examples

```
vinclozolin.m1 <- drm(effect~conc, exper, data=vinclozolin, fct = LL.3())
plot(vinclozolin.m1, xlim=c(0,50), ylim=c(0,2800), conLevel=1e-4)

vinclozolin.m2 <- drm(effect~conc, data=vinclozolin, fct = LL.3())
plot(vinclozolin.m2, xlim=c(0,50), conLevel=1e-4, add=TRUE, type="none", col="red")

## Are the ED50 values indetical across experiments?
vinclozolin.m3 <- update(vinclozolin.m1, pmodels=data.frame(exper, exper, 1))
anova(vinclozolin.m3, vinclozolin.m1)  # No!
```

W1.2 *The two-parameter Weibull functions*

Description

'W1.2' is the two-parameter Weibull function where the lower limit is fixed at 0 and the upper limit is fixed at 1, mostly suitable for binomial/quantal responses.

Usage

```
W1.2(upper = 1, fixed = c(NA, NA), names = c("b", "e"), ...)

W2.2(upper = 1, fixed = c(NA, NA), names = c("b", "e"), ...)
```

Arguments

`upper`	numeric value. The fixed, upper limit in the model. Default is 1.
`fixed`	numeric vector. Specifies which parameters are fixed and at what value they are fixed. NAs for parameter that are not fixed.
`names`	a vector of character strings giving the names of the parameters. The default is reasonable.
`...`	additional arguments to be passed from the convenience functions.

Details

The two-parameter Weibull model is given by the expression

$$f(x) = \exp(-\exp(b(\log(x) - e))).$$

The function is asymmetric about the inflection point, that is the parameter $\exp(e)$.

Value

See `weibull1`.

Note

This function is for use with the function drm.

Author(s)

Christian Ritz

See Also

Related functions are W1.3, W1.4, weibull1 and weibull2.

Examples

```
## Fitting a two-parameter Weibull model
earthworms.m1 <- drm(number/total~dose, weights = total,
data = earthworms, fct = W1.2(), type = "binomial")

summary(earthworms.m1)
```

W1.3 *The three-parameter Weibull functions*

Description

'W1.3' and W2.3 provide the three-parameter Weibull function, self starter function and names of the parameters.

'W1.3u' and 'W2.3u' provide three-parameter Weibull function where the upper limit is equal to 1, mainly for use with binomial/quantal response.

Usage

```
W1.3(fixed = c(NA, NA, NA), names = c("b", "d", "e"), ...)

W2.3(fixed = c(NA, NA, NA), names = c("b", "d", "e"), ...)

W2x.3(fixed = c(NA, NA, NA), names = c("d", "e", "t0"), ...)

W1.3u(upper = 1, fixed = c(NA, NA, NA), names = c("b", "c", "e"), ...)

W2.3u(upper = 1, fixed = c(NA, NA, NA), names = c("b", "c", "e"), ...)
```

Arguments

upper	numeric value. The fixed, upper limit in the model. Default is 1.
fixed	numeric vector. Specifies which parameters are fixed and at what value they are fixed. NAs for parameter that are not fixed.
names	a vector of character strings giving the names of the parameters. The default is reasonable.
...	additional arguments to be passed from the convenience functions.

Details

The three-parameter Weibull model is given by the expression

$$f(x) = 0 + (d - 0)\exp(-\exp(b(\log(x) - e))).$$

The function is asymmetric about the inflection point, that is the parameter $\exp(e)$.

The three-parameter Weibull model with upper limit 1 is given by the expression

$$f(x) = 0 + (1 - 0)\exp(-\exp(b(\log(x) - e))).$$

Value

See weibull1.

Note

This function is for use with the function drm.

Author(s)

Christian Ritz

See Also

Related functions are W1.4 and weibull1.

Examples

```
## Fitting a three-parameter Weibull model
ryegrass.m1 <- drm(rootl ~ conc, data = ryegrass, fct = W1.3())
ryegrass.m1
```

`W1.4` *The four-parameter Weibull functions*

Description

'W1.4' and 'W2.4' provide the four-parameter Weibull functions, self starter function and names of the parameters.

Usage

```
W1.4(fixed = c(NA, NA, NA, NA), names = c("b", "c", "d", "e"), ...)

W2.4(fixed = c(NA, NA, NA, NA), names = c("b", "c", "d", "e"), ...)
```

Arguments

`fixed`	numeric vector. Specifies which parameters are fixed and at what value they are fixed. NAs for parameter that are not fixed.
`names`	a vector of character strings giving the names of the parameters. The default is reasonable.
`...`	additional arguments to be passed from the convenience functions.

Details

The equations for the mean functions are given at `weibull1`.

Value

See `weibull1`.

Note

This function is for use with the model fitting function `drm`.

Author(s)

Christian Ritz

References

Seber, G. A. F. and Wild, C. J (1989) *Nonlinear Regression*, New York: Wiley \& Sons (pp. 330–331).

Ritz, C (2009) Towards a unified approach to dose-response modeling in ecotoxicology *To appear in Environ Toxicol Chem*.

See Also

Setting $c = 0$ yields W1.3. A more flexible function, allowing fixing or constraining parameters, is available through weibull1.

Examples

```
## Fitting a four-parameter Weibull (type 1) model
terbuthylazin.m1 <- drm(rgr~dose, data = terbuthylazin, fct = W1.4())
summary(terbuthylazin.m1)

## Fitting a first-order multistage model
## to data from BMDS by EPA
## (Figure 3 in Ritz (2009))
bmds.ex1 <- data.frame(ad.dose=c(0,50,100), dose=c(0, 2.83, 5.67),
num=c(6,10,19), total=c(50,49,50))

bmds.ex1.m1<-drm(num/total~dose, weights=total, data=bmds.ex1,
fct=W2.4(fixed=c(1,NA,1,NA)), type="binomial")

modelFit(bmds.ex1.m1)  # same as in BMDS

summary(bmds.ex1.m1)  # same background estimate as in BMDS

logLik(bmds.ex1.m1)

## BMD estimate identical to BMDS result
## BMDL estimate differs from BMDS result (different method)
ED(bmds.ex1.m1, 10, ci="delta")

## Better fit

bmds.ex1.m2<-drm(num/total~dose, weights=total, data=bmds.ex1,
fct=W1.4(fixed=c(-1,NA,1,NA)), type="binomial")
modelFit(bmds.ex1.m2)
summary(bmds.ex1.m2)

ED(bmds.ex1.m2, 50, ci = "delta")

## Creating Figure 3 in Ritz (2009)
bmds.ex1.m3 <- drm(num/total~dose, weights=total, data=bmds.ex1,
fct=LL.4(fixed=c(-1,NA,1,NA)), type="binomial")

plot(bmds.ex1.m1, ylim = c(0.05, 0.4), log = "", lty = 3, lwd = 2,
xlab = "Dose (mg/kg/day)", ylab = "",
cex=1.2, cex.axis=1.2, cex.lab=1.2)

mtext("Tumor incidence", 2, line=4, cex=1.2)  # tailored y axis label

plot(bmds.ex1.m2, ylim = c(0.05, 0.4), log = "", add = TRUE, lty = 2, lwd = 2)

plot(bmds.ex1.m3, ylim = c(0.05, 0.4), log = "", add = TRUE, lty = 1, lwd = 2)
```

```
arrows(2.6 , 0.14, 2, 0.14, 0.15, lwd=2)
text(2.5, 0.14, "Weibull-1", pos=4, cex=1.2)
```

`weibull1` *Weibull model functions*

Description

'weibull' and 'weibull2' provide a very general way of specifying Weibull dose response functions, under various constraints on the parameters.

Usage

```
weibull1(fixed = c(NA, NA, NA, NA),
         names = c("b", "c", "d", "e"),
         method = c("1", "2", "3", "4"),
         ssfct = NULL,
         fctName, fctText)

weibull2(fixed = c(NA, NA, NA, NA),
         names = c("b", "c", "d", "e"),
         method = c("1", "2", "3", "4"),
         ssfct = NULL,
         fctName, fctText)

weibull2x(fixed = rep(NA, 5),
         names = c("b", "c", "d", "e", "t0"),
         method = c("1", "2", "3", "4"),
         ssfct = NULL,
         fctName, fctText)
```

Arguments

`fixed`	numeric vector. Specifies which parameters are fixed and at what value they are fixed. NAs for parameter that are not fixed.
`names`	a vector of character strings giving the names of the parameters (should not contain ":"). The default is reasonable (see under 'Usage'). The order of the parameters is: b, c, d, e (see under 'Details').
`method`	character string indicating the self starter function to use.
`ssfct`	a self starter function to be used.
`fctName`	optional character string used internally by convenience functions.
`fctText`	optional character string used internally by convenience functions.

Details

As pointed out in Seber and Wild (1989), there exist two different parameterisations of the Weibull model. They do not yield the same fitted curve for a given dataset (see under Examples).

One four-parameter Weibull model ('weibull1') is

$$f(x) = c + (d - c)\exp(-\exp(b(\log(x) - \log(e)))).$$

Another four-parameter Weibull model ('weibull2') is

$$f(x) = c + (d - c)(1 - \exp(-\exp(b(\log(x) - \log(e))))).$$

Both four-parameter functions are asymmetric with inflection point at the dose e.

Value

The value returned is a list containing the non-linear function, the self starter function and the parameter names.

Note

The functions are for use with the function `drm`.

Author(s)

Christian Ritz

References

Seber, G. A. F. and Wild, C. J (1989) *Nonlinear Regression*, New York: Wiley \& Sons (pp. 338–339).

See Also

For convenience several special cases of the function 'weibull1' are available: `W1.2`, `W1.3` and `W1.4`.

Special cases of 'weibull2' are: `W2.2`, `W2.3` and `W2.4`.

These convenience functions should be used rather than the underlying functions `weibull1` and `weibull2`.

Examples

```
## Fitting two different Weibull models
ryegrass.m1 <- drm(ryegrass, fct = W1.4())
plot(ryegrass.m1, conLevel=0.5)

ryegrass.m2 <- drm(ryegrass, fct = W2.4())
plot(ryegrass.m2, conLevel=0.5, add = TRUE, type = "none", col = 2)
# you could also look at the ED values to see the difference
```

```
## A four-parameter Weibull model with b fixed at 1
ryegrass.m3 <- drm(ryegrass, fct = W1.4(fixed = c(1, NA, NA, NA)))
summary(ryegrass.m3)

## A four-parameter Weibull model with the constraint b>3
ryegrass.m4 <- drm(ryegrass, fct = W1.4(), lowerl = c(3, -Inf, -Inf, -Inf),
control = drmc(constr=TRUE))
summary(ryegrass.m4)
```

`yieldLoss` *Calculating yield loss parameters*

Description

Calculation of parameters in the re-parameterization of the Michaelis-Menten model that is commonly used to assess yield loss (the rectangular hyperbola model)

Usage

```
yieldLoss(object, interval = c("none", "as"), level = 0.95, display = TRUE)
```

Arguments

`object`	object of class 'drc
`interval`	character string specifying the type of confidence intervals to be supplied. The default is "none". Use "as" for asymptotically-based confidence intervals.
`level`	numeric. The level for the confidence intervals. The default is 0.95.
`display`	logical. If TRUE results are displayed. Otherwise they are not (useful in simulations).

Details

The rectangular hyperbola model is a reparameterization of the Michaelis-Menten in terms of parameters A and I

$$Y_L = \frac{Id}{1 + Id/A}$$

where d denotes the weed density and Y_L the resulting yield loss.

Value

For each of the two parameters, a matrix with two or more columns, containing the estimates and the corresponding estimated standard errors and possibly lower and upper confidence limits.

Note

This function is only for use with model fits based on Michaelis-Menten models.

Author(s)

Christian Ritz

References

Cousens, R. (1985). A simple model relating yield loss to weed density, *Ann. Appl. Biol.*, **107**, 239–252.

Examples

```
## Fitting Michaelis-Menten model
met.mm.m1 <- drm(gain~dose, product, data = methionine, fct = MM.3(),
pmodels = list(~1, ~factor(product), ~factor(product)))

## Yield loss parameters with standard errrors
yieldLoss(met.mm.m1)

## Also showing confidence intervals
yieldLoss(met.mm.m1, "as")
```

Index