# Package ‘nplr’

September 4, 2016

**Type** Package  
**Title** N-Parameter Logistic Regression  
**Version** 0.1-5  
**Date** 2016-8-30  
**Maintainer** Frederic Commo <fredcommo@gmail.com>  
**Depends** methods  
**Imports** stats, graphics, utils  
**Suggests** RUnit, knitr  
**VignetteBuilder** knitr  
**Description** Performing drug response analyses and IC50 estimations using n-Parameter logistic regression. Can also be applied to proliferation analyses.  
**License** GPL  
**URL** [https://github.com/fredcommo/nplr](https://github.com/fredcommo/nplr)  
**NeedsCompilation** no  
**Author** Frederic Commo [aut, cre],  
Brian M. Bot [aut]  
**Repository** CRAN  
**Date/Publication** 2016-09-04 20:14:40

### R topics documented:

- `convertToProp` .......................................................... 2
- `getEstimates` ............................................................. 3
- `nplr` ...................................................................... 5
- `nplrAccessors` ............................................................ 8
- `overlay` ................................................................... 8
- `plot.nplr` ................................................................. 9
- `summary.nplr` ............................................................ 11

**Index**  

1
convertToProp

Function to Convert a Vector Into Proportions.

Description
Convert a vector of values to proportions, given a minimum and a maximum value (optional). See Details and Examples.

Usage
convertToProp(y, T0 = NULL, Ctrl = NULL)

Arguments
y : a vector of values (responses to x).
T0 : the minimal value to consider. If NULL (default), \( \min(y, \ na.rm=TRUE) \) will be used. See Details and Warning.
Ctrl : the maximal value to consider. If NULL (default), \( \max(y, \ na.rm=TRUE) \) will be used. See Details and Warning.

Details
In typical cell viability experiments, responses to drug concentrations (inhibition rate) may be estimated with respect to a time zero (T0) and an untreated condition values (Ctrl), as described in [1]:

If none of the T0 and Ctrl values are provided, \( \min(y, \ na.rm=TRUE) \) and \( \max(y, \ na.rm=TRUE) \) will be used, respectively. See Warning.

Value
a vector of values.

Warning
Note that, for drug response analyses, rescaling the responses between 0 to 1 using to the min and max of y, would lead to estimate a EC50 (the half effect between the maximum and the minimum of the observed effects), rather than a IC50.

Note
The data used as examples come from the NCI-60 Growth Inhibition Data: [https://wiki.nci.nih.gov/display/NCIDTPdata/NCI-60+Growth+Inhibition+Data](https://wiki.nci.nih.gov/display/NCIDTPdata/NCI-60+Growth+Inhibition+Data), except for multicell.tsv which are simulated data.
**getEstimates**  

Function to Estimate x Given y.

**Description**

This function takes as its first argument a model returned by `nplr()`. By inverting the logistic model, it estimates the x values corresponding to one (or a vector of) y target(s) provided. The standard error of the model, defined as the mean squared error on the fitted values, is used to estimate a confidence interval on the predicted x values, according to the specified `conf.level`. See `Details`.

**Usage**

```r
## S4 method for signature 'nplr'
getEstimates(object, targets = seq(.9, .1, by = -.1), B = 1e4, conf.level = .95)
```
getEstimates

Arguments

  object : an object of class nplr.
  targets : one, of a vector of, numerical value(s) for which the corresponding x has to be estimated. Default are target values from .9 to .1.
  B : the length of the y distribution from which the x confidence interval is estimated.
  conf.level : the estimated x confidence interval, bounded by (1-conf.level)/2 and 1 - (1-conf.level)/2 (by default .95, which gives x.025 and x.975).

Details

  In n-parameter logistic regressions, none of the parameters follow any particular distribution from which confidence intervals can be estimated. To overcome this issue, the standard error is used to generate a normal distribution of the target(s) passed to the function. The quantiles of that distribution are used in order to provide estimated bounds for the corresponding x value, with respect to conf.level. See also Warning.

Value

  A data set containing:

    y : the target value.
    x.05 : the lower bound of the estimated 95% confidence interval (default). If another value is passed to conf.level, x will be labelled as x.(1-conf.level)/2.
    x : the estimated value.
    x.95 : the upper bound of the estimated 95% confidence interval (default). If another value is passed to conf.level, x will be labelled as x.1-(1-conf.level)/2.

Warning

  Notice that, if any target <= B or target >= T, in other words outside the 2 asymptotes, the maximal (or minimal) possible value the model can estimates is returned.

Note

  The data used in the examples are samples from the NCI-60 Growth Inhibition Data: https://wiki.nci.nih.gov/display/NCIDTPdata/NCI-60+Growth+Inhibition+Data, except for multicell.tsv which are simulated data.

Author(s)

  Frederic Commo, Brian M. Bot

See Also

  nplr.plot.nplr, nplrAccessors
Examples

# Using the PC-3 data
require(nplr)
path <- system.file("extdata", "pc3.txt", package="nplr")
pc3 <- read.delim(path)
model <- nplr(x = pc3$CONC, y = pc3$GPROP)
getEstimates(model)
getEstimates(model, c(.3, .6), conf.level = .9)

nplr

Function to Fit n-Parameter Logistic Regressions.

Description

This function computes a weighted n-parameters logistic regression, given x (typically compound concentrations) and y values (responses: optic densities, fluorescence, cell counts,...). See Details.

Usage

nplr(x, y, useLog = TRUE, LPweight = 0.25, npars = "all",
method = c("res", "sdw", "gw"), silent = FALSE)

Arguments

x : a vector of numeric values, e.g. a vector of drug concentrations.
y : a vector of numeric values, e.g. a vector of responses, typically provided as proportions of control.
useLog : Logical. Should x-values be Log10-transformed. Default to TRUE, set to FALSE if x is already in Log10.
LPweight : a coefficient to adjust the weights. LPweight = 0 will compute a non-weighted np-logistic regression.
npars : a numeric value (or "all") to specify the number of parameters to use in the model. If "all" the logistic model will be tested with 2 to 5 parameters, and the best option will be returned. See Details
method : a character string to specify what weight method to use. Options are "res" (Default), "sdw", "gw". See Details
silent : Logical. Specify whether warnings ad/or messages has to be silenced. Default to FALSE.

Details

The 5-parameter logistic regression is of the form:

\[ y = B + (T - B)/(1 + 10^b \times (x_{\text{mid}} - x)^s) \]
where $B$ and $T$ are the bottom and top asymptotes, respectively, $b$ and $x_{mid}$ are the Hill slope and the x-coordinate at the inflexion point, respectively, and $s$ is an asymmetric coefficient. This equation is sometimes referred to as the Richards’ equation [1,2].

When specifying `npars = 4`, the $s$ parameter is forced to be 1, and the corresponding model is a 4-parameter logistic regression, symmetrical around its inflexion point. When specifying `npars = 3` or `npars = 2`, add 2 more constraints and force $B$ and $T$ to be 0 and 1, respectively.

Weight methods:
The model parameters are optimized, simultaneously, using `nlm`, given a sum of squared errors function, $sse(Y)$, to minimize:

$$sse(Y) = \Sigma [W.(Y_{obs} - Y_{fit})^2]$$

where $Y_{obs}$, $Y_{fit}$ and $W$ are the vectors of observed values, fitted values and weights, respectively.

In order to reduce the effect of possible outliers, the weights can be computed in different ways, specified in `nplr`:

- **residual weights**, "res":
  $$W = (1/residuals)^LPweight$$
  where `residuals` and `LPweight` are the squared error between the observed and fitted values, and a tuning parameter, respectively. Best results are generally obtained by setting $LPweight = 0.25$ (default value), while setting $LPweight = 0$ results in computing a non-weighted sum of squared errors.

- **standard weights**, "sdw":
  $$W = 1/Var(Y_{obs_r})$$
  where $Var(Y_{obs_r})$ is the vector of the within-replicates variances.

- **general weights**, "gw":
  $$W = 1/Y_{fit}^{LPweight}$$
  where $Y_{fit}$ are the fitted values. As for the residuals-weights method, setting $LPweight = 0$ results in computing a non-weighted sum of squared errors.

The standard weights and general weights methods are described in [3].

**Value**

An object of class `nplr`.

**slots**

- `x`: the x values as they are used in the model. It can be $\log(x)$ if `useLog` was set to `TRUE`.
- `y`: the y values.
- `useLog`: logical.
- `npars`: the best number of parameters if `npars="all"`, the specified number of parameters, otherwise.
- `LPweight`: the weights tuning parameter.
- `yFit`: the y fitted values.
• xCurve: the x values generated to draw the curve. 200 points between the min and max of x.
• yCurve: the fitted values used to draw the curve. the fitted values corresponding to xCurve.
• inflPoint: the inflexion point x and y coordinates.
• goodness: the goodness-of-fit. The correlation between the fitted and the observed y values
• stdErr: the mean squared error between the fitted and the observed y values
• pars: the model parameters.
• AUC: the area under the curve estimated using both the trapezoid method and the Simpson's rule.

Note
The data used in the examples are samples from the NCI-60 Growth Inhibition Data: https://wiki.nci.nih.gov/display/NCIDPdata/NCI-60+Growth+Inhibition+Data, except for multicell.tsv which are simulated data.

Author(s)
Frederic Commo, Brian M. Bot

References

See Also
convertToProp, getEstimates, plot.nplr, nplrAccessors

Examples
# Using the PC-3 data
require(nplr)
path <- system.file("extdata", "pc3.txt", package = "nplr")
pc3 <- read.delim(path)
model <- nplr(x = pc3$CONC, y = pc3$GIPROP)
plot(model)
### nplrAccessors

**nplr accessor functions**

**Description**

Methods for extracting information from an object of class `nplr`. Each of the below methods are simply convenience functions which extract the corresponding slots (as the name of each method suggests) from the object of class `nplr`.

**Methods**

```r
signature(object = "nplr")
  * getX(object)
    - getY(object)
    - getXcurve(object)
    - getYcurve(object)
    - getFitValues(object)
    - getInflexion(object)
    - getPar(object)
    - getAUC(object)
    - getGoodness(object)
    - getStdErr(object)
```

**See Also**

`nplr`, `getEstimates`

---

### overlay

**Plotting Multiple nplr Objects**

**Description**

To superimpose multiple logistic models fitted using `nplr`.

**Usage**

```r
overlay(modellist = NULL, showLegend = TRUE, Cols = NULL, ...)
```

**Arguments**

- `modellist` : list. A list of objects of class `nplr`.
- `showLegend` : logical. Whether the legend has to be displayed.
- `Cols` : character. A vector of colors to use. If NULL (default), greys will be used.
- `...` : Other graphical parameters. See `par`.
plot.nplr

Details
None

Source
None

References
None

See Also
plot.nplr

Examples

```r
path <- system.file("extdata", "multicell.tsv", package="nplr")
multicell <- read.delim(path)

# Computing models (to store in a list)
cellsList <- split(multicell, multicell$cell)
Models <- lapply(cellsList, function(tmp){
  nplr(tmp$conc, tmp$resp, silent = TRUE)
})

# Visualizing
overlay(Models, xlab = expression(Log[10](Conc.)), ylab = "Resp.",
       main="Superimposing multiple curves", cex.main=1.5)
```

plot.nplr    Plotting nplr Objects

Description
This function allows visualizing logistic models fitted using nplr.

Usage

```r
## S3 method for class 'nplr'
plot(x, pcol = "aquamarine!", lcol = "red3",
     showEstim = FALSE, showCI = TRUE, showGOF = TRUE, showInf1 = FALSE,
     showPoints = TRUE, showSDe = FALSE, B = 1e4, conf.level = .95, unit = ",", ...)
```
Arguments

- **x**: an object of class \texttt{nplr}
- **pcol**: the points color.
- **lcol**: the line color.
- **showEstim**: logical/numeric. If a numerical value is passed (a y value to reach), the estimated x value, and interval, is displayed on the plot. Default is \texttt{FALSE}
- **showCI**: logical. show the estimated confidence interval
- **showGOF**: logical. show the estimated goodness-of-fit.
- **showInfl**: logical. add the inflexion point on the plot.
- **showPoints**: logical. add the points on the plot.
- **showSDev**: logical. add the standard errors on the plot (maybe useful in case of experiment with replicates).
- **B**: the length of simulated y values. Used to estimate the confidence interval
- **conf.level**: the confidence level. See \texttt{getEstimates}
- **unit**: the unit to specify when \texttt{showEstim} is \texttt{TRUE}. Default is an empty string.
- **...**: other graphical parameters. See \texttt{par}.

Details

None

Note

The data used in the examples are samples from the NCI-60 Growth Inhibition Data:
\url{https://wiki.nci.nih.gov/display/NCIDTPdata/NCI-60+Growth+Inhibition+Data}, except for multicell.tsv which are simulated data.

Source

None

References

None

See Also

\texttt{overlay}

Examples

```r
# Using the PC-3 data
require(nplr)
path <- system.file("extdata", "pc3.txt", package = "nplr")
pc3 <- read.delim(path)
model <- nplr(x = pc3$CONC, y = pc3$GIPROP)
plot(model, showEstim = 0.5, unit = "nM")
```
summary.nplr

summary.nplr  summarying nplr  Objects

Description
A S3 method to visualize a model summary as a table.

Usage
```r
## S3 method for class 'nplr'
summary(object, ...)
```

Arguments
- `object`: an object of class `nplr`
- `...`: other optional parameters (not used).

Details
None

Note
The data used in the examples are samples from the NCI-60 Growth Inhibition Data: [https://wiki.nci.nih.gov/display/NCIDTPdata/NCI-60+Growth+Inhibition+Data](https://wiki.nci.nih.gov/display/NCIDTPdata/NCI-60+Growth+Inhibition+Data), except for multicell.tsv which are simulated data.

Source
None

References
None

See Also
- `plot.nplr`

Examples
```r
# Using the PC-3 data
require(nplr)
path <- system.file("extdata", "pc3.txt", package = "nplr")
pc3 <- read.delim(path)
model <- nplr(x = pc3$CONC, y = pc3$GIPROP)
summary(model)
```
Index

*Topic **textasciitildedekwd1**
  - convertToProp, 2
  - getEstimates, 3
*Topic **textasciitildedekwd2**
  - convertToProp, 2
  - getEstimates, 3
*Topic **datasets**
  - overlay, 8
  - plot.nplr, 9
  - summary.nplr, 11
  - convertToProp, 2, 7
  - getAUC (nplrAccessors), 8
  - getAUC, nplr-method (nplrAccessors), 8
  - getAUC-methods (nplrAccessors), 8
  - getEstimates, 3, 7, 8, 10
  - getEstimates, nplr-method
    (getEstimates), 3
  - getEstimates-methods (getEstimates), 3
  - getFitValues (nplrAccessors), 8
  - getFitValues, nplr-method
    (nplrAccessors), 8
  - getFitValues-methods (nplrAccessors), 8
  - getGoodness (nplrAccessors), 8
  - getGoodness, nplr-method
    (nplrAccessors), 8
  - getGoodness-methods (nplrAccessors), 8
  - getInflexion (nplrAccessors), 8
  - getInflexion, nplr-method
    (nplrAccessors), 8
  - getInflexion-methods (nplrAccessors), 8
  - getPar (nplrAccessors), 8
  - getPar, nplr-method (nplrAccessors), 8
  - getPar-methods (nplrAccessors), 8
  - getStdErr (nplrAccessors), 8
  - getStdErr, nplr-method (nplrAccessors), 8
  - getStdErr-methods (nplrAccessors), 8
  - getX (nplrAccessors), 8
  - getX, nplr-method (nplrAccessors), 8
  - nplr, 3, 4, 5, 8–11
  - nplr-class (nplr), 5
  - nplrAccessors, 4, 7, 8
  - nplrAccessors, nplr-method
    (nplrAccessors), 8
  - nplrAccessors-methods (nplrAccessors), 8
  - overlay, 8, 10
  - par, 8, 10
  - plot.nplr, 4, 7, 9, 9, 11
  - summary.nplr, 11