Package ‘quickpsy’

October 2, 2019

Type    Package
Title   Fits Psychometric Functions for Multiple Groups
Version 0.1.5.1
URL     http://dlinares.org/quickpsy.html
Description Quickly fits and plots psychometric functions (normal, logistic, Weibull or any or any function defined by the user) for multiple groups.
Depends R (>= 3.1.2), DEoptim, dplyr, ggplot2
Imports MPDiR
Encoding UTF-8
License MIT + file LICENSE
LazyData true
RoxygenNote 6.0.1
NeedsCompilation no
Author Linares Daniel [aut, cre], Llopis-Moliner Joan [aut]
Maintainer Linares Daniel <danilinares@gmail.com>
Repository CRAN
Date/Publication 2019-10-02 15:54:02 UTC

R topics documented:

aic .................................................. 2
avbootstrap ...................................... 3
cum_normal_fun .................................. 3
deviance .......................................... 4
devianceboot ..................................... 4
get_functions ................................. 5
inv_cum_normal_fun ............................. 5
inv_logistic_fun ............................... 6
inv_weibull_fun ............................... 6
Description

aic calculates the AICs.

Usage

aic(qp)

Arguments

qp output from quickpsy
avbootstrap

*Description*

avbootstrap creates bootstrap samples

*Usage*

avbootstrap(qp, bootstrap = "parametric", B = 100)

*Arguments*

- `qp`: output from quickpsy
- `bootstrap`: 'parametric' performs parametric bootstrap; 'nonparametric' performs non-parametric bootstrap; 'none' does not perform bootstrap (default is 'parametric').
- `B`: number of bootstrap samples (default is 100 ONLY).

---

cum_normal_fun

*Description*

Cumulative normal function.

*Usage*

cum_normal_fun(x, p)

*Arguments*

- `x`: Vector of values of the explanatory variable.
- `p`: Vector of parameters \( p = (\text{mean}, \text{standard deviation}) \).

*Value*

Probability at each \( x \).

*See Also*

inv_cum_normal_fun

*Examples*

```r
xseq <- seq(0,4,.01)
yseq <- cum_normal_fun(xseq, c(2, .5))
curve <- data.frame(x = xseq, y = yseq)
ggplot(curve, aes(x = x, y = y)) + geom_line()
```
deviance

*Calculates the deviances*

**Description**

deviance calculates the deviances.

**Usage**

deviance(qp)

**Arguments**

qp  
output from quickpsy

**Examples**

```r
library(MPDiR) # contains the Vernier data
fit <- quickpsy(Vernier, Phaseshift, NumUpward, N,
    grouping = .(Direction, Waveform, TempFreq), B = 20)
deviance(fit)
```

devianceboot

*Calculates the bootstrap deviances*

**Description**

deviance calculates the bootstrap deviances.

**Usage**

devianceboot(qp)

**Arguments**

qp  
output from quickpsy

**Examples**

```r
library(MPDiR) # contains the Vernier data
fit <- quickpsy(Vernier, Phaseshift, NumUpward, N,
    grouping = .(Direction, Waveform, TempFreq), B = 20)
devianceboot(fit)
```
**get_functions**

Predefined functions

---

**Description**

getfunctions lists the predefined functions in quickpsy.

**Usage**

get_functions()

**See Also**

cum_normal_fun, logistic_fun, weibull_fun

---

**inv_cum_normal_fun**

Inverse cumulative normal function

---

**Description**

Inverse cumulative normal function

**Usage**

inv_cum_normal_fun(prob, p)

**Arguments**

- **prob**: Vector of probabilities.
- **p**: Vector of parameters \( p = (\text{mean}, \text{standard deviation}) \).

**Value**

x at each probability. #' @seealso cum_normal_fun

**Examples**

```r
yseq <- seq(0, 1, .01)
xseq <- inv_cum_normal_fun(yseq, c(2, .5))
curve <- data.frame(x = xseq, y = yseq)
ggplot(curve, aes(x = x, y = y)) + geom_line()
```
inv_logistic_fun  Inverse logistic function

Description
Inverse logistic function

Usage
inv_logistic_fun(q, p)

Arguments
q  Vector of probabilities.
p  Vector of parameters \( p = c(\alpha, \beta) \).

Value
x at each probability.

See Also
logistic_fun

Examples
yseq <- seq(0, 1, .01)
xseq <- inv_logistic_fun(yseq, c(2, 4))
curve <- data.frame(x = xseq, y = yseq)
ggplot(curve, aes(x = x, y = y)) + geom_line()

inv_weibull_fun  Inverse Weibull function

Description
Inverse Weibull function

Usage
inv_weibull_fun(q, p)

Arguments
q  Vector of probabilities.
p  Vector of parameters \( p = c(\alpha, \beta) \).
logistic_fun

Value

x at each probability.

See Also

weibull_fun

Examples

```r
yseq <- seq(0, 1, .01)
xseq <- inv_weibull_fun(yseq, c(2, 4))
curve <- data.frame(x = xseq, y = yseq)
ggplot(curve, aes(x = x, y = y)) + geom_line()
```

---

logistic_fun  Logistic function

Description

Logistic function of the form \( (1 + \exp(-\beta \cdot (x - \alpha))) - 1 \)

Usage

`logistic_fun(x, p)`

Arguments

- `x` Vector of values of the explanatory variable.
- `p` Vector of parameters \( p = c(\alpha, \beta) \).

Value

Probability at each x.

See Also

inv_logistic_fun

Examples

```r
xseq <- seq(0, 4, .01)
yseq <- logistic_fun(xseq, c(2, 4))
curve <- data.frame(x = xseq, y = yseq)
ggplot(curve, aes(x = x, y = y)) + geom_line()
```
logliks

*Calculates the loglikelihoods* logliks calculates the loglikelihoods.

**Description**

Calculates the loglikelihoods logliks calculates the loglikelihoods.

**Usage**

logliks(qp)

**Arguments**

qp output from quickpsy

---

logliksboot

*Calculates the bootstrap loglikelihoods*

**Description**

logliksboot calculates the bootstrap loglikelihoods.

**Usage**

logliksboot(qp)

**Arguments**

qp output from quickpsy

**Examples**

```r
library(MPDiR) # contains the Vernier data
fit <- quickpsy(Vernier, Phaseshift, NumUpward, N,
                 grouping = .(Direction, Waveform, TempFreq), B = 20)
logliksboot(fit)
```
logliksbootsaturated

Calculates the bootstrap loglikelihoods for the saturated model

Description

logliks calculates the bootstrap loglikelihoods for the saturated model.

Usage

logliksbootsaturated(qp)

Arguments

qp output from quickpsy

Examples

library(MPDiR) # contains the Vernier data
fit <- quickpsy(Vernier, Phaseshift, NumUpward, N,
    grouping = .(Direction, Waveform, TempFreq), B = 20)
logliksbootsaturated(fit)

loglikssaturated

Calculates the loglikelihoods of the saturated model

Description

loglikssaturated calculates the loglikelihoods of the saturated model.

Usage

loglikssaturated(qp)

Arguments

qp output from quickpsy

Examples

library(MPDiR) # contains the Vernier data
fit <- quickpsy(Vernier, Phaseshift, NumUpward, N,
    grouping = .(Direction, Waveform, TempFreq), B = 20)
loglikssaturated(fit)
parbootstrap

*Creates bootstrap samples of the parameters*

**Description**

parbootstrap creates bootstrap samples of the parameters.

**Usage**

parbootstrap(qp)

**Arguments**

- **qp**
  - output from quickpsy

plotcurves

*Plot the curves*

**Description**

plotcurves plot the curves.

**Usage**

plotcurves(qp, panel = NULL, xpanel = NULL, ypanel = NULL, color = NULL,
           averages = T, curves = T, thresholds = T, ci = T)

**Arguments**

- **qp**
  - output from quickpsy
- **panel**
  - Name of the variable to be split in panels.
- **xpanel**
  - Name of the variable to be split in horizontal panels.
- **ypanel**
  - Name of the variable to be split in vertical panels.
- **color**
  - Name of the variable coded by color.
- **averages**
  - If FALSE averaged probabilities are not plotted (default is TRUE).
- **curves**
  - If FALSE curves are not plotted (default is TRUE)
- **thresholds**
  - If FALSE thresholds are not plotted (default is TRUE)
- **ci**
  - If FALSE confidence intervals are not plotted (default is TRUE)

**See Also**

plotcurves_
Examples

```r
library(MPDiR) # contains the Vernier data
fit <- quickpsy(Vernier, Phaseshift, NumUpward, N,
                grouping = .(Direction, WaveForm, TempFreq), B = 5)
plotcurves(fit)
plotcurves(fit, xpanel = Direction)
plotcurves(fit, xpanel = Direction, color = WaveForm, ci = FALSE)
```

Description

`plotcurves_` is the standard evaluation SE function associated to the non-standard evaluation NSE function `plotcurves`. SE functions can be more easily called from other functions. In SE functions, you need to quote the names of the variables.

Usage

```r
plotcurves_(qp, panel = NULL, xpanel = NULL, ypanel = NULL,
            color = NULL, averages = TRUE, curves = TRUE, thresholds = TRUE,
            ci = TRUE)
```

Arguments

- `qp` output from `quickpsy`
- `panel` Name of the variable to be split in panels.
- `xpanel` Name of the variable to be split in horizontal panels.
- `ypanel` Name of the variable to be split in vertical panels.
- `color` Name of the variable coded by color.
- `averages` If `FALSE` averaged probabilities are not plotted (default is `TRUE`).
- `curves` If `FALSE` curves are not plotted (default is `TRUE`)
- `thresholds` If `FALSE` thresholds are not plotted (default is `TRUE`)
- `ci` If `FALSE` confidence intervals are not plotted (default is `TRUE`)

See Also

- `plotcurves`
Examples

```r
library(MPDiR) # contains the Vernier data
data(Vernier) # ?Vernier for the reference
fit <- quickpsy(Vernier, Phaseshift, NumUpward, N,
    grouping = .(Direction, WaveForm, TempFreq), B = 5)
plotcurves_(fit, xpanel = 'Direction')
plotcurves_(fit, color = 'Direction')
plotcurves_(fit, xpanel = 'Direction', color = 'WaveForm', ci = FALSE)
```

---

**plotpar**  
*Plot the values of the parameters*

Description

plotpar plot the values of the parameters.

Usage

```r
plotpar(qp, x = NULL, panel = NULL, xpanel = NULL, ypanel = NULL,
    color = NULL, geom = "bar", ci = T)
```

Arguments

- `qp`  
  output from quickpsy.
- `x`  
  Name of the variable to displayed in the x-axis.
- `panel`  
  Name of the variable to be split in panels.
- `xpanel`  
  Name of the variable to be split in horizontal panels.
- `ypanel`  
  Name of the variable to be split in vertical panels.
- `color`  
  Name of the variable coded by color.
- `geom`  
  If 'bar' displays bars. If 'point' displays points (default is 'bar').
- `ci`  
  If FALSE confidence intervals are not plotted (default is TRUE).

See Also

- `plotpar`

Examples

```r
library(MPDiR) # contains the Vernier data
fit <- quickpsy(Vernier, Phaseshift, NumUpward, N,
    grouping = .(Direction, WaveForm, TempFreq), B = 10)
plotpar(fit)
plotpar(fit, x = WaveForm)
plotpar(fit, xpanel = Direction)
plotpar(fit, color = Direction)
plotpar(fit, color = Direction, ypanel = WaveForm, geom = 'point')
```
Description

`plotpar_` is the standard evaluation SE function associated to the non-standard evaluation NSE function `plotpar`. SE functions can be more easily called from other functions. In SE functions, you need to quote the names of the variables.

Usage

```r
plotpar_(qp, x = NULL, panel = NULL, xpanel = NULL, ypanel = NULL,
        color = NULL, geom = "bar", ci = T)
```

Arguments

- `qp` output from `quickpsy`.
- `x` Name of the variable to displayed in the x-axis.
- `panel` Name of the variable to be split in panels.
- `xpanel` Name of the variable to be split in horizontal panels.
- `ypanel` Name of the variable to be split in vertical panels.
- `color` Name of the variable coded by color.
- `geom` If `"bar"` displays bars. If `"point"` displays points (default is `"bar"`).
- `ci` If FALSE confidence intervals are not plotted (default is TRUE).

See Also

`plotpar`

Examples

```r
library(MPDiR) # contains the Vernier data
fit <- quickpsy(Vernier, Phaseshift, NumUpward, N,
                grouping = .(Direction, WaveForm, TempFreq), bootstrap = 'none')

plotpar_(fit, x = 'WaveForm')
plotpar_(fit, xpanel = 'Direction')
plotpar_(fit, color = 'Direction')
plotpar_(fit, color = 'Direction', ypanel = 'WaveForm', geom = 'point')
```
plotthresholds

Plot the thresholds

Description

plotthresholds plot the thresholds.

Usage

plotthresholds(qp, x = NULL, panel = NULL, xpanel = NULL, ypanel = NULL,
              color = NULL, geom = "bar", ci = T, sizeerrorbar = 0.5)

Arguments

- **qp**: output from quickpsy.
- **x**: Name of the variable to displayed in the x-axis.
- **panel**: Name of the variable to be split in panels.
- **xpanel**: Name of the variable to be split in horizontal panels.
- **ypanel**: Name of the variable to be split in vertical panels.
- **color**: Name of the variable coded by color.
- **geom**: If 'bar' displays bars.
- **ci**: If FALSE confidence intervals are not plotted (default is TRUE).
- **sizeerrorbar**: Line width of the error bars. If 'point' displays points (default is 'bar').

See Also

plotthresholds_

Examples

library(MPDiR) # contains the Vernier data
fit <- quickpsy(Vernier, Phaseshift, NumUpward, N,
                grouping = .(Direction, WaveForm, TempFreq), B = 10)
plotthresholds(fit)
plotthresholds(fit, x = WaveForm)
plotthresholds(fit, xpanel = Direction)
plotthresholds(fit, color = Direction, ypanel = WaveForm, geom = 'point')
plotthresholds_  
*Plot the thresholds*

**Description**

plotthresholds_ is the standard evaluation SE function associated to the non-standard evaluation NSE function plotthresholds. SE functions can be more easily called from other functions. In SE functions, you need to quote the names of the variables.

**Usage**

```r
plotthresholds_(qp, x = NULL, panel = NULL, xpanel = NULL, ypanel = NULL, color = NULL, geom = "bar", ci = T, sizeerrorbar = 0.5)
```

**Arguments**

- `qp`: output from quickpsy.
- `x`: Name of the variable to displayed in the x-axis.
- `panel`: Name of the variable to be split in panels.
- `xpanel`: Name of the variable to be split in horizontal panels.
- `ypanel`: Name of the variable to be split in vertical panels.
- `color`: Name of the variable coded by color.
- `geom`: If 'bar' displays bars.
- `ci`: If FALSE confidence intervals are not plotted (default is TRUE).
- `sizeerrorbar`: Line width of the error bars. If 'point' displays points (default is 'bar').

**See Also**

plotthresholds

**Examples**

```r
library(MPDiR) # contains the Vernier data
fit <- quickpsy(Vernier, Phaseshift, NumUpward, N, 
    grouping = .(Direction, Waveform, TempFreq), B = 10)

plotthresholds_(fit, x = 'Waveform')
plotthresholds_(fit, xpanel = 'Direction')
plotthresholds_(fit, color = 'Direction')
plotthresholds_(fit, color = 'Direction', ypanel = 'Waveform', geom = 'point')
```
qpdat  

*Data set for demonstration*

**Description**

It is part of the data associated with the paper 'Motion signal and the perceived positions of moving objects'.

**Usage**

qpdat

**Format**

An object of class `grouped_df` (inherits from `tbl_df`, `tbl`, `data.frame`) with 6240 rows and 8 columns.

**References**


quickpsy  

*Fits psychometric functions*

**Description**

quickpsy fits, by direct maximization of the likelihood (Prins and Kingdom, 2010; Knoblauch and Maloney, 2012), psychometric functions of the form

\[ \psi(x) = \gamma + (1 - \gamma - \lambda) \ast fun(x) \]

where \( \gamma \) is the guess rate, \( \lambda \) is the lapse rate and \( fun \) is a sigmoidal-shape function with asymptotes at 0 and 1.

**Usage**

quickpsy(d, x = x, k = k, n = n, grouping, random, within, between, xmin = NULL, xmax = NULL, log = FALSE, fun = cum_normal_fun, parini = NULL, guess = 0, lapses = 0, prob = NULL, thresholds = T, bootstrap = "parametric", B = 100, ci = 0.95, optimization = "optim")
Arguments

d  Data frame with the results of a Yes-No experiment to fit. It should have a tidy form in which each column corresponds to a variable and each row is an observation.

x  Name of the explanatory variable.

k  Name of the response variable. The response variable could be the number of trials in which a yes-type response was given or a vector of 0s (or -1s; no-type response) and 1s (yes-type response) indicating the response on each trial.

n  Only necessary if k refers to the number of trials in which a yes-type response was given. It corresponds to the name of the variable indicating the total number of trials.

grouping  Name of the grouping variables. It should be specified as grouping = .(variable_name1,variable_name2).

random  Name of the random variable. It should be specified as random = .(variable_name1,variable_name2).

In the current version of quickpsy, the random variable has not special treatment. It does the same as grouping.

within  Name of the within variable. It should be specified as within = .(variable_name1,variable_name2).

In the current version of quickpsy, the within variable has not special treatment. It does the same as grouping.

between  Name of the between variable. It should be specified as between = .(variable_name1,variable_name2).

In the current version of quickpsy, the between variable has not special treatment. It does the same as grouping.

xmin  Minimum value of the explanatory variable for which the curves should be calculated (the default is the minimum value of the explanatory variable).

xmax  Maximum value of the explanatory variable for which the curves should be calculated (the default is the maximum value of the explanatory variable).

log  If TRUE, the logarithm of the explanatory variable is used to fit the curves (default is FALSE).

fun  Name of the shape of the curve to fit. It could be a predefined shape (cum_normal_fun, logistic_fun, weibull_fun) or the name of a function introduced by the user (default is cum_normal_fun).

parini  Initial parameters. quickpsy calculates default initial parameters using probit analysis, but it is also possible to specify a vector of initial parameters or a list of the form list(c(par1min,par1max),c(par2min,par2max)) to constraint the lower and upper bounds of the parameters (when optimization = 'DE', parini should be also a list).

guess  Value indicating the guess rate \( \gamma \) (default is 0). If TRUE, the guess rate is estimated as the \( i + 1 \) parameter where \( i \) corresponds to the number of parameters of fun. If, for example, fun is a predefined shape with parameters p1 and p2, then the guess rate corresponds to parameter p3.

lapses  Value indicating the lapse rate \( \lambda \) (default is 0). If TRUE, the lapse rate is estimated as the \( i + 1 \) parameter where \( i \) corresponds to the number of parameters of fun plus one if the guess rate is estimated. If, for example, fun is a predefined shape with parameters p1 and p2, then the lapse rate corresponds to parameter p3. If the guess rate is also estimated, p3 will be the guess rate and p4 the lapse rate.
prob Probability to calculate the threshold (default is guess + .5 * (1 - guess)).
thresholds If FALSE, thresholds are not calculated (default is TRUE).
bootstrap 'parametric' performs parametric bootstrap; 'nonparametric' performs non-parametric bootstrap; 'none' does not perform bootstrap (default is 'parametric').
B number of bootstrap samples (default is 100 ONLY).
ci Confidence intervals level based on percentiles (default is .95).
optimization Method used for optimization. The default is 'optim' which uses the optim function. It can also be 'DE' which uses de function DEoptim from the package DEoptim, which performs differential evolution optimization. By using DEoptim, it is less likely that the optimization finishes in a local minimum, but the optimization is slow. When 'DE' is used, parini should be specified as a list with lower and upper bounds.

Value

A list containing the following components:

- x, k, n
- groups The grouping variables.
- funname String with the name of the shape of the curve.
- psyfunguesslapses Curve including guess and lapses.
- limits Limits of the curves.
- parini Initial parameters.
- optimization Method to optimize.
- pariniset FALSE if initial parameters are not given.
- ypred Predicted probabilities at the values of the explanatory variable.
- curves Curves.
- par Fitted parameters and its confidence intervals.
- curvesbootstrap Bootstrap curves.
- thresholds Thresholds.
- thresholdsci Confidence intervals for the thresholds.
- logliks Log-likelihoods of the model.
- loglikssaturated Log-likelihoods of the saturated model.
- deviance Deviance of the model and the p-value calculated by bootstraping.
- aic AIC of the model defined as

\[-2 \times \loglik + 2 \times k\]

where k is the number of parameters of the model.
References


See Also

quickpsy

Examples

# make sure that all the requires packages are installed
# and loaded; instructions at https://github.com/danilinares/quickpsy
library(MPDiR) # contains the Vernier data; use ?Vernier for the reference
fit <- quickpsy(Vernier, Phaseshift, NumUpward, N, 
grouping = .(Direction, Waveform, TempFreq), B = 10)
plotcurves(fit)
plotpar(fit)
plotthresholds(fit, geom = 'point')

quickpsy_ Fits psychometric functions

Description

quickpsy_ is the standard evaluation SE function associated to the non-standard evaluation NSE function quickpsy. SE functions can be more easily called from other functions. In SE functions, you need to quote the names of the variables.

Usage

quickpsy_(d, x = "x", k = "k", n = "n", grouping, random, within, between, 
xmin = NULL, xmax = NULL, log = FALSE, fun = "cum_normal_fun", 
parini = NULL, guess = 0, lapses = 0, prob = NULL, thresholds = T, 
bootstrap = "parametric", B = 100, ci = 0.95, optimization = "optim")

Arguments

d Data frame with the results of a Yes-No experiment to fit. It should have a tidy form in which each column corresponds to a variable and each row is an observation.

x Name of the explanatory variable.

k Name of the response variable. The response variable could be the number of trials in which a yes-type response was given or a vector of 0s (or -1s; no-type response) and 1s (yes-type response) indicating the response on each trial.
Only necessary if \( k \) refers to the number of trials in which a yes-type response was given. It corresponds to the name of the variable indicating the total number of trials.

**grouping**
Name of the grouping variables. It should be specified as `grouping = .(variable_name1,variable_name2)`.

**random**
Name of the random variable. It should be specified as `random = .(variable_name1,variable_name2)`.
In the current version of quickpsy, the random variable has not special treatment.
It does the same as grouping.

**within**
Name of the within variable. It should be specified as `within = .(variable_name1,variable_name2)`.
In the current version of quickpsy, the within variable has not special treatment.
It does the same as grouping.

**between**
Name of the between variable. It should be specified as `between = .(variable_name1,variable_name2)`.
In the current version of quickpsy, the between variable has not special treatment. It does the same as grouping.

**xmin**
Minimum value of the explanatory variable for which the curves should be calculated (the default is the minimum value of the explanatory variable).

**xmax**
Maximum value of the explanatory variable for which the curves should be calculated (the default is the maximum value of the explanatory variable).

**log**
If \( \text{TRUE} \), the logarithm of the explanatory variable is used to fit the curves (default is \( \text{FALSE} \)).

**fun**
Name of the shape of the curve to fit. It could be a predefined shape (`cum_normal_fun`, `logistic_fun`, `weibull_fun`) or the name of a function introduced by the user (default is `cum_normal_fun`).

**parini**
Initial parameters. quickpsy calculates default initial parameters using probit analysis, but it is also possible to specify a vector of initial parameters or a list of the form `list(c(par1min,par1max),c(par2min,par2max))` to constraint the lower and upper bounds of the parameters (when optimization = 'DE', parini should be also a list).

**guess**
Value indicating the guess rate \( \gamma \) (default is 0). If \( \text{TRUE} \), the guess rate is estimated as the \( i + 1 \) parameter where \( i \) corresponds to the number of parameters of `fun`. If, for example, `fun` is a predefined shape with parameters `p1` and `p2`, then the guess rate corresponds to parameter `p3`.

**lapses**
Value indicating the lapse rate \( \lambda \) (default is 0). If \( \text{TRUE} \), the lapse rate is estimated as the \( i + 1 \) parameter where \( i \) corresponds to the number of parameters of `fun` plus one if the guess rate is estimated. If, for example, `fun` is a predefined shape with parameters `p1` and `p2`, then the lapse rate corresponds to parameter `p3`. If the guess rate is also estimated, `p3` will be the guess rate and `p4` the lapse rate.

**prob**
Probability to calculate the threshold (default is guess + .5 * (1 - guess)).

**thresholds**
If \( \text{FALSE} \), thresholds are not calculated (default is \( \text{TRUE} \)).

**bootstrap**
'parametric' performs parametric bootstrap; 'nonparametric' performs non-parametric bootstrap; 'none' does not perform bootstrap (default is 'parametric').

**B**
Number of bootstrap samples (default is 100 ONLY).

**ci**
Confidence intervals level based on percentiles (default is .95).
optimization

Method used for optimization. The default is 'optim' which uses the optim function. It can also be 'DE' which uses de function DEoptim from the package DEoptim, which performs differential evolution optimization. By using DEoptim, it is less likely that the optimization finishes in a local minimum, but the optimization is slow. When 'DE' is used, parini should be specified as a list with lower and upper bounds.

See Also

quickpsy

Description

quickreadfiles builds a data frame from several txt files. It assumes that in each file, the first row has the names of the variables.

Usage

quickreadfiles(path = getwd(), extension = "txt", ...)

Arguments

path

Path of the file (default is the working directory).

extension

Specify whether the file extension is 'txt' or 'csv'.

... arguments of the form name_var = c('value1', 'value2'...). A new column with variable name name_var is added to the data frame.

Examples

# download the 3 files in
# https://github.com/danilinares/quickpsy/tree/master/inst/extdata/example1
# and add them to your working directory
# dat <- quickreadfiles(subject = c('aa', 'bb', 'cc'), session = c('1', '2'))
# fit <- quickpsy(dat, phase, resp, grouping=(subject), lapses = T, guess = T)
# plotcurves(fit)
Summary

**sse**

*Sum of squared errors of prediction*

---

**Description**

`ypred` calculates the sum of squared errors of prediction.

**Usage**

```r
sse(qp)
```

**Arguments**

- `qp`: output from `quickpsy`

---

**summary.quickpsy**

*Plot the parameters and its confidence intervals*

---

**Description**

Plot the parameters and its confidence intervals.

**Usage**

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

**Arguments**

- `object`: An object for which a summary is desired.
- `...`: Additional arguments affecting the summary produced.
weibull_fun

Weibull function

Description
Weibull function of the form $(1 - \exp(-(x/\alpha)\beta))$

Usage
weibull_fun(x, p)

Arguments
x
Vector of values of the explanatory variable.
p
Vector of parameters $p = c(\alpha, \beta)$.

Value
Probability at each x.

Examples
xseq <- seq(0, 4, .01)
yseq <- weibull_fun(xseq, c(2, 4))
curve <- data.frame(x = xseq, y = yseq)
ggplot(curve, aes(x = x, y = y)) + geom_line()

ypred
Predicted probabilities

Description
ypred calculates the predicted probabilities at the values of the explanatory variable.

Usage
ypred(qp)

Arguments
qp
output from quickpsy

Examples
library(MPDiR) # contains the Vernier data
data(Vernier) # ?Vernier for the reference
fit <- quickpsy(Vernier, Phaseshift, NumUpward, N,
                grouping = .(Direction, WaveForm, TempFreq), B = 20)
ypred(fit)
Index

*Topic datasets
  qpdat, 16

aic, 2
avbootstrap, 3
cum_normal_fun, 3, 5
deviance, 4
devianceboot, 4
get_functions, 5
inv_cum_normal_fun, 3, 5
inv_logistic_fun, 6, 7
inv_weibull_fun, 6
logistic_fun, 5, 6, 7
logliks, 8
logliksboot, 8
logliksbootsaturated, 9
loglikssaturated, 9
parbootstrap, 10
plotcurves, 10, 11
plotcurves_, 10, 11
plotpar, 12, 13
plotpar_, 12, 13
plotthresholds, 14, 15
plotthresholds_, 14, 15
qpdat, 16
quickpsy, 16, 21
quickpsy_, 19, 19
quickreadfiles, 21
sse, 22
summary.quickpsy, 22
weibull_fun, 5, 7, 23
ypred, 23