Package ‘AATtools’

January 12, 2021

Type Package

Title Reliability and Scoring Routines for the Approach-Avoidance Task

Version 0.0.2

Description Compute approach bias scores using different scoring algorithms,
compute bootstrapped and exact split-half reliability estimates,
and compute confidence intervals for individual participant scores.

Depends R (¿= 3.6.0)

Imports magrittr, dplyr, doParallel, foreach

License GPL-3

Encoding UTF-8

BugReports https://github.com/Spiritspeak/AATtools/issues

LazyData true

ByteCompile true

RoxygenNote 7.1.1

R topics documented:

aat_alpha — *Compute reliability using Cronbach's Alpha*

Description

aat_alpha computes approach-avoidance scores for each stimulus within each participant. It then computes Cronbach's alpha by treating each stimulus as an item on a questionnaire and measuring how much these scores correlate within each participant.

aat_alpha_jackknife additionally leaves out one stimulus or participant at a time and computes Cronbach's alpha for each such exclusion. This gives a glimpse of which stimuli may have especially confused participants or otherwise induced responses that are unlike those to other stimuli.

Please note that this method does not tolerate missing values - for every movement direction, within every stimulus, within every participant, there must be at least one trial - otherwise either the entire stimulus or participant must be excluded. q_reliability or aat_splithalf are recommended instead.

Usage

```
aat_alpha(
  ds,
  subjvar,
  stimvar,
  pullvar,
  rtvar,
  algorithm = c("aat_singlemeandiff", "aat_singlemediandiff"),
  delete.missing = c("subjects", "stimuli", "both", "none")
)

aat_alpha_jackknife(
  ds,
  subjvar,
  stimvar,
  pullvar,
  rtvar,
  algorithm = c("aat_singlemeandiff", "aat_singlemediandiff"),
  delete.missing = c("subjects", "stimuli", "both", "none")
)

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

## S3 method for class 'aat_alpha'
plot(x, ...)

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

## S3 method for class 'aat_alpha_jackknife'
plot(x, exclusions = c("both", "stimulus", "participant"), ...)
```

Arguments

`ds`	a data.frame
`subjvar`	character naming the column with participant IDs
`stimvar`	character naming the column with stimulus IDs
`pullvar`	character naming the column for movement direction
`rtvar`	character indicating the column with reaction times
`algorithm`	character name of the algorithm used to compute per-stimulus approach bias scores
`delete.missing`	character denoting the deletion strategy when missing values are encountered, which may occur when one participant saw a stimulus which another participant did not see, or when all approach trials for a certain stimulus were excluded for one participant. Defaults to excluding the entire participant's dataset.
`x`	an `aat_alpha` or `aat_alpha_jackknife` object
`...`	Ignored.
`exclusions`	Character. Should the function display Cronbach's alpha for individually excluded participants, stimuli, or both?

Value

`aat_alpha` returns an `aat_alpha` S3 object, `aat_alpha_jackknife` returns an `aat_alpha_jackknife` S3 object, both with print and plot methods.

Author(s)

Sercan Kahveci

References

Cousijn, J., Goudriaan, A. E., & Wiers, R. W. (2011). Reaching out towards cannabis: Approach-bias in heavy cannabis users predicts changes in cannabis use. Addiction, 106(9), 1667-1674.

Examples

```
data(erotica)
#We artificially reduce the number of stimuli here because the original
#erotica dataset is not suitable for computing Cronbach's alpha.
erotica$stimulus<- substr(as.character(erotica$stimulus),5,5)

myalpha<-aat_alpha(erotica,"subject","stimulus","is_pull","RT")
print(myalpha)
plot(myalpha)

myalpha2<-aat_alpha_jackknife(erotica,"subject","stimulus","is_pull","RT")
print(myalpha2)
plot(myalpha2)
```

aat_bootstrap *Compute bootstrapped approach-bias scores*

Description

Compute bootstrapped approach-bias scores with confidence intervals.

Usage

```
aat_bootstrap(
  ds,
  subjvar,
  pullvar,
  targetvar = NULL,
  rtvar,
  iters,
  algorithm = c("aat_doublemeandiff", "aat_doublemediandiff", "aat_dscore",
    "aat_dscore_multiblock", "aat_regression", "aat_standardregression",
    "aat_doublemeanquotient", "aat_doublemedianquotient", "aat_singlemeandiff",
    "aat_singlemediandiff"),
 trialdropfunc = c("prune_nothing", "trial_prune_3SD", "trial_prune_SD_dropcases",
  "trial_recode_SD", "trial_prune_percent_subject", "trial_prune_percent_sample"),
  errortrialfunc = c("prune_nothing", "error_replace_blockmeanplus",
    "error_prune_dropcases"),
  plot = TRUE,
  include.raw = FALSE,
  parallel = TRUE,
  ...
)

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

## S3 method for class 'aat_bootstrap'
plot(x, ...)
```

Arguments

ds	a longformat data.frame
subjvar	Quoted name of the participant identifier column
pullvar	Quoted name of the column indicating pull trials. Pull trials should either be represented by 1, or by the second level of a factor.
targetvar	Name of the column indicating trials featuring the target stimulus. Target stimuli should either be represented by 1, or by the second level of a factor.
rtvar	Name of the reaction time column.
iters	Total number of desired iterations. At least 200 are required to get confidence intervals that make sense.
algorithm	Function (without brackets or quotes) to be used to compute AAT scores. See Algorithms for a list of usable algorithms.

`trialdropfunc` Function (without brackets or quotes) to be used to exclude outlying trials in each half. The way you handle outliers for the reliability computation should mimic the way you do it in your regular analyses. It is recommended to exclude outlying trials when computing AAT scores using the mean double-dfference scores and regression scoring approaches, but not when using d-scores or median double-difference scores.

- `prune_nothing` excludes no trials (default)
- `trial_prune_3SD` excludes trials deviating more than 3SD from the mean per participant.
- `trial_prune_SD_dropcases` removes trials deviating more than a specific number of standard deviations from the participant's mean, and removes participants with an excessive percentage of outliers. Required arguments:
  - `trialsd` - trials deviating more than `trialsd` standard deviations from the participant's mean are excluded (optional; default is 3)
  - `maxoutliers` - participants with a higher percentage of outliers are removed from the data. (optional; default is .15)
- `trial_recode_SD` recodes outlying reaction times to the nearest non-outlying value, with outliers defined as reaction times deviating more than a certain number of standard deviations from the participant's mean. Required argument:
  - `trialsd` - trials deviating more than this many standard deviations from the mean are classified as outliers.
- `trial_prune_percent_subject` and `trial_prune_percent_sample` remove trials below and/or above certain percentiles, on a subject-by-subject basis or sample-wide, respectively. The following arguments are available:
  - `lowerpercent` and `uppperpercent` (optional; defaults are .01 and .99).

`errortrialfunc` Function (without brackets or quotes) to apply to an error trial.

- `prune_nothing` removes no errors (default).
- `error_replace_blockmeanplus` replaces error trial reaction times with the block mean, plus an arbitrary extra quantity. If used, the following additional arguments are required:
  - `blockvar` - Quoted name of the block variable (mandatory)
  - `errorvar` - Quoted name of the error variable, where errors are 1 or TRUE and correct trials are 0 or FALSE (mandatory)
  - `errorbonus` - Amount to add to the reaction time of error trials. Default is 0.6 (recommended by `Greenwald,Nosek,& Banaji,2003`)
- `error_prune_dropcases` removes errors and drops participants if they have more errors than a given percentage. The following arguments are available:
  - `errorvar` - Quoted name of the error variable, where errors are 1 or TRUE and correct trials are 0 or FALSE (mandatory)
  - `maxerrors` - participants with a higher percentage of errors are excluded from the dataset. Default is .15.

`plot` Plot the bias scores and their confidence intervals after computation is complete. This gives a good overview of the data.

include.raw	logical indicating whether raw split-half data should be included in the output object.
parallel	If TRUE (default), will use parallel computing to compute results faster. If a doParallel backend has not been registered beforehand, this function will register a cluster and stop it after finishing, which takes some extra time.
...	Other arguments, to be passed on to the algorithm or outlier rejection functions (see arguments above)
x	An `aat_bootstrap` object.

Value

A list, containing bootstrapped bias scores, their variance, bootstrapped 95 percent confidence intervals, the number of iterations, and a matrix of bias scores for each iteration.

Author(s)

Sercan Kahveci

Examples

```
# Compute 10 bootstrapped AAT scores.
boot<-aat_bootstrap(ds=erotica[erotica$is_irrelevant==0,], subjvar="subject",
                    pullvar="is_pull", targetvar="is_target",rtvar="RT",
                    iters=10,algorithm="aat_doublemediandiff",
                    trialdropfunc="trial_prune_3SD",
                    plot=FALSE, parallel=FALSE)
plot(boot)
print(boot)
```

aat_compute *Compute simple AAT scores*

Description

Compute simple AAT scores, with optional outlier exclusion and error trial recoding.

Usage

```
aat_compute(
  ds,
  subjvar,
  pullvar,
  targetvar = NULL,
  rtvar,
  algorithm = c("aat_doublemeandiff", "aat_doublemediandiff", "aat_dscore",
    "aat_dscore_multiblock", "aat_regression", "aat_standardregression",
    "aat_doublemeanquotient", "aat_doublemedianquotient", "aat_singlemeandiff",
    "aat_singlemediandiff"),
 trialdropfunc = c("prune_nothing", "trial_prune_3SD", "trial_prune_SD_dropcases",
  "trial_recode_SD", "trial_prune_percent_subject", "trial_prune_percent_sample"),
  errortrialfunc = c("prune_nothing", "error_replace_blockmeanplus",
    "error_prune_dropcases"),
  ...
)
```

Arguments

`ds`	a long-format data.frame
`subjvar`	column name of subject variable
`pullvar`	column name of pull/push indicator variable, must be numeric or logical (where pull is 1 or TRUE)
`targetvar`	column name of target stimulus indicator, must be numeric or logical (where target is 1 or TRUE)
`rtvar`	column name of reaction time variable
`algorithm`	Function (without brackets or quotes) to be used to compute AAT scores. See Algorithms for a list of usable algorithms.

`trialdropfunc` Function (without brackets or quotes) to be used to exclude outlying trials in each half. The way you handle outliers for the reliability computation should mimic the way you do it in your regular analyses. It is recommended to exclude outlying trials when computing AAT scores using the mean double-dfference scores and regression scoring approaches, but not when using d-scores or median double-difference scores.

- `prune_nothing` excludes no trials (default)
- `trial_prune_3SD` excludes trials deviating more than 3SD from the mean per participant.
- `trial_prune_SD_dropcases` removes trials deviating more than a specific number of standard deviations from the participant's mean, and removes participants with an excessive percentage of outliers. Required arguments:
  - `trialsd` - trials deviating more than `trialsd` standard deviations from the participant's mean are excluded (optional; default is 3)
  - `maxoutliers` - participants with a higher percentage of outliers are removed from the data. (optional; default is .15)
- `trial_recode_SD` recodes outlying reaction times to the nearest non-outlying value, with outliers defined as reaction times deviating more than a certain number of standard deviations from the participant's mean. Required argument:
  - `trialsd` - trials deviating more than this many standard deviations from the mean are classified as outliers.
- `trial_prune_percent_subject` and `trial_prune_percent_sample` remove trials below and/or above certain percentiles, on a subject-by-subject basis or sample-wide, respectively. The following arguments are available:
  - `lowerpercent` and `uppperpercent` (optional; defaults are .01 and .99).

`errortrialfunc` Function (without brackets or quotes) to apply to an error trial.

- `prune_nothing` removes no errors (default).
- `error_replace_blockmeanplus` replaces error trial reaction times with the block mean, plus an arbitrary extra quantity. If used, the following additional arguments are required:

  - `blockvar` - Quoted name of the block variable (mandatory)
  - `errorvar` - Quoted name of the error variable, where errors are 1 or TRUE and correct trials are 0 or FALSE (mandatory)
  - `errorbonus` - Amount to add to the reaction time of error trials. Default is 0.6 (recommended by `Greenwald,Nosek,& Banaji,2003`)
- `error_prune_dropcases` removes errors and drops participants if they have more errors than a given percentage. The following arguments are available:
  - `errorvar` - Quoted name of the error variable, where errors are 1 or TRUE and correct trials are 0 or FALSE (mandatory)
  - `maxerrors` - participants with a higher percentage of errors are excluded from the dataset. Default is .15.

`...` Other arguments, to be passed on to the algorithm or outlier rejection functions (see arguments above)

Examples

```
#Compute the correlation between relevant-feature and irrelevant-feature AAT scores
ds<-erotica[erotica$correct==1,]
relevant <- aat_compute(ds=ds[ds$is_irrelevant==0,],
                        pullvar="is_pull",targetvar="is_target",
                        rtvar="RT",subjvar="subject",
                        trialdropfunc="trial_prune_3SD",
                        algorithm="aat_doublemediandiff")

irrelevant <- aat_compute(ds=ds[ds$is_irrelevant==1,],
                        pullvar="is_pull",targetvar="is_target",
                        rtvar="RT",subjvar="subject",
                        trialdropfunc="trial_prune_3SD",
                        algorithm="aat_doublemediandiff")

comparison.df <- merge(relevant, irrelevant, by = "subject")
cor(comparison.df$ab.x, comparison.df$ab.y)
# 0.1145726
```

`aat_splithalf` *Compute the bootstrapped split-half reliability for approach-avoidance task data*

Description

Compute bootstrapped split-half reliability for approach-avoidance task data.

Usage

```
aat_splithalf(
  ds,
  subjvar,
  pullvar,
  targetvar = NULL,
  rtvar,
  iters,
  algorithm = c("aat_doublemeandiff", "aat_doublemediandiff", "aat_dscore",
    "aat_dscore_multiblock", "aat_regression", "aat_standardregression",
    "aat_doublemedianquotient", "aat_doublemeanquotient", "aat_singlemeandiff",
    "aat_singlemediandiff"),
 trialdropfunc = c("prune_nothing", "trial_prune_3SD", "trial_prune_SD_dropcases",
  "trial_recode_SD", "trial_prune_percent_subject", "trial_prune_percent_sample"),
  errortrialfunc = c("prune_nothing", "error_replace_blockmeanplus",
    "error_prune_dropcases"),
  casedropfunc = c("prune_nothing", "case_prune_3SD"),
  plot = TRUE,
  include.raw = FALSE,
  parallel = TRUE,
  ...
)

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

## S3 method for class 'aat_splithalf'
plot(x, type = c("median", "minimum", "maximum", "random"), ...)
```

Arguments

`ds`	a longformat data.frame
`subjvar`	Quoted name of the participant identifier column
`pullvar`	Quoted name of the column indicating pull trials. Pull trials should either be represented by 1, or by the second level of a factor.
`targetvar`	Name of the column indicating trials featuring the target stimulus. Target stimuli should either be represented by 1, or by the second level of a factor.
`rtvar`	Name of the reaction time column.
`iters`	Total number of desired iterations. At least 200 are recommended for reasonable confidence intervals; If you want to see plots of your data, 1 iteration is enough.
`algorithm`	Function (without brackets or quotes) to be used to compute AAT scores. See Algorithms for a list of usable algorithms.
`trialdropfunc`	Function (without brackets or quotes) to be used to exclude outlying trials in each half. The way you handle outliers for the reliability computation should mimic the way you do it in your regular analyses. It is recommended to exclude outlying trials when computing AAT scores using the mean double-dfference scores and regression scoring approaches, but not when using d-scores or median double-difference scores.

- `prune_nothing` excludes no trials (default)
- `trial_prune_3SD` excludes trials deviating more than 3SD from the mean per participant.
- `trial_prune_SD_dropcases` removes trials deviating more than a specific number of standard deviations from the participant's mean, and removes participants with an excessive percentage of outliers. Required arguments:
  - `trialsd` - trials deviating more than `trialsd` standard deviations from the participant's mean are excluded (optional; default is 3)

  - `maxoutliers` - participants with a higher percentage of outliers are removed from the data. (optional; default is .15)
- `trial_recode_SD` recodes outlying reaction times to the nearest non-outlying value, with outliers defined as reaction times deviating more than a certain number of standard deviations from the participant's mean. Required argument:
  - `trialsd` - trials deviating more than this many standard deviations from the mean are classified as outliers.
- `trial_prune_percent_subject` and `trial_prune_percent_sample` remove trials below and/or above certain percentiles, on a subject-by-subject basis or sample-wide, respectively. The following arguments are available:
  - `lowerpercent` and `uppperpercent` (optional; defaults are .01 and .99).

`errortrialfunc` Function (without brackets or quotes) to apply to an error trial.

- `prune_nothing` removes no errors (default).
- `error_replace_blockmeanplus` replaces error trial reaction times with the block mean, plus an arbitrary extra quantity. If used, the following additional arguments are required:
  - `blockvar` - Quoted name of the block variable (mandatory)
  - `errorvar` - Quoted name of the error variable, where errors are 1 or TRUE and correct trials are 0 or FALSE (mandatory)
  - `errorbonus` - Amount to add to the reaction time of error trials. Default is 0.6 (recommended by `Greenwald,Nosek,& Banaji,2003`)
- `error_prune_dropcases` removes errors and drops participants if they have more errors than a given percentage. The following arguments are available:
  - `errorvar` - Quoted name of the error variable, where errors are 1 or TRUE and correct trials are 0 or FALSE (mandatory)
  - `maxerrors` - participants with a higher percentage of errors are excluded from the dataset. Default is .15.

`casedropfunc` Function (without brackets or quotes) to be used to exclude outlying participant scores in each half. The way you handle outliers here should mimic the way you do it in your regular analyses.

- `prune_nothing` excludes no participants (default)
- `case_prune_3SD` excludes participants deviating more than 3SD from the sample mean.

`plot` Create a scatterplot of the AAT scores computed from each half of the data from the last iteration. This is highly recommended, as it helps to identify outliers that can inflate or diminish the reliability.

`include.raw` logical indicating whether raw split-half data should be included in the output object.

`parallel` If TRUE (default), will use parallel computing to compute results faster. If a doParallel backend has not been registered beforehand, this function will register a cluster and stop it after finishing, which takes some extra time.

`...` Other arguments, to be passed on to the algorithm or outlier rejection functions (see arguments above)

x	an aat_splithalf object
type	Character argument indicating which iteration should be chosen. Must be an abbreviation of "median" (default), "minimum", "maximum", or "random".

Value

A list, containing the mean bootstrapped split-half reliability, bootstrapped 95 a list of data.frames used over each iteration, and a vector containing the split-half reliability of each iteration.

Author(s)

Sercan Kahveci

See Also

q_reliability

Examples

```
split <- aat_splithalf(ds=erotica[erotica$is_irrelevant==0,],
                       subjvar="subject",pullvar="is_pull",targetvar="is_target",
                       rtvar="RT",iters=10,trialdropfunc="trial_prune_3SD",
                       casedropfunc="case_prune_3SD",algorithm="aat_dscore",
                       plot=FALSE,parallel=FALSE)

print(split)
#Mean reliability: 0.521959
#Spearman-Brown-corrected r: 0.6859041
#95%CI: [0.4167018, 0.6172474]

plot(split)


#Regression Splithalf
aat_splithalf(ds=erotica[erotica$is_irrelevant==0,],
              subjvar="subject", pullvar="is_pull", targetvar="is_target",
              rtvar="RT", iters=10, trialdropfunc="trial_prune_3SD",
              casedropfunc="case_prune_3SD", algorithm="aat_regression",
              formula = RT ~ is_pull * is_target, aatterm = "is_pull:is_target",
              plot=FALSE, parallel=FALSE)
#Mean reliability: 0.5313939
#Spearman-Brown-corrected r: 0.6940003
#95%CI: [0.2687186, 0.6749176]
```

Algorithms *AAT score computation algorithms*

Description

- `aat_doublemeandiff` computes a mean-based double-difference score:
  `(mean(push_target) -mean(pull_target)) -(mean(push_control) -mean(pull_control))`
- `aat_doublemediandiff` computes a median-based double-difference score:
  `(median(push_target) -median(pull_target)) -(median(push_control) -median(pull_control))`
- `aat_dscore` computes D-scores for a 2-block design (see Greenwald, Nosek, and Banaji, 2003):
  `((mean(push_target) -mean(pull_target)) -(mean(push_control) -mean(pull_control))) / sd(participant_reaction_times)`
- `aat_dscore_multiblock` computes D-scores for pairs of sequential blocks and averages the resulting score (see Greenwald, Nosek, and Banaji, 2003). Requires extra `blockvar` argument, indicating the name of the block variable.
- `aat_regression` and `aat_standardregression` fit regression models to participants' reaction times and extract a term that serves as AAT score. `aat_regression` extracts the raw coefficient, equivalent to a mean difference score. `aat_standardregression` extracts the t-score of the coefficient, standardized on the basis of the variability of the participant's reaction times. These algorithms can be used to regress nuisance variables out of the data before computing AAT scores. When using these functions, additional arguments must be provided:
  - `formula` - a formula to fit to the data
  - `aatterm` - the term within the formula that indicates the approach bias; this is usually the interaction of the pull and target terms.
- `aat_doublemeanquotient` and `aat_doublemedianquotient` compute a log-transformed ratio of approach to avoidance for both stimulus categories and subtract these ratios:
  `log(mean(pull_target) / mean(push_target)) -log(mean(pull_control) / mean(push_control))`
- `aat_singlemeandiff` and `aat_singlemediandiff` subtract the mean or median approach reaction time from the mean or median avoidance reaction time. These algorithms are only sensible if the supplied data contain a single stimulus category.

Usage

```
aat_doublemeandiff(ds, subjvar, pullvar, targetvar, rtvar, ...)

aat_doublemediandiff(ds, subjvar, pullvar, targetvar, rtvar, ...)

aat_dscore(ds, subjvar, pullvar, targetvar, rtvar, ...)

aat_dscore_multiblock(ds, subjvar, pullvar, targetvar, rtvar, blockvar, ...)

aat_regression(ds, subjvar, formula, aatterm, ...)

aat_standardregression(ds, subjvar, formula, aatterm, ...)

aat_doublemedianquotient(ds, subjvar, pullvar, targetvar, rtvar, ...)

aat_doublemeanquotient(ds, subjvar, pullvar, targetvar, rtvar, ...)

aat_singlemeandiff(ds, subjvar, pullvar, rtvar, ...)

aat_singlemediandiff(ds, subjvar, pullvar, rtvar, ...)
```

Arguments

`ds`	A long-format data.frame
`subjvar`	Column name of the participant identifier variable
`pullvar`	Column name of the movement variable (0: avoid; 1: approach)
`targetvar`	Column name of the stimulus category variable (0: control stimulus; 1: target stimulus)
`rtvar`	Column name of the reaction time variable
`...`	Other arguments passed on by functions (ignored)
`blockvar`	name of the variable indicating block number
`formula`	A regression formula to fit to the data to compute an AAT score
`aatterm`	A character naming the formula term representing the approach bias. Usually this is the interaction of the movement-direction and stimulus-category terms.

Value

A data.frame containing participant number and computed AAT score.

`erotica` *AAT examining approach bias for erotic stimuli*

Description

AAT

Usage

```
erotica
```

Format

An object of class `"data.frame"`

Source

osf.io repository

References

Kahveci, S., Van Bockstaele, B.D., & Wiers, R.W. (in preparation). Pulling for Pleasure? Erotic Approach-Bias Associated With Porn Use, Not Problems. DOI:10.17605/OSF.IO/6H2RJ

`Preprocessing` *Pre-processing rules*

Description

These are pre-processing rules that can be used in aat_splithalf, aat_bootstrap, and aat_compute.

- The following rules are to be used for the `trialdropfunc` argument. The way you handle outliers for the reliability computation and bootstrapping more broadly should mimic the way you do it in your regular analyses. It is recommended to exclude outlying trials when computing AAT scores using the mean double-dfference scores and regression scoring approaches, but not when using d-scores or median double-difference scores.
  - `prune_nothing` excludes no trials (default)
  - `trial_prune_3SD` excludes trials deviating more than 3SD from the mean per participant.
  - `trial_prune_SD_dropcases` removes trials deviating more than a specific number of standard deviations from the participant's mean, and removes participants with an excessive percentage of outliers. Required arguments:
    - `trialsd` - trials deviating more than `trialsd` standard deviations from the participant's mean are excluded (optional; default is 3)
    - `maxoutliers` - participants with a higher percentage of outliers are removed from the data. (optional; default is .15)
  - `trial_recode_SD` recodes outlying reaction times to the nearest non-outlying value, with outliers defined as reaction times deviating more than a certain number of standard deviations from the participant's mean. Required argument:
    - `trialsd` - trials deviating more than this many standard deviations from the mean are classified as outliers.
  - `trial_prune_percent_subject` and `trial_prune_percent_sample` remove trials below and/or above certain percentiles, on a subject-by-subject basis or sample-wide, respectively. The following arguments are available:
    - `lowerpercent` and `uppperpercent` (optional; defaults are .01 and .99).
- The following pre-procesing rules are to be used for the `errortrialfunc` argument. They determine what is to be done with errors - remove or recode?
  - `prune_nothing` removes no errors (default).
  - `error_replace_blockmeanplus` replaces error trial reaction times with the block mean, plus an arbitrary extra quantity. If used, the following additional arguments are required:
    - `blockvar` - Quoted name of the block variable (mandatory)
    - `errorvar` - Quoted name of the error variable, where errors are 1 or TRUE and correct trials are 0 or FALSE (mandatory)
    - `errorbonus` - Amount to add to the reaction time of error trials. Default is 0.6 (recommended by `Greenwald,Nosek,& Banaji,2003`)
  - `error_prune_dropcases` removes errors and drops participants if they have more errors than a given percentage. The following arguments are available:
    - `errorvar` - Quoted name of the error variable, where errors are 1 or TRUE and correct trials are 0 or FALSE (mandatory)

    * `maxerrors` - participants with a higher percentage of errors are excluded from the dataset. Default is .15.

- These are pre-processing rules to be used for the `casedropfunc` argument. The way you handle outliers here should mimic the way you do it in your regular analyses.
  - `prune_nothing` excludes no participants (default)
  - `case_prune_3SD` excludes participants deviating more than 3SD from the sample mean.

Usage

```
prune_nothing(ds, ...)

trial_prune_percent_subject(
  ds,
  subjvar,
  rtvar,
  lowerpercent = 0.01,
  upperpercent = 0.99,
  ...
)

trial_prune_percent_sample(
  ds,
  rtvar,
  lowerpercent = 0.01,
  upperpercent = 0.99,
  ...
)

trial_prune_3SD(ds, subjvar, rtvar, ...)

trial_prune_SD_dropcases(
  ds,
  subjvar,
  rtvar,
  trialsd = 3,
  maxoutliers = 0.15,
  ...
)

trial_recode_SD(ds, subjvar, rtvar, trialsd = 3, ...)

case_prune_3SD(ds, ...)

error_replace_blockmeanplus(
  ds,
  subjvar,
  rtvar,
  blockvar,
  errorvar,
  errorbonus,
  ...
)

error_prune_dropcases(ds, subjvar, errorvar, maxerrors = 0.15, ...)
```

Arguments

`ds`	A data.frame.
`...`	Other arguments (ignored).
`subjvar`	The name of the subject variable.
`rtvar`	The name of the reaction time variable.
`lowerpercent, upperpercent`	for `trial_prune_percent_subject` and `trial_prune_percent_sample`, the lower and upper proportions beyond which trials are considered outliers and removed (defaults to .01 and .99).
`trialsd`	The amount of deviation from the participant mean (in SD) after which a trial is considered an outlier and excluded (defaults to 3).
`maxoutliers`	for `trial_prune_SD_dropcases`, the maximum percentage of outliers, after which a participant is excluded from the data.
`blockvar`	The name of the block variable.
`errorvar`	The name of the error variable.
`errorbonus`	for `error_replace_blockmeanplus`, the amount of seconds to add to the block mean and use as a replacement for error trial reaction times (default is 0.6).
`maxerrors`	for `error_prune_dropcases`, the maximum percentage of errors, after which a participant is excluded from the data.

`q_reliability` *Compute psychological experiment reliability*

Description

This function can be used to compute an exact reliability score for a psychological task whose results involve a difference score. The resulting intraclass correlation coefficient is equivalent to the average all possible split-half reliability scores. It ranges from -1 to 1, with -1 implying that all variance in the data is explained by within-subjects variability, 1 implying that all variance is explained by between-subjects variability, and 0 implying that within-subjects and between-subjects variability contribute equally to the total variance in the sample.

Usage

```
q_reliability(ds, subjvar, formula, aatterm = NA)

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

## S3 method for class 'qreliability'
plot(x, ...)
```

Arguments

`ds`	a long-format data.frame
`subjvar`	name of the subject variable
`formula`	a formula predicting the participant's reaction time using trial-level variables such as movement direction and stimulus category
`aaterm`	a string denoting the term in the formula that contains the participant's approach bias
`x`	a `qreliability` object
`...`	Other arguments passed to the generic `print` and `plot` functions.

Value

a qreliability object, containing the reliability coefficient, and a data.frame with participants' bias scores and score variance.

Author(s)

Sercan Kahveci

Examples

```
# Double-difference score reliability
q_reliability(ds=erotica,subjvar="subject",
             formula= RT ~ is_pull * is_target, aaterm = "is_pull:is_target")

# Single-difference reliability for target stimuli
q_reliability(ds=erotica[erotica$is_target ==1,],subjvar="subject",
             formula= RT ~ is_pull, aaterm = "is_pull")

# Reliability of the mean reaction time of approaching target stimuli (no difference score)
q_reliability(ds=erotica[erotica$is_target ==1 & erotica$is_pull ==1,],subjvar="subject",
             formula= RT ~ 1, aaterm = "1")
```

`SpearmanBrown` *Spearman-Brown corrections for Correlation Coefficients*

Description

Perform a Spearman-Brown correction on the provided correlation score.

Usage

```
SpearmanBrown(
  corr,
  ntests = 2,
  fix.negative = c("nullify", "bilateral", "none")
)
```

Arguments

`corr`	To-be-corrected correlation coefficient
`ntests`	An integer indicating how many times larger the full test is, for which the corrected correlation coefficient is being computed. When `ntests=2`, the formula will compute what the correlation coefficient would be if the test were twice as long.
`fix.negative`	Determines how to deal with a negative value. "nullify" sets it to zero, "bilateral" applies the correction as if it were a positive number, and then sets it to negative. "none" gives the raw value.

Details

Correct a correlation coefficient for being based on only a subset of the data.

Value

Spearman-Brown-corrected correlation coefficient.

Index