

Package ‘scan’

September 22, 2021

Type Package

Title Single-Case Data Analyses for Single and Multiple Baseline Designs

Version 0.53

Date 2021-9-22

Author Juergen Wilbert [cre, aut],
Timo Lueke [aut]

Imports stats, nlme, utils, methods, graphics, car, knitr, kableExtra,
readxl, mblm, meta, magrittr

Depends R (>= 3.5.0)

Description A collection of procedures for analysing, visualising, and managing single-case data. These include piecewise linear regression models, multilevel models, overlap indices (PND, PEM, PAND, PET, tauU, baseline corrected tau, CDC), and randomization tests. Data preparation functions support outlier detection, handling missing values, scaling, truncating, rank transformation, and smoothing. An exporting function helps to generate html and latex tables in a publication friendly style. More details can be found at <<https://jazznbass.github.io/scan-Book/>>.

License GPL (>= 3)

URL <https://github.com/jazznbass/scan/>,
<https://jazznbass.github.io/scan-Book/>

BugReports <https://github.com/jazznbass/scan/issues/>

LazyData true

Encoding UTF-8

RoxygenNote 7.1.2

NeedsCompilation no

Maintainer Juergen Wilbert <juergen.wilbert@uni-potsdam.de>

Repository CRAN

Date/Publication 2021-09-22 09:40:02 UTC

R topics documented:

scan-package	3
.inheritParams	3
add_12	4
as.data.frame.scdf	5
as_scdf	6
autocorr	6
Beretvas2008	7
cdc	9
check_scdf	11
combine	11
convert	12
corrected_tau	13
describe	14
estimate_design	15
export	17
fill_missing	19
hplm	21
is.scdf	23
mplm	23
nap	25
outlier	26
overlap	27
pand	29
pem	31
pet	32
plm	34
plot.scdf	36
pnd	39
power_test	40
print.sc	41
print.scdf	43
random	44
rand_test	47
ranks	50
rci	51
readSC	52
sample_names	54
scdf	54
scdf_attr	57
select_cases	58
select_phases	58
set_vars	59
shift	60
smooth_cases	61
standardize	62
style_plot	63

<i>scan-package</i>	3
subset.scdf	65
summary.scdf	66
tau_u	66
trend	68
truncate_phase	69
writeSC	71
\$.scdf	72
%>%	72
Index	73

scan-package	<i>Single-Case Data Analyses</i>
--------------	----------------------------------

Description

A collection of procedures for analysing, visualising, and managing single-case data.

Author(s)

Juergen Wilbert [aut, cre]

.inheritParams	<i>Dummy function to inherit global descriptions of parameters</i>
----------------	--

Description

Dummy function to inherit global descriptions of parameters

Usage

```
.inheritParams(
  data,
  scdf,
  dvar,
  mvar,
  pvar,
  decreasing,
  phases,
  model,
  trend,
  level,
  slope,
  ...
)
```

Arguments

data	A single-case data frame. See scdf to learn about this format.
scdf	A single-case data frame. See scdf to learn about this format.
dvar	Character string with the name of the dependent variable. Defaults to the attributes in the scdf file.
mvar	Character string with the name of the measurement time variable. Defaults to the attributes in the scdf file.
pvar	Character string with the name of the phase variable. Defaults to the attributes in the scdf file.
decreasing	If you expect data to be lower in the B phase, set decreasing = TRUE. Default is decreasing = FALSE.
phases	A vector of two characters or numbers indicating the two phases that should be compared. E.g., phases = c("A", "C") or phases = c(2, 4) for comparing the second to the fourth phase. Phases could be combined by providing a list with two elements. E.g., phases = list(A = c(1, 3), B = c(2, 4)) will compare phases 1 and 3 (as A) against 2 and 4 (as B). Default is phases = c("A", "B").
model	Model used for calculating the slope parameter (see Huitema & McKean, 2000). Default is model = "B&L-B". Possible values are: "B&L-B", "H-M", "Mohr#1", "Mohr#2", "JW", "JW2", and "Manly".
trend	A logical indicating if a trend parameters is included in the model.
level	A logical indicating if a level parameters is included in the model.
slope	A logical indicating if a slope parameters is included in the model.
...	Further arguments passed to the function.

 add_l2

Add level 2

Description

Add level 2 variables from a level 2 dataset to an scdf file

Usage

```
add_l2(data, data_l2, cvar = "case")
```

Arguments

data	A single-case data frame. See scdf to learn about this format.
data_l2	A level 2 dataset.
cvar	Character string with the name of the "case" variable in the L2 dataset (default is 'case').

Value

An scdf

Examples

```
Leidig2018 %>% add_l2(Leidig2018_l2)
```

as.data.frame.scdf	<i>Creating a long format data frame from several single-case data frames (scdf).</i>
--------------------	---

Description

The `as.data.frame` function transposes an scdf into one long data frame. Additionally, a data frame can be merged that includes level 2 data of the subjects. This might be helpful to prepare data to be used with other packages than scan.

Usage

```
## S3 method for class 'scdf'  
as.data.frame(x, ..., l2 = NULL, id = "case")  
  
longSCDF(...)
```

Arguments

x	An scdf object
...	Not implemented
l2	A data frame providing additional variables at Level 2. The scdf has to have names for all cases and the Level 2 data frame has to have a column with corresponding case names.
id	Variable name of the Level 2 data frame that contains the case names.

Value

Returns one data frame with data of all single-cases structured by the case variable.

Author(s)

Juergen Wilbert

See Also

Other data manipulation functions: [fill_missing\(\)](#), [outlier\(\)](#), [ranks\(\)](#), [shift\(\)](#), [smooth_cases\(\)](#), [standardize\(\)](#), [truncate_phase\(\)](#)

Examples

```
## Convert the list of three single-case data frames from Grosche (2011) into one long data frame
Grosche2011
Grosche2011_long <- as.data.frame(Grosche2011)
Grosche2011_long

## Combine an scdf with data for 12
Leidig2018_long <- as.data.frame(Leidig2018, 12 = Leidig2018_12)
names(Leidig2018_long)
summary(Leidig2018_long)
```

as_scdf	<i>as_scdf Converts a data frame to an scdf object</i>
---------	--

Description

as_scdf Converts a data frame to an scdf object

Usage

```
as_scdf(object)
```

Arguments

object	A scdf object
--------	---------------

autocorr	<i>Autocorrelation for single-case data</i>
----------	---

Description

The autocorr function calculates autocorrelations within each phase and across all phases.

Usage

```
autocorr(data, dvar, pvar, mvar, lag.max = 3, ...)

autocorrSC(...)
```

Arguments

<code>data</code>	A single-case data frame. See scdf to learn about this format.
<code>dvar</code>	Character string with the name of the dependent variable. Defaults to the attributes in the scdf file.
<code>pvar</code>	Character string with the name of the phase variable. Defaults to the attributes in the scdf file.
<code>mvar</code>	Character string with the name of the measurement time variable. Defaults to the attributes in the scdf file.
<code>lag.max</code>	The lag up to which autocorrelations will be computed. Default is <code>lag.max = 3</code> .
<code>...</code>	Further arguments passed to the acf function

Value

A data frame containing separate autocorrelations for each phase and for all phases (for each single-case). If `lag.max` exceeds the length of a phase minus one, NA is returned for this cell.

Author(s)

Juergen Wilbert

See Also

[trend](#), [plm](#), [acf](#)

Examples

```
## Compute autocorrelations for a list of four single-cases up to lag 2.  
autocorr(Huber2014, lag.max = 2)
```

Beretvas2008

Single-case example data

Description

The scan package comes with a set of fictitious and authentic single-case study data, by courtesy of the particular authors.

Usage

```
Beretvas2008
```

Format

An object of class `scdf` (inherits from `list`) of length 1.

Value

- Beretvas2008 Fictitious single-case intervention study. **Reference:** Beretvas, S., & Chung, H. (2008). An evaluation of modified R2-change effect size indices for single-subject experimental designs. *Evidence-Based Communication Assessment and Intervention, 2*, 120-128.
- Borckardt2014 Fictitious daily pain ratings evaluating a psychological treatment. **Reference:** Borckardt, J. J., & Nash, M. R. (2014). Simulation modelling analysis for small sets of single-subject data collected over time. *Neuropsychological Rehabilitation, 24*, 492-506.
- Huitema2000 Fictitious single-case intervention study. **Reference:** Huitema, B. E., & McKean, J. W. (2000). Design specification issues in time-series intervention models. *Educational and Psychological Measurement, 60*, 38-58.
- Waddell2011 Fictitious single-case intervention study. **Reference:** Waddell, D. E., Nassar, S. L., & Gustafson, S. A. (2011). Single-Case Design in Psychophysiological Research: Part II: Statistical Analytic Approaches. *Journal of Neurotherapy, 15*, 160-169.
- byHeart2011 Multiple-baseline (11 cases) intervention study on flash card vocabulary learning by Juergen Wilbert.
- Grosche2011 Multiple-baseline (3 cases) intervention study on a direct-instructive reading intervention. **Reference:** Grosche, M. (2011). Effekte einer direkt-instruktiven Foerderung der Lesegenauigkeit. *Empirische Sonderpaedagogik, 3*, 147-161.
- Grosche2014 Multiple-baseline (3 cases x 3 materials) intervention study on a reading intervention. **Reference:** Grosche, M., Lueke, T., & Wilbert, J. (in prep.).
- GruenkeWilbert2014 Multiple-baseline (6 cases) intervention study on story mapping. **Reference:** Gruenke, M., Wilbert, J., & Stegemann-Calder, K. (2013). Analyzing the effects of story mapping on the reading comprehension of children with low intellectual abilities. *Learning Disabilities: A Contemporary Journal, 11*, 51-64.
- Huber2014 Multiple-baseline (4 cases) intervention study on behavioral compliance. Scores refer to compliant behavior in percent. **Reference:** Huber, C. (in prep.).
- Lenz2013 Fictitious example from the paper Lenz, A. S. (2013). Calculating Effect Size in Single-Case Research: A Comparison of Nonoverlap Methods. *Measurement and Evaluation in Counseling and Development, 46*(1), 64-73.
- Leidig2018
- Leidig2018_l2
- SSDforR2017 Example from the R package SSDforR.
- Parker2011 Example from Parker, R. I., Vannest, K. J., Davis, J. L., & Sauber, S. B. (2011). Combining Nonoverlap and Trend for Single-Case Research: Tau-U. *Behavior Therapy, 42*(2), 284-299. <https://doi.org/10.1016/j.beth.2010.08.006>

Author(s)

Juergen Wilbert

Description

The `cdc` function applies the Conservative Dual-Criterion Method (Fisher, Kelley, & Lomas, 2003) to `scdf` objects. It compares phase B data points to both phase A mean and trend (OLS, bi-split, tri-split) with an additional increase/decrease of .25 SD. A binomial test against a 50/50 distribution is computed and p-values below .05 are labeled "systematic change".

Usage

```
cdc(
  data,
  dvar,
  pvar,
  mvar,
  decreasing = FALSE,
  trend_method = "OLS",
  conservative = 0.25,
  phases = c(1, 2)
)
```

Arguments

<code>data</code>	A single-case data frame. See scdf to learn about this format.
<code>dvar</code>	Character string with the name of the dependent variable. Defaults to the attributes in the <code>scdf</code> file.
<code>pvar</code>	Character string with the name of the phase variable. Defaults to the attributes in the <code>scdf</code> file.
<code>mvar</code>	Character string with the name of the measurement time variable. Defaults to the attributes in the <code>scdf</code> file.
<code>decreasing</code>	If you expect data to be lower in the B phase, set <code>decreasing = TRUE</code> . Default is <code>decreasing = FALSE</code> .
<code>trend_method</code>	Method used to calculate the trend line. Default is <code>trend_method = "OLS"</code> . Possible values are: "OLS", "bisplit", and "trisplit". "bisplit", and "trisplit" should only be used for cases with at least five data-points in both relevant phases.
<code>conservative</code>	The CDC method adjusts the original mean and trend lines by adding (expected increase) or subtracting (expected decrease) an additional .25 SD before evaluating phase B data. Default is the CDC method with <code>conservative = .25</code> . To apply the Dual-Criterion (DC) method, set <code>conservative = 0</code> .
<code>phases</code>	A vector of two characters or numbers indicating the two phases that should be compared. E.g., <code>phases = c("A", "C")</code> or <code>phases = c(2, 4)</code> for comparing the second to the fourth phase. Phases could be combined by providing a list with two elements. E.g., <code>phases = list(A = c(1, 3), B = c(2, 4))</code> will compare phases 1 and 3 (as A) against 2 and 4 (as B). Default is <code>phases = c("A", "B")</code> .

Value

cdc	CDC Evaluation based on a p-value below .05.
cdc_exc	Number of phase B datapoints indicating expected change.
cdc_nb	Number of phase B datapoints.
cdc_p	P value of Binomial Test.
cdc_all	Overall CDC Evaluation based on all instances/cases of a Multiple Baseline Design.
N	Number of cases.
decreasing	Logical argument from function call (see Arguments above).
conservative	Numeric argument from function call (see Arguments above).
case_names	Assigned name of single-case.
phases	-

Author(s)

Timo Lueke

References

Fisher, W. W., Kelley, M. E., & Lomas, J. E. (2003). Visual Aids and Structured Criteria for Improving Visual Inspection and Interpretation of Single-Case Designs. *Journal of Applied Behavior Analysis*, 36, 387-406. <https://doi.org/10.1901/jaba.2003.36-387>

Examples

```
## Apply the CDC method (standard OLS line)
design <- design_rSC(n = 1, slope = 0.2)
dat <- rSC(design, seed = 42)
cdc(dat)

## Apply the CDC with Koenig's bi-split and an expected decrease in phase B.
cdc(exampleAB_decreasing, decreasing = TRUE, trend_method = "bisplit")

## Apply the CDC with Tukey's tri-split, comparing the first and fourth phase.
cdc(exampleABAB, trend_method = "trisplit", phases = c(1,4))

## Apply the Dual-Criterion (DC) method (i.e., mean and trend without shifting).
cdc(exampleAB_decreasing, decreasing = TRUE, trend_method = "bisplit", conservative = 0)
```

check_scdf	<i>Validity check for an scdf object</i>
------------	--

Description

Validity check for an scdf object

Usage

```
check_scdf(object)
```

Arguments

object An scdf object

Value

TRUE or list with error and warning messages.

Examples

```
check_scdf(exampleAB)
check_scdf(c(exampleAB, exampleABC))
```

combine	<i>Combine single-case data frames</i>
---------	--

Description

Combine single-case data frames

Usage

```
combine(..., dvar = NULL, pvar = NULL, mvar = NULL)

## S3 method for class 'scdf'
c(...)
```

Arguments

...	scdf objects
dvar	Character string. Name of the dependent variable. Defaults to the dependent variable of the first case provided.
pvar	Character string. Name of the phase variable. Defaults to the phase variable of the first case provided.
mvar	Character string. Name of the measurement-time variable. Defaults to the measurement-time variable of the first case provided.

Value

A scdf. If not set differently, the attributes of this scdf are copied from the first scdf provided (i.e the first argument of the function).

convert	<i>Covert Converts an scdf object to a text syntax</i>
---------	--

Description

Covert Converts an scdf object to a text syntax

Usage

```
convert(scdf, file = "", study_name = "study")
```

Arguments

scdf	A single-case data frame. See scdf to learn about this format.
file	A filename for exporting the syntax.
study_name	Character string. Name of the single case study name.

Examples

```
filename <- tempfile()
convert(exampleABC, file = filename)
source(filename)
all.equal(study, exampleABC)
unlink(filename)
```

corrected_tau	<i>Baseline corrected tau</i>
---------------	-------------------------------

Description

Kendalls tau correlation for the dependent variable and the phase variable is calculated after correcting for a baseline trend.

Usage

```
corrected_tau(
  data,
  dvar,
  pvar,
  mvar,
  phases = c(1, 2),
  alpha = 0.05,
  continuity = TRUE,
  repeated = TRUE
)

corrected_tauSC(...)
```

Arguments

data	A single-case data frame. See scdf to learn about this format.
dvar	Character string with the name of the dependent variable. Defaults to the attributes in the scdf file.
pvar	Character string with the name of the phase variable. Defaults to the attributes in the scdf file.
mvar	Character string with the name of the measurement time variable. Defaults to the attributes in the scdf file.
phases	A vector of two characters or numbers indicating the two phases that should be compared. E.g., <code>phases = c("A", "C")</code> or <code>phases = c(2, 4)</code> for comparing the second to the fourth phase. Phases could be combined by providing a list with two elements. E.g., <code>phases = list(A = c(1, 3), B = c(2, 4))</code> will compare phases 1 and 3 (as A) against 2 and 4 (as B). Default is <code>phases = c("A", "B")</code> .
alpha	Sets the p-value at and below which a baseline correction is applied.
continuity	If TRUE applies a continuity correction for calculating p
repeated	If TRUE applies the repeated median method for calculating slope and intercept (mb1m)
...	Further arguments passed to the function.

Details

This method has been proposed by Tarlow (2016). The baseline data are checked for a significant autocorrelation (based on Kendalls Tau). If so, a non-parameteric Theil-Sen regression is applied for the baseline data where the dependent values are regressed on the measurement time. The resulting slope information is then used to predict data of the B-phase. The dependent variable is now corrected for this baseline trend and the residuals of the Theil-Sen regression are taken for further calculations. Finally, Kendalls tau is calculated for the dependent variable and the dichotomous phase variable. The function here provides two extensions to this procedure: The more accurate Siegel repeated median regression is applied when `repeated = TRUE` and a continuity correction is applied when `continuity = TRUE` (both are the default settings).

References

Tarlow, K. R. (2016). An Improved Rank Correlation Effect Size Statistic for Single-Case Designs: Baseline Corrected Tau. *Behavior Modification*, 41(4), 427–467. <https://doi.org/10.1177/0145445516676750>

See Also

Other regression functions: [hplm\(\)](#), [mplm\(\)](#), [plm\(\)](#)

Other overlap functions: [nap\(\)](#), [overlap\(\)](#), [pand\(\)](#), [pem\(\)](#), [pet\(\)](#), [pnd\(\)](#), [tau_u\(\)](#)

Examples

```
dat <- scdf(c(A = 33,25,17,25,14,13,15, B = 15,16,16,5,7,9,6,5,3,3,8,11,7))
corrected_tau(dat)
```

describe

Descriptive statistics for single-case data

Description

The `describe` function provides common descriptive statistics for single-case data.

Usage

```
describe(data, dvar, pvar, mvar)
```

```
describeSC(...)
```

Arguments

<code>data</code>	A single-case data frame. See scdf to learn about this format.
<code>dvar</code>	Character string with the name of the dependent variable. Defaults to the attributes in the <code>scdf</code> file.
<code>pvar</code>	Character string with the name of the phase variable. Defaults to the attributes in the <code>scdf</code> file.

`mvar` Character string with the name of the measurement time variable. Defaults to the attributes in the scdf file.

`...` Further arguments passed to the function.

Details

`n` = number of measurements; `mis` = number of missing values; `m` = mean; `md` = median; `sd` = standard deviation; `mad` = median average deviation; `min` = minimum; `max` = maximum; `trend` = weight of depended variable regressed on time (values ~ mt).

Value

A list containing a data frame of descriptive statistics (descriptives); the cse design (design); the number of cases (N)

Author(s)

Juergen Wilbert

See Also

[overlap, plot.scdf](#)

Examples

```
## Descriptive statistics for a study of three single-cases
describe(Grosche2011)

## Descriptives of a three phase design
describe(exampleABC)

## Write descriptive statistics to .csv-file
study <- describe(Waddell2011)
write.csv(study$descriptives, file = tempfile())
```

estimate_design

Estimate single-case design

Description

This functions takes an scdf and extracts design parameters. The resulting object can be used to randomly create new scdf files with the same underlying parameters. This is usefull for monte-carlo studies and bootstrapping procedures.

Usage

```
estimate_design(
  data,
  dvar,
  pvar,
  mvar,
  m = NULL,
  s = NULL,
  rtt = NULL,
  between = TRUE,
  model = "JW",
  ...
)
```

Arguments

<code>data</code>	A single-case data frame. See scdf to learn about this format.
<code>dvar</code>	Character string with the name of the dependent variable. Defaults to the attributes in the scdf file.
<code>pvar</code>	Character string with the name of the phase variable. Defaults to the attributes in the scdf file.
<code>mvar</code>	Character string with the name of the measurement time variable. Defaults to the attributes in the scdf file.
<code>m</code>	The mean depicting the overall distribution of which all cases are a random sample of. <code>m</code> is estimated when <code>m = NULL</code> .
<code>s</code>	The standard deviation depicting the between case variance of the overall performance. If more than two single-cases are included in the scdf, the variance is estimated if <code>s</code> is set to <code>NULL</code> .
<code>rtt</code>	The reliability of the measurements. The reliability is estimated when <code>rtt = NULL</code> .
<code>between</code>	If <code>FALSE</code> trend, level, and slope effect estimations will be identical for each case. If <code>TRUE</code> effects are estimated for each case separately.
<code>model</code>	Model used for calculating the slope parameter (see Huitema & McKean, 2000). Default is <code>model = "B&L-B"</code> . Possible values are: <code>"B&L-B"</code> , <code>"H-M"</code> , <code>"Mohr#1"</code> , <code>"Mohr#2"</code> , <code>"JW"</code> , <code>"JW2"</code> , and <code>"Manly"</code> .
<code>...</code>	Further arguments passed to the <code>lm</code> function used for parameter estimation.

Value

A list of parameters for each single-case. Parameters include name, length, and starting measurementtime of each phase, trend level, and slope effects for each phase, mean, standarddeviation, and reliability for each case.

export

Export scan objects to html or latex

Description

This function is in an experimental status. Export creates html files of tables or displays them directly in the viewer pane of rstudio. When applied in rmarkdown, tables can also be created for pdf/latex output.

Usage

```
export(object, ...)  
  
## S3 method for class 'scdf'  
export(  
  object,  
  caption = NA,  
  footnote = NA,  
  filename = NA,  
  kable_styling_options = list(),  
  kable_options = list(),  
  cols,  
  ...  
)  
  
## S3 method for class 'sc_desc'  
export(  
  object,  
  caption = NA,  
  footnote = NA,  
  filename = NA,  
  kable_styling_options = list(),  
  kable_options = list(),  
  flip = FALSE,  
  ...  
)  
  
## S3 method for class 'sc_hplm'  
export(  
  object,  
  caption = NA,  
  footnote = NA,  
  filename = NA,  
  kable_styling_options = list(),  
  kable_options = list(),  
  round = 2,  
  nice = TRUE,
```

```
    ...
  )

## S3 method for class 'sc_plm'
export(
  object,
  caption = NA,
  footnote = NA,
  filename = NA,
  kable_styling_options = list(),
  kable_options = list(),
  round = 2,
  nice = TRUE,
  ...
)

## S3 method for class 'sc_overlap'
export(
  object,
  caption = NA,
  footnote = NA,
  filename = NA,
  kable_styling_options = list(),
  kable_options = list(),
  round = 2,
  flip = FALSE,
  ...
)

## S3 method for class 'sc_trend'
export(
  object,
  caption = NA,
  footnote = NA,
  filename = NA,
  kable_styling_options = list(),
  kable_options = list(),
  round = 2,
  flip = TRUE,
  ...
)

## S3 method for class 'sc_tauu'
export(
  object,
  caption = NA,
  footnote = NA,
  filename = NA,
```

```

    kable_styling_options = list(),
    kable_options = list(),
    ...
)

```

Arguments

object	An scdf or an object exported from a scan function.
...	Further Arguments passed to internal functions.
caption	Character string with table caption. If left NA (default) a caption will be created based on the exported object.
footnote	Character string with table footnote. If left NA (default) a footnote will be created based on the exported object.
filename	Character string with the filename. If a filename is provided the output will be written into this file.
kable_styling_options	list with arguments passed to the kable_styling function.
kable_options	list with arguments passed to the kable function.
cols	Defines which columns are included when a scdf is exported. It is either a vector with variable names or the string "main" will select the central variables.
flip	If TRUE, some objects are exported with rows and columns flipped.
round	value for the digits argument passed to the internally used round function.
nice	If set TRUE (default) output values are rounded and optimized for publication tables.

Value

Returns a specif formatted html (or latex).

fill_missing	<i>Replacing missing measurement times in single-case data</i>
--------------	--

Description

The fillmissingSC function replaces missing measurements in single-case data.

Usage

```

fill_missing(data, dvar, mvar, interpolation = "linear", na.rm = TRUE)

fillmissingSC(...)

```

Arguments

data	A single-case data frame. See scdf to learn about this format.
dvar	Character string with the name of the dependent variable. Defaults to the attributes in the scdf file.
mvar	Character string with the name of the measurement time variable. Defaults to the attributes in the scdf file.
interpolation	Alternative options not yet included. Default is interpolation = "linear".
na.rm	If set TRUE, NA values are also interpolated. Default is na.rm = TRUE.
...	Further arguments passed to the function.

Details

This procedure is recommended if there are gaps between measurement times (e.g. MT: 1, 2, 3, 4, 5, ... 8, 9) or explicitly missing values in your single-case data and you want to calculate overlap indices ([overlapSC](#)) or a randomization test ([randSC](#)).

Value

A single-case data frame (SCDF) with missing data points interpolated. See [scdf](#) to learn about the SCDF Format.

Author(s)

Juergen Wilbert

See Also

Other data manipulation functions: [as.data.frame.scdf\(\)](#), [outlier\(\)](#), [ranks\(\)](#), [shift\(\)](#), [smooth_cases\(\)](#), [standardize\(\)](#), [truncate_phase\(\)](#)

Examples

```
## In his study, Grosche (2011) could not realize measurements each single week for
## all participants. During the course of 100 weeks, about 20 measurements per person
## at different times were administered.
```

```
## Fill missing values in a single-case dataset with discontinuous measurement times
Grosche2011filled <- fill_missing(Grosche2011)
study <- c(Grosche2011[2], Grosche2011filled[2])
names(study) <- c("Original", "Filled")
plot(study)
```

```
## Fill missing values in a single-case dataset that are NA
Maggie <- rSC(design_rSC(level = list(0,1)), seed = 123)
Maggie_n <- Maggie
replace.positions <- c(10,16,18)
Maggie_n[[1]][replace.positions,"values"] <- NA
Maggie_f <- fill_missing(Maggie_n)
```

```
study <- c(Maggie, Maggie_n, Maggie_f)
names(study) <- c("original", "missing", "interpolated")
plot(study, marks = list(positions = replace.positions), style = "grid2")
```

hplm

Hierarchical piecewise linear model / piecewise regression

Description

The `hplm` function computes a hierarchical piecewise regression model.

Usage

```
hplm(
  data,
  dvar,
  pvar,
  mvar,
  model = "B&L-B",
  method = "ML",
  control = list(opt = "optim"),
  random.slopes = FALSE,
  lr.test = FALSE,
  ICC = TRUE,
  trend = TRUE,
  level = TRUE,
  slope = TRUE,
  fixed = NULL,
  random = NULL,
  update.fixed = NULL,
  data.l2 = NULL,
  ...
)
```

Arguments

<code>data</code>	A single-case data frame. See scdf to learn about this format.
<code>dvar</code>	Character string with the name of the dependent variable. Defaults to the attributes in the <code>scdf</code> file.
<code>pvar</code>	Character string with the name of the phase variable. Defaults to the attributes in the <code>scdf</code> file.
<code>mvar</code>	Character string with the name of the measurement time variable. Defaults to the attributes in the <code>scdf</code> file.
<code>model</code>	Model used for calculating the slope parameter (see Huitema & McKean, 2000). Default is <code>model = "B&L-B"</code> . Possible values are: <code>"B&L-B"</code> , <code>"H-M"</code> , <code>"Mohr#1"</code> , <code>"Mohr#2"</code> , <code>"JW"</code> , <code>"JW2"</code> , and <code>"Manly"</code> .

method	Method used to fit your model. Pass "REML" to maximize the restricted log-likelihood or "ML" for maximized log-likelihood. Default is "ML".
control	A list of settings for the estimation algorithm, replacing the default values passed to the function <code>lmeControl</code> of the <code>nlme</code> package.
random.slopes	If <code>random.slopes = TRUE</code> random slope effects of the level, trend, and treatment parameter are estimated.
lr.test	If set <code>TRUE</code> likelihood ratio tests are calculated comparing model with vs. without random slope parameters.
ICC	If <code>ICC = TRUE</code> an intraclass-correlation is estimated.
trend	A logical indicating if a trend parameters is included in the model.
level	A logical indicating if a level parameters is included in the model.
slope	A logical indicating if a slope parameters is included in the model.
fixed	Defaults to the fixed part of the standard piecewise regression model. The parameter phase followed by the phase name (e.g., <code>phaseB</code>) indicates the level effect of the corresponding phase. The parameter 'inter' followed by the phase name (e.g., <code>interB</code>) addresses the slope effect based on the method provide in the model argument (e.g., "B&L-B"). The formula can be changed for example to include further L1 or L2 variables into the regression model.
random	The random part of the model.
update.fixed	An easier way to change the fixed model part (e.g., <code>. ~ . + newvariable</code>).
data.l2	A dataframe providing additional variables at Level 2. The <code>scdf</code> File has to have names for all cases and the Level 2 dataframe has to have a column named 'cases' with the names of the cases the Level 2 variables belong to.
...	Further arguments passed to the <code>lme</code> function.

Value

<code>model</code>	List containing information about the applied model
<code>N</code>	Number of single-cases.
<code>formula</code>	A list containing the fixed and the random formulas of the <code>hplm</code> model.
<code>hplm</code>	Object of class <code>lme</code> containing the multilevel model
<code>model.0</code>	Object of class <code>lme</code> containing the Zero Model.
<code>ICC</code>	List containing intraclass correlation and test parameters.
<code>model.without</code>	Object of class <code>gls</code> containing the fixed effect model.

Author(s)

Juergen Wilbert

See Also

Other regression functions: [corrected_tau\(\)](#), [mplm\(\)](#), [plm\(\)](#)

Examples

```
## Compute hplm model on a MBD over fifty cases (restricted log-likelihood)
hplm(exampleAB_50, method = "REML", random.slopes = FALSE)

## Analyzing with additional L2 variables
hplm(Leidig2018, data.l2 = Leidig2018_l2,
     update.fixed = .~. + gender + migration + ITRF_TOTAL*phaseB,
     slope = FALSE, random.slopes = FALSE, lr.test = FALSE)
```

is.scdf	<i>scdf objects Tests for objects of type "scdf"</i>
---------	--

Description

scdf objects Tests for objects of type "scdf"

Usage

```
is.scdf(x)
```

Arguments

x An object to be tested

Value

Returns TRUE or FALSE depending on whether its argument is of scdf type or not.

mplm	<i>Multivariate Piecewise linear model / piecewise regression</i>
------	---

Description

The mplm function computes a multivariate piecewise regression model.

Usage

```
mplm(
  data,
  dvar,
  mvar,
  pvar,
  model = "B&L-B",
  trend = TRUE,
```

```

    level = TRUE,
    slope = TRUE,
    formula = NULL,
    update = NULL,
    na.action = na.omit,
    ...
)

```

Arguments

<code>data</code>	A single-case data frame. See scdf to learn about this format.
<code>dvar</code>	Character string with the name of the dependent variable. Defaults to the attributes in the scdf file.
<code>mvar</code>	Character string with the name of the measurement time variable. Defaults to the attributes in the scdf file.
<code>pvar</code>	Character string with the name of the phase variable. Defaults to the attributes in the scdf file.
<code>model</code>	Model used for calculating the slope parameter (see Huitema & McKean, 2000). Default is <code>model = "B&L-B"</code> . Possible values are: "B&L-B", "H-M", "Mohr#1", "Mohr#2", "JW", "JW2", and "Manly".
<code>trend</code>	A logical indicating if a trend parameters is included in the model.
<code>level</code>	A logical indicating if a level parameters is included in the model.
<code>slope</code>	A logical indicating if a slope parameters is included in the model.
<code>formula</code>	Defaults to the standard piecewise regression model. The parameter phase followed by the phase name (e.g., <code>phaseB</code>) indicates the level effect of the corresponding phase. The parameter 'inter' followed by the phase name (e.g., <code>interB</code>) addresses the slope effect based on the method provide in the model argument (e.g., "B&L-B"). The formula can be changed for example to include further variables into the regression model.
<code>update</code>	An easier way to change the regression formula (e.g., <code>. ~ . + newvariable</code>).
<code>na.action</code>	Defines how to deal with missing values
<code>...</code>	Further arguments passed to the <code>lm</code> function.

Value

<code>model</code>	Character string from function call (see Arguments above).
<code>full.model</code>	Full regression model list

Author(s)

Juergen Wilbert

See Also

Other regression functions: [corrected_tau\(\)](#), [hplm\(\)](#), [plm\(\)](#)

Examples

```
res <- mp1m(Leidig2018$`1a1`, dvar = c("academic_engagement", "disruptive_behavior"))
print(res)
## also report standardized coefficients:
print(res, std = TRUE)
```

nap	<i>Nonoverlap of all Pairs</i>
-----	--------------------------------

Description

The `nap` function calculates the nonoverlap of all pairs (NAP; Parker & Vannest, 2009). NAP summarizes the overlap between all pairs of phase A and phase B data points. If an increase of phase B scores is expected, a non-overlapping pair has a higher phase B data point. The NAP equals *numberofpairsshowingnooverlap/numberofpairs*. Because NAP can only take values between 50 and 100 percent, a rescaled and therefore more intuitive NAP (0-100%) is also displayed.

Usage

```
nap(data, dvar, pvar, decreasing = FALSE, phases = c(1, 2))
```

Arguments

<code>data</code>	A single-case data frame. See scdf to learn about this format.
<code>dvar</code>	Character string with the name of the dependent variable. Defaults to the attributes in the <code>scdf</code> file.
<code>pvar</code>	Character string with the name of the phase variable. Defaults to the attributes in the <code>scdf</code> file.
<code>decreasing</code>	If you expect data to be lower in the B phase, set <code>decreasing = TRUE</code> . Default is <code>decreasing = FALSE</code> .
<code>phases</code>	A vector of two characters or numbers indicating the two phases that should be compared. E.g., <code>phases = c("A", "C")</code> or <code>phases = c(2, 4)</code> for comparing the second to the fourth phase. Phases could be combined by providing a list with two elements. E.g., <code>phases = list(A = c(1, 3), B = c(2, 4))</code> will compare phases 1 and 3 (as A) against 2 and 4 (as B). Default is <code>phases = c("A", "B")</code> .

Value

<code>nap</code>	A data frame with NAP and additional values for each case.
<code>N</code>	Number of cases.

Author(s)

Juergen Wilbert

References

Parker, R. I., & Vannest, K. (2009). An improved effect size for single-case research: Nonoverlap of all pairs. *Behavior Therapy*, 40, 357-367.

See Also

Other overlap functions: [corrected_tau\(\)](#), [overlap\(\)](#), [pand\(\)](#), [pem\(\)](#), [pet\(\)](#), [pnd\(\)](#), [tau_u\(\)](#)

Examples

```
## Calculate NAP for a study with lower expected phase B scores (e.g. aggressive behavior)
gretchen <- scdf(c(A = 12,14,9,10, B = 10,6,4,5,3,4))
nap(gretchen, decreasing = TRUE)

## Request NAP for all cases from the Grosche2011 data
nap(Grosche2011)
```

outlier *Handling outliers in single-case data*

Description

Identifies and drops outliers within a single-case data frame (scdf).

Usage

```
outlier(data, dvar, pvar, mvar, criteria = c("MAD", "3.5"))

outlierSC(...)
```

Arguments

data	A single-case data frame. See scdf to learn about this format.
dvar	Character string with the name of the dependent variable. Defaults to the attributes in the scdf file.
pvar	Character string with the name of the phase variable. Defaults to the attributes in the scdf file.
mvar	Character string with the name of the measurement time variable. Defaults to the attributes in the scdf file.
criteria	Specifies the criteria for outlier identification. Set <code>criteria = c("SD", 2)</code> to define two standard deviations as limit. This is also the default setting. To use the 99% Confidence Interval use <code>criteria = c("CI", 0.99)</code> . Set <code>criteria = c("Cook", "4/n")</code> to define any data point with a Cook's Distance greater than $4/n$ as an outlier, based on the Piecewise Linear Regression Model.
...	Further arguments passed to the function.

Value

<code>data</code>	A single-case data frame with substituted outliers.
<code>dropped.n</code>	A list with the number of dropped data points for each single-case.
<code>dropped.mt</code>	A list with the measurement-times of dropped data points for each single-case (values are based on the <code>mt</code> variable of each single-case data frame).
<code>sd.matrix</code>	A list with a matrix for each case with values for the upper and lower boundaries based on the standard deviation.
<code>ci.matrix</code>	A list with a matrix for each single-case with values for the upper and lower boundaries based on the confidence interval.
<code>cook</code>	A list of Cook's Distances for each measurement of each single-case.
<code>criteria</code>	Criteria used for outlier analysis.
<code>N</code>	Number of single-cases.
<code>case.names</code>	Case identifier.

Author(s)

Juergen Wilbert

See Also

Other data manipulation functions: [as.data.frame.scdf\(\)](#), [fill_missing\(\)](#), [ranks\(\)](#), [shift\(\)](#), [smooth_cases\(\)](#), [standardize\(\)](#), [truncate_phase\(\)](#)

Examples

```
## Identify outliers using 1.5 standard deviations as criterion
susanne <- rSC(level = 1.0)
res_outlier <- outlier(susanne, criteria = c("SD", 1.5))
plot(susanne, marks = res_outlier)

## Identify outliers in the original data from Grosche (2011) using Cook's Distance
## greater than 4/n as criterion
res_outlier <- outlier(Grosche2011, criteria = c("Cook", "4/n"))
plot(Grosche2011, marks = res_outlier)
```

overlap

Overlap indices for single-case data

Description

The `overlap` function provides the most common overlap indices for single-case data and some additional statistics.

Usage

```
overlap(data, dvar, pvar, mvar, decreasing = FALSE, phases = c(1, 2))

overlapSC(...)
```

Arguments

data	A single-case data frame. See scdf to learn about this format.
dvar	Character string with the name of the dependent variable. Defaults to the attributes in the scdf file.
pvar	Character string with the name of the phase variable. Defaults to the attributes in the scdf file.
mvar	Character string with the name of the measurement time variable. Defaults to the attributes in the scdf file.
decreasing	If you expect data to be lower in the B phase, set decreasing = TRUE. Default is decreasing = FALSE.
phases	A vector of two characters or numbers indicating the two phases that should be compared. E.g., phases = c("A", "C") or phases = c(2, 4) for comparing the second to the fourth phase. Phases could be combined by providing a list with two elements. E.g., phases = list(A = c(1, 3), B = c(2, 4)) will compare phases 1 and 3 (as A) against 2 and 4 (as B). Default is phases = c("A", "B").
...	Further arguments passed to the function.

Details

See corresponding functions of PND, PEM, PET, NAP, PAND for calculation. Tau_U reports "A vs. B + Trend B - Trend A". Base_Tau is baseline corrected tau (correction applied when autocorrelation in phase A is significant). Diff_mean is the mean difference. Diff_trend is the difference in the regression estimation of the dependent variable on measurement-time ($x \sim mt$) for each phase. SMD is the mean difference divided by the standard deviation of phase A. Hedges_g is the mean difference divided by the pooled standard deviation $[\sqrt{((nA - 1) * sdA^2 + (nB - 1) * sdB^2) / (nA + nB - 2)}]$ with a hedges correction applied $[(1 - (3 / (4 * n - 9)))]$.

Value

overlap	A data frame consisting of the following indices for each single-case for all cases: PND, PEM, PET, NAP, PAND, Tau-U (A vs. B - Trend A), Diff_mean, Diff_trend, SMD, Hedges-g.
phases.A	Selection for A phase.
phases.B	Selection for B phase.
design	Phase design.

Author(s)

Juergen Wilbert

See Also

Other overlap functions: [corrected_tau\(\)](#), [nap\(\)](#), [pand\(\)](#), [pem\(\)](#), [pet\(\)](#), [pnd\(\)](#), [tau_u\(\)](#)

Examples

```
## Display overlap indices for one single-case
overlap(Huitema2000, decreasing = TRUE)

## Display overlap indices for six single-cases
overlap(GruenkeWilbert2014)

## Combining phases for analyzing designs with more than two phases
overlap(exampleA1B1A2B2, phases = list(c("A1", "A2"), c("B1", "B2")))
```

pand	<i>Percentage of all non-overlapping data</i>
------	---

Description

The `pand` function calculates the percentage of all non-overlapping data (PAND; Parker, Hagan-Burke, & Vannest, 2007), an index to quantify a level increase (or decrease) in performance after the onset of an intervention.

Usage

```
pand(data, dvar, pvar, decreasing = FALSE, correction = TRUE, phases = c(1, 2))
```

Arguments

<code>data</code>	A single-case data frame. See scdf to learn about this format.
<code>dvar</code>	Character string with the name of the dependent variable. Defaults to the attributes in the <code>scdf</code> file.
<code>pvar</code>	Character string with the name of the phase variable. Defaults to the attributes in the <code>scdf</code> file.
<code>decreasing</code>	If you expect data to be lower in the B phase, set <code>decreasing = TRUE</code> . Default is <code>decreasing = FALSE</code> .
<code>correction</code>	The default <code>correction = TRUE</code> makes <code>pand</code> use a frequency matrix, which is corrected for ties. A tie is counted as the half of a measurement in both phases. Set <code>correction = FALSE</code> to use the uncorrected matrix, which is not recommended.
<code>phases</code>	A vector of two characters or numbers indicating the two phases that should be compared. E.g., <code>phases = c("A", "C")</code> or <code>phases = c(2, 4)</code> for comparing the second to the fourth phase. Phases could be combined by providing a list with two elements. E.g., <code>phases = list(A = c(1, 3), B = c(2, 4))</code> will compare phases 1 and 3 (as A) against 2 and 4 (as B). Default is <code>phases = c("A", "B")</code> .

Details

The PAND indicates nonoverlap between phase A and B data (like PND), but uses all data and is therefore not based on one single (probably unrepresentative) datapoint. Furthermore, PAND allows the comparison of real and expected associations (Chi-square test) and estimation of the effect size Phi, which equals Pearsons r for dichotomous data. Thus, phi-Square is the amount of explained variance. The original procedure for computing the PAND (Parker, Hagan-Burke, & Vannest, 2007) does not account for ambivalent datapoints (ties). The newer NAP overcomes this problem and has better precision-power (Parker, Vannest, & Davis, 2014).

Value

PAND	Percentage of all non-overlapping data.
phi	Effect size Phi based on expected and observed values.
POD	Percentage of overlapping data points.
OD	Number of overlapping data points.
n	Number of data points.
N	Number of cases.
nA	Number of data points in phase A.
nB	Number of data points in phase B.
pA	Percentage of data points in phase A.
pB	Percentage of data points in phase B.
matrix	2x2 frequency matrix of phase A and B comparisons.
matrix.counts	2x2 counts matrix of phase A and B comparisons.
correlation	A list of the correlation values: statistic, parameter, p.value, estimate, null.value, alternative, method, data.name, correction.
correction	Logical argument from function call (see Arguments above).

Author(s)

Juergen Wilbert

References

- Parker, R. I., Hagan-Burke, S., & Vannest, K. (2007). Percentage of All Non-Overlapping Data (PAND): An Alternative to PND. *The Journal of Special Education, 40*, 194-204.
- Parker, R. I., & Vannest, K. (2009). An Improved Effect Size for Single-Case Research: Nonoverlap of All Pairs. *Behavior Therapy, 40*, 357-367.

See Also

Other overlap functions: [corrected_tau\(\)](#), [nap\(\)](#), [overlap\(\)](#), [pem\(\)](#), [pet\(\)](#), [pnd\(\)](#), [tau_u\(\)](#)

Examples

```
## Calculate the PAND for a MMBD over three cases
gunnar <- scdf(c(2,3,1,5,3,4,2,6,4,7), B.start = 5)
birgit <- scdf(c(3,3,2,4,7,4,2,1,4,7), B.start = 4)
bodo <- scdf(c(2,3,4,5,3,4,7,6,8,7), B.start = 6)
mbd <- c(gunnar, birgit, bodo)
pand(mbd)
pand(bodo)

## Calculate the PAND with an expected decrease of phase B scores
cubs <- scdf(c(20,22,24,17,21,13,10,9,20,9,18), B.start = 5)
pand(cubs, decreasing = TRUE)
```

pem	<i>Percent exceeding the median</i>
-----	-------------------------------------

Description

The pem function returns the percentage of phase B data exceeding the phase A median. Additionally, a chi square test against a 50/50 distribution is computed. Different measures of central tendency can be addressed for alternative analyses.

Usage

```
pem(
  data,
  dvar,
  pvar,
  decreasing = FALSE,
  binom.test = TRUE,
  chi.test = FALSE,
  FUN = median,
  phases = c(1, 2),
  ...
)
```

Arguments

data	A single-case data frame. See scdf to learn about this format.
dvar	Character string with the name of the dependent variable. Defaults to the attributes in the scdf file.
pvar	Character string with the name of the phase variable. Defaults to the attributes in the scdf file.
decreasing	If you expect data to be lower in the B phase, set decreasing = TRUE. Default is decreasing = FALSE.

<code>binom.test</code>	Computes a binomial test for a 50/50 distribution. Default is <code>binom.test = TRUE</code> .
<code>chi.test</code>	Computes a Chi-square test. The default setting <code>chi.test = FALSE</code> skips the Chi-square test.
<code>FUN</code>	Data points are compared with the phase A median. Use this argument to implement alternative measures of central tendency. Default is <code>FUN = median</code>
<code>phases</code>	A vector of two characters or numbers indicating the two phases that should be compared. E.g., <code>phases = c("A", "C")</code> or <code>phases = c(2, 4)</code> for comparing the second to the fourth phase. Phases could be combined by providing a list with two elements. E.g., <code>phases = list(A = c(1, 3), B = c(2, 4))</code> will compare phases 1 and 3 (as A) against 2 and 4 (as B). Default is <code>phases = c("A", "B")</code> .
<code>...</code>	Additional arguments for the <code>FUN</code> parameter (e.g. <code>FUN = mean, trim = 0.1</code> will use the 10 percent trimmed arithmetic mean instead of the median for comparisons). The function must take a vector of numeric values and the <code>na.rm</code> argument and return a numeric value.

Author(s)

Juergen Wilbert

See Also

Other overlap functions: [corrected_tau\(\)](#), [nap\(\)](#), [overlap\(\)](#), [pand\(\)](#), [pet\(\)](#), [pnd\(\)](#), [tau_u\(\)](#)

Examples

```
## Calculate the PEM including the Binomial and Chi-square tests for a single-case
dat <- rSC(5, level = 0.5)
pem(dat, chi.test = TRUE)
```

pet

Percent exceeding the trend

Description

The `pet` function provides the percentage of phase B data points exceeding the prediction based on the phase A trend. A binomial test against a 50/50 distribution is computed. Furthermore, the percentage of phase B data points exceeding the upper (or lower) 95 percent confidence interval of the predicted progress is computed.

Usage

```
pet(data, dvar, pvar, mvar, ci = 0.95, decreasing = FALSE, phases = c(1, 2))
```


Arguments

data	A single-case data frame. See scdf to learn about this format.
dvar	Character string with the name of the dependent variable. Defaults to the attributes in the scdf file.
pvar	Character string with the name of the phase variable. Defaults to the attributes in the scdf file.
mvar	Character string with the name of the measurement time variable. Defaults to the attributes in the scdf file.
ci	Width of the confidence interval. Default is $ci = 0.95$.
decreasing	If you expect data to be lower in the B phase, set <code>decreasing = TRUE</code> . Default is <code>decreasing = FALSE</code> .
phases	A vector of two characters or numbers indicating the two phases that should be compared. E.g., <code>phases = c("A", "C")</code> or <code>phases = c(2, 4)</code> for comparing the second to the fourth phase. Phases could be combined by providing a list with two elements. E.g., <code>phases = list(A = c(1, 3), B = c(2, 4))</code> will compare phases 1 and 3 (as A) against 2 and 4 (as B). Default is <code>phases = c("A", "B")</code> .

Value

PET	Percent exceeding the trend.
PET.ci	Percent exceeding the upper / lower 95%-CI boundary.
p	P value of Binomial Test.
ci.percent	Width of confidence interval in percent.
se.factors	Standard error.
N	Number of cases.
decreasing	Logical argument from function call (see Arguments above).
case.names	Assigned name of single-case.
phases	-

Author(s)

Juergen Wilbert

See Also

Other overlap functions: [corrected_tau\(\)](#), [nap\(\)](#), [overlap\(\)](#), [pand\(\)](#), [pem\(\)](#), [pnd\(\)](#), [tau_u\(\)](#)

Examples

```
## Calculate the PET and use a 99%-CI for the additional calculation
# create random example data
design <- design_rSC(n = 5, slope = 0.2)
dat <- rSC(design, seed = 23)
pet(dat, ci = .99)
```

plm

*Piecewise linear model / piecewise regression***Description**

The `plm` function computes a piecewise regression model (see Huitema & McKean, 2000).

Usage

```
plm(
  data,
  dvar,
  pvar,
  mvar,
  AR = 0,
  model = "B&L-B",
  family = "gaussian",
  trend = TRUE,
  level = TRUE,
  slope = TRUE,
  formula = NULL,
  update = NULL,
  na.action = na.omit,
  ...
)
```

Arguments

<code>data</code>	A single-case data frame. See scdf to learn about this format.
<code>dvar</code>	Character string with the name of the dependent variable. Defaults to the attributes in the <code>scdf</code> file.
<code>pvar</code>	Character string with the name of the phase variable. Defaults to the attributes in the <code>scdf</code> file.
<code>mvar</code>	Character string with the name of the measurement time variable. Defaults to the attributes in the <code>scdf</code> file.
<code>AR</code>	Maximal lag of autoregression. Modeled based on the Autoregressive-Moving Average (ARMA) function. When <code>AR</code> is set, the <code>family</code> argument must be set to <code>family = "gaussian"</code> .
<code>model</code>	Model used for calculating the slope parameter (see Huitema & McKean, 2000). Default is <code>model = "B&L-B"</code> . Possible values are: <code>"B&L-B"</code> , <code>"H-M"</code> , <code>"Mohr#1"</code> , <code>"Mohr#2"</code> , <code>"JW"</code> , <code>"JW2"</code> , and <code>"Manly"</code> .
<code>family</code>	Set the distribution family. Defaults to a gaussian distribution. See the <code>family</code> function for more details.
<code>trend</code>	A logical indicating if a trend parameters is included in the model.
<code>level</code>	A logical indicating if a level parameters is included in the model.

slope	A logical indicating if a slope parameters is included in the model.
formula	Defaults to the standard piecewise regression model. The parameter phase followed by the phase name (e.g., phaseB) indicates the level effect of the corresponding phase. The parameter 'inter' followed by the phase name (e.g., interB) addresses the slope effect based on the method provide in the model argument (e.g., "B&L-B"). The formula can be changed for example to include further variables into the regression model.
update	An easier way to change the regression formula (e.g., . ~ . + newvariable).
na.action	Defines how to deal with missing values
...	Further arguments passed to the glm function.

Value

formula	plm formula. Useful if you want to use the update or formula argument and you don't know the names of the parameters.
model	Character string from function call (see Arguments above).
F.test	F-test values of modelfit.
r.squares	Explained variance R squared for each model parameter.
ar	Autoregression lag from function call (see Arguments above).
family	Distribution family from function call (see Arguments above).
full.model	Full regression model list from the gls or glm function.

Author(s)

Juergen Wilbert

References

- Beretvas, S., & Chung, H. (2008). An evaluation of modified R2-change effect size indices for single-subject experimental designs. *Evidence-Based Communication Assessment and Intervention*, 2, 120-128.
- Huitema, B. E., & McKean, J. W. (2000). Design specification issues in time-series intervention models. *Educational and Psychological Measurement*, 60, 38-58.

See Also

Other regression functions: [corrected_tau\(\)](#), [hplm\(\)](#), [mplm\(\)](#)

Examples

```
## Compute a piecewise regression model for a random single-case
set.seed(123)
AB <- design_rSC(
  phase.design = list(A = 10, B = 20),
  level = list(A = 0, B = 1), slope = list(A = 0, B = 0.05),
  trend = list(0.05)
```

```

)
dat <- rSC(design = AB)
plm(dat, AR = 3)

## Another example with a more complex design
A1B1A2B2 <- design_rSC(
  phase.design = list(A1 = 15, B1 = 20, A2 = 15, B2 = 20),
  level = list(A1 = 0, B1 = 1, A2 = -1, B2 = 1),
  slope = list(A1 = 0, B1 = 0.0, A1 = 0, B2 = 0.0),
  trend = list(0.0))
dat <- rSC(design = A1B1A2B2, seed = 123)
plm(dat, model = "JW")

## no slope effects were found. Therefore you might want to drop slope estimation:
plm(dat, slope = FALSE, model = "JW")

## and now drop the trend estimation as well
plm(dat, slope = FALSE, trend = FALSE, model = "JW")

```

plot.scdf

Plot single-case data

Description

This function provides a plot of a single-case or multiple single-cases.

Usage

```

## S3 method for class 'scdf'
plot(...)

plotSC(
  data,
  dvar,
  pvar,
  mvar,
  ylim = NULL,
  xlim = NULL,
  xinc = 1,
  lines = NULL,
  marks = NULL,
  phase.names = NULL,
  xlab = NULL,
  ylab = NULL,
  main = "",
  case.names = NULL,
  style = getOption("scan.plot.style"),
  ...
)

```

Arguments

...	Further arguments passed to the plot command.
data	A single-case data frame. See scdf to learn about this format.
dvar	Character string with the name of the dependent variable. Defaults to the attributes in the scdf file.
pvar	Character string with the name of the phase variable. Defaults to the attributes in the scdf file.
mvar	Character string with the name of the measurement time variable. Defaults to the attributes in the scdf file.
ylim	Lower and upper limits of the y-axis (e.g., <code>ylim = c(0, 20)</code> sets the y-axis to a scale from 0 to 20). With multiple single-cases you can use <code>ylim = c(0, NA)</code> to scale the y-axis from 0 to the maximum of each case. <code>ylim</code> is not set by default, which makes scan set a proper scale based on the given data.
xlim	Lower and upper limits of the x-axis (e.g., <code>xlim = c(0, 20)</code> sets the x-axis to a scale from 0 to 20). With multiple single-cases you can use <code>xlim = c(0, NA)</code> to scale the x-axis from 0 to the maximum of each case. <code>xlim</code> is not set by default, which makes scan set a proper scale based on the given data.
xinc	An integer. Increment of the x-axis. 1 :each mt value will be printed, 2 : every other value, 3 : every third values etc.
lines	A list defining one or multiple lines or curves to be plotted. The argument is passed as a list (e.g., <code>list(type = "median")</code>). Some of the procedures can be refined with an additional argument (e.g., <code>lines = list(type = "mean", trim = 0.2)</code> adds a 20% trimmed mean line. For multiple lines, provide a list element for each line (e.g., <code>list(list(type = "median", col = "red"), list(type = "trend", col = "blue"))</code>). Possible lines are: <ul style="list-style-type: none"> • "median" Separate lines for phase A and B medians. • "mean" Separate lines for phase A and B means. By default it is 10%-trimmed. Other trims can be set, using a trim parameter (e.g., <code>lines = list(type = "mean", trim = 0.2)</code> draws a 20%-trimmed mean line). • "trend" Separate lines for phase A and B trends. • "trendA" OLS trend line for phase A, extrapolated throughout phase B. • "trendA_bisplit" Split middle (bi-split) trend line for phase A, extrapolated throughout phase B. • "trendA_trisplit" Tukey tri-split trend line for phase A, extrapolated throughout phase B. • "maxA/minA" Line at the level of the highest or lowest phase A score. • "medianA" Line at the phase A median score. • "meanA" Line at the phase A 10%-trimmed mean score. Apply a different trim, by using the additional argument (e.g., <code>lines = list(type = "meanA", trim = 0.2)</code>). • "plm" Regression lines for piecewise linear regression model. • "plm.ar" Regression lines for piecewise autoregression model. The lag is specified like this: <code>lines = list(type = "plm.ar", ar = 2)</code>. Default lag is set to 2.

- "movingMean" Draws a moving mean curve, with a specified lag: `lines = list(type = "movingMean", lag = 2)`. Default is a lag 1 curve.
- "movingMedian" Draws a moving median curve, with a specified lag: `lines = list(type = "movingMedian", lag = 3)`. Default is a lag 1 curve.
- "loreg" Draws a non-parametric local regression line. The proportion of data influencing each data point can be specified using `lines = list(type = "loreg", mf = 0.66)`. The default is 0.5.
- "lty" Use this argument to define the line type. Examples are: "solid", "dashed", "dotted".
- "lwd" Use this argument to define the line's thickness, e.g., `lwd = 4`.
- "col" Use this argument to define the line's color, e.g., `col = "red"`.

marks A list of parameters defining markings of certain data points.

- "positions" A vector or a list of vectors indicating measurement-times to be highlighted. In case of a vector, the marked measurement-times are the same for all plotted cases. In case of a list of vectors, marks are set differently for each case. The list must have the same length as there are cases in the data file.
- "col" Color of the marks.
- "cex" Size of the marks.

Use for example `marks = list(positions = c(1, 8, 15), col = "red", cex = 3)` to make the MTs one, eight and 18 appear big and red.

phase.names By default phases are labeled based on the levels of the phase variable. Use this argument to specify different labels: `phase.names = c("Baseline", "Intervention")`.

xlab The label of the x-axis. Default is `xlab = "Measurement time"`.

ylab The labels of the y-axis. Default is `ylab = "Score"`.

main Main title of the plot.

case.names Case names. If not provided, names are taken from the `scdf`. Set `case.names = ""` if you don't like to include case names.

style Either a character with the name of a pre-implemented style or a style object. See [style_plot](#) to learn about this format.

Value

Returns a plot of one or multiple single-cases.

Author(s)

Juergen Wilbert

See Also

[style_plot](#), [describeSC](#), [overlapSC](#)

Examples

```
## Request the default plot of the data from Borckhardt (2014)
plot(Borckardt2014)

## Plot the three cases from Grosche (2011) and visualize the phase A trend
plot(Grosche2011, style = "grid", lines = "trendA")

## Request the local regression line for Georg from that data set and customize the plot
plot(Grosche2011$Georg, style = "sienna", ylim = c(0,NA),
      xlab = "Training session", ylab = "Words per minute",
      phase.names = c("Baseline", "Intervention"), xinc = 5,
      lines = list(type = "loreg", f = 0.2, lty = "solid", col = "black", lwd = 3))

## Plot a random MBD over three cases and mark interesting MTs
dat <- rSC(design = design_rSC(3))
plot(dat, marks = list(positions = list(c(2,4,5),c(1,2,3),c(7,8,9))), col = "blue",
      cex = 1.4), style = c("grid", "annotate", "tiny"))
```

pnd

Percentage of non-overlapping data

Description

This function returns the percentage of non-overlapping data. Due to its error-proneness the PND should not be used, but [nap](#) or [pand](#) instead (see Parker & Vannest, 2009).

Usage

```
pnd(data, dvar, pvar, decreasing = FALSE, phases = c("A", "B"))
```

Arguments

data	A single-case data frame. See scdf to learn about this format.
dvar	Character string with the name of the dependent variable. Defaults to the attributes in the scdf file.
pvar	Character string with the name of the phase variable. Defaults to the attributes in the scdf file.
decreasing	If you expect data to be lower in the B phase, set decreasing = TRUE. Default is decreasing = FALSE.
phases	A vector of two characters or numbers indicating the two phases that should be compared. E.g., phases = c("A", "C") or phases = c(2, 4) for comparing the second to the fourth phase. Phases could be combined by providing a list with two elements. E.g., phases = list(A = c(1, 3), B = c(2, 4)) will compare phases 1 and 3 (as A) against 2 and 4 (as B). Default is phases = c("A", "B").

Value

PND Percentage of non-overlapping data.

Author(s)

Juergen Wilbert

See Also

Other overlap functions: [corrected_tau\(\)](#), [nap\(\)](#), [overlap\(\)](#), [pand\(\)](#), [pem\(\)](#), [pet\(\)](#), [tau_u\(\)](#)

Examples

```
## Calculate the PND for multiple single-case data
pnd(GruenkeWilbert2014)
```

power_test

Empirical power analysis for single-case data

Description

The `power_test` command conducts a Monte-Carlo study on the test-power and alpha-error of a set of single-cases. The distribution values of the Monte-Carlo sample are either specified by the user or estimated based on actual data.

Usage

```
power_test(
  design,
  stat = c("plm_level", "rand", "tauU"),
  n_sim = 100,
  alpha = 0.05
)

power_testSC(...)
```

Arguments

<code>design</code>	An object created by <code>design_rSC</code>
<code>stat</code>	Defines the tests the power analysis is based on. The default <code>stat = c("plm_level", "rand", "tauU")</code> computes a power analysis based on tau_u , randSC and plm analyses. Further possibilities are: "plm_slope", "plm_poisson_level", "plm_poisson_slope", "hplm_level", "hplm_slope", "base_tau".
<code>n_sim</code>	Number of sample studies created for the the Monte-Carlo study. Default is <code>n = 100</code>

alpha Alpha level used to calculate the proportion of significant tests. Default is alpha = 0.05.

... Further arguments passed to the function.

Author(s)

Juergen Wilbert

See Also

[plm](#), [randSC](#)

Examples

```
## Assume you want to conduct a single-case study with 15 MTs, using a highly reliable test,
## an expected level effect of  $d = 1.4$ , and randomized start points between MTs 5
## and 12 can you expect to identify the effect using plm or randomization test?
design <- design_rSC(
  n = 1, phase.design = list(A = 6, B = 9),
  rtt = 0.8, level = 1.4
)
res <- power_test(design, n_sim = 10)

## Would you achieve higher power by setting up a MBD with three cases?
design <- design_rSC(
  n = 3, phase.design = list(A = 6, B = 9),
  rtt = 0.8, level = 1.4
)
res <- power_test(design, n_sim = 10, stat = c("hplm_level", "rand"))
```

print.sc

Print method for scan objects

Description

Print method for scan objects

Usage

```
## S3 method for class 'sc_power'
print(x, digits = "auto", ...)

## S3 method for class 'sc_ac'
print(x, digits = "auto", ...)

## S3 method for class 'sc_cdc'
print(x, nice = TRUE, ...)
```

```
## S3 method for class 'sc_bctau'  
print(x, nice = TRUE, digits = "auto", ...)  
  
## S3 method for class 'sc_desc'  
print(x, digits = "auto", ...)  
  
## S3 method for class 'sc_design'  
print(x, ...)  
  
## S3 method for class 'sc_hplm'  
print(x, ...)  
  
## S3 method for class 'sc_overlap'  
print(x, digits = "auto", ...)  
  
## S3 method for class 'sc_mplm'  
print(x, digits = "auto", std = FALSE, ...)  
  
## S3 method for class 'sc_nap'  
print(x, digits = "auto", ...)  
  
## S3 method for class 'sc_outlier'  
print(x, digits = "auto", ...)  
  
## S3 method for class 'sc_pand'  
print(x, ...)  
  
## S3 method for class 'sc_pem'  
print(x, ...)  
  
## S3 method for class 'sc_pet'  
print(x, ...)  
  
## S3 method for class 'sc_pnd'  
print(x, ...)  
  
## S3 method for class 'sc_tauu'  
print(x, complete = FALSE, digits = "auto", ...)  
  
## S3 method for class 'sc_plm'  
print(x, ...)  
  
## S3 method for class 'sc_trend'  
print(x, digits = 3, ...)  
  
## S3 method for class 'sc_rand'  
print(x, ...)
```

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

Arguments

x	Object
digits	The minimum number of significant digits to be use. If set to "auto" (default), values are predefined.
...	Further parameters passed to the print function
nice	If set TRUE (default) output values are rounded and optimized for publication tables.
std	If TRUE, a table with standardized estimates is included.
complete	Print further parameters.

print.scdf	<i>Print an scdf</i>
------------	----------------------

Description

Print an scdf

Usage

```
## S3 method for class 'scdf'
print(
  x,
  cases = getOption("scan.print.cases"),
  rows = getOption("scan.print.rows"),
  cols = getOption("scan.print.cols"),
  long = getOption("scan.print.long"),
  digits = getOption("scan.print.digits"),
  ...
)
```

Arguments

x	An scdf object
cases	Number of cases to be printed. "fit" fits the number to the current screen width.
rows	Number of rows to be printed.
cols	Columns to be printed. "Main" only prints the dependent, measurement-time and phase variable.
long	Logical. If TRUE cases are printed in one by a time.
digits	Number of digits.
...	Further arguments passed to the print function.

Details

Print options for scdf objects could be set globally: `option(scan.print.cases = "all")`, `option(scan.print.rows = 10)`, `option(scan.print.cols = "main")`, `option(scan.print.long = TRUE)`, `option(scan.print.digits = 0)`, `option(scan.print.scdf.name = FALSE)`

 random

Single-case data generator

Description

The `rSC` function generates random single-case data frames for monte-carlo studies and demonstration purposes. `design_rSC` is used to set up a design matrix with all parameters needed for the `rSC` function.

Usage

```
rSC(design = NULL, round = NA, random.names = FALSE, seed = NULL, ...)
```

```
design_rSC(
  n = 1,
  phase.design = list(A = 5, B = 15),
  trend = list(0),
  level = list(0),
  slope = list(0),
  rtt = list(0.8),
  m = list(50),
  s = list(10),
  extreme.p = list(0),
  extreme.d = c(-4, -3),
  missing.p = list(0),
  distribution = "normal",
  prob = 0.5,
  MT = NULL,
  B.start = NULL
)
```

Arguments

<code>design</code>	A design matrix which is created by <code>design_rSC</code> and specifies all parameters.
<code>round</code>	Rounds the scores to the defined decimal. To round to the second decimal, set <code>round = 2</code> .
<code>random.names</code>	Is <code>FALSE</code> by default. If set <code>random.names = TRUE</code> cases are assigned random first names. If set "male" or "female" only male or female names are chosen. The names are drawn from the 2,000 most popular names for newborns in 2012 in the U.S. (1,000 male and 1,000 female names).
<code>seed</code>	A seed number for the random generator.

...	Paramteres that are directly passed from the rSC function to the design_rSC function for a more concise coding.
n	Number of cases to be created (Default is n = 1).
phase.design	A vector defining the length and label of each phase. E.g., phase.length = c(A1 = 10, B1 = 10, A2 = 10, B2 = 10).
trend	Defines the effect size <i>d</i> of a trend per MT added across the whole data-set. To assign different trends to several single-cases, use a vector of values (e.g. trend = c(.1, .3, .5)). If the number of cases exceeds the length of the vector, values are repeated. While using a binomial or poisson distribution, d.trend indicates an increase in points / counts per MT.
level	Defines the level increase (effect size <i>d</i>) at the beginning of phase B. To assign different level effects to several single-cases, use a vector of values (e.g. level = c(.2, .4, .6)). If the number of cases exceeds the length of the vector, values are repeated. While using a binomial or poisson distribution, level indicates an increase in points / counts with the onset of the B-phase.
slope	Defines the increase in scores - starting with phase B - expressed as effect size <i>d</i> per MT. slope = .1 generates an incremental increase of 0.1 standard deviations per MT for all phase B measurements. To assign different slope effects to several single-cases, use a vector of values (e.g. slope = c(.1, .2, .3)). If the number of cases exceeds the length of the vector, values are repeated. While using a binomial or poisson distribution, d.slope indicates an increase in points / counts per MT.
rtt	Reliability of the underlying simulated measurements. Set rtt = .8 by default. To assign different reliabilities to several single-cases, use a vector of values (e.g. rtt = c(.6, .7, .8)). If the number of cases exceeds the length of the vector, values are repeated. rtt has no effect when you're using binomial or poisson distributed scores.
m	Mean of the sample distribution the scores are drawn from. Default is m = 50. To assign different means to several single-cases, use a vector of values (e.g. m = c(50, 42, 56)). If the number of cases exceeds the length of the vector, values are repeated.
s	Standard deviation of the sample distribution the scores are drawn from. Set to s = 10 by default. To assign different variances to several single-cases, use a vector of values (e.g. s = c(5, 10, 15)). If the number of cases exceeds the length of the vector, values are repeated.
extreme.p	Probability of extreme values. extreme.p = .05 gives a five percent probability of an extreme value. A vector of values assigns different probabilities to multiple cases. If the number of cases exceeds the length of the vector, values are repeated.
extreme.d	Range for extreme values, expressed as effect size <i>d</i> . extreme.d = c(-7, -6) uses extreme values within a range of -7 and -6 standard deviations. In case of a binomial or poisson distribution, extreme.d indicates points / counts. Caution: the first value must be smaller than the second, otherwise the procedure will fail.
missing.p	Portion of missing values. missing.p = 0.1 creates 10% of all values as missing. A vector of values assigns different probabilities to multiple cases. If the number of cases exceeds the length of the vector, values are repeated.

distribution	Distribution of the scores. Default is <code>distribution = "normal"</code> . Possible values are "normal", "binomial", and "poisson". If set to "normal", the sample of scores will be normally distributed with the parameters <code>m</code> and <code>s</code> as mean and standard deviation of the sample, including a measurement error defined by <code>rtt</code> . If set to "binomial", data are drawn from a binomial distribution with the expectation value <code>m</code> . This setting is useful for generating criterial data like correct answers in a test. If set to "poisson", data are drawn from a poisson distribution, which is very common for count-data like behavioral observations. There's no measurement error is included. <code>m</code> defines the expectation value of the poisson distribution, <code>lambda</code> .
prob	If <code>distribution</code> (see below) is set "binomial", <code>prob</code> passes the probability of occurrence.
MT	Number of measurements (in each study). Default is <code>MT = 20</code> .
B.start	Phase B starting point. The default setting <code>B.start = 6</code> would assign the first five scores (of each case) to phase A, and all following scores to phase B. To assign different starting points for a set of multiple single-cases, use a vector of starting values (e.g. <code>B.start = c(6, 7, 8)</code>). If the number of cases exceeds the length of the vector, values will be repeated.

Value

A single-case data frame. See [scdf](#) to learn about this format.

Author(s)

Juergen Wibert

Examples

```
## Create random single-case data and inspect it
design <- design_rSC(
  n = 3, rtt = 0.75, slope = 0.1, extreme.p = 0.1,
  missing.p = 0.1
)
dat <- rSC(design, round = 1, random.names = TRUE, seed = 123)
describeSC(dat)
plotSC(dat)

## And now have a look at poisson-distributed data
design <- design_rSC(
  n = 3, B.start = c(6, 10, 14), MT = c(12, 20, 22), m = 10,
  distribution = "poisson", level = -5, missing.p = 0.1
)
dat <- rSC(design, seed = 1234)
pand(dat, decreasing = TRUE, correction = FALSE)
```

Description

The randSC function computes a randomization test for single or multiple baseline single-case data. The function is based on an algorithm from the SCRT package (Bulte & Onghena, 2009, 2012), but rewritten and extended for the use in AB designs.

Usage

```
rand_test(  
  data,  
  dvar,  
  pvar,  
  statistic = "Mean B-A",  
  number = 500,  
  complete = FALSE,  
  limit = 5,  
  startpoints = NA,  
  exclude.equal = FALSE,  
  graph = FALSE,  
  output = "c",  
  phases = c("A", "B"),  
  seed = NULL  
)  
  
randSC(...)  
  
rand.test(...)
```

Arguments

data	A single-case data frame. See scdf to learn about this format.
dvar	Character string with the name of the dependent variable. Defaults to the attributes in the scdf file.
pvar	Character string with the name of the phase variable. Defaults to the attributes in the scdf file.
statistic	Defines the statistic on which the comparison of phases A and B is based on. Default setting is <code>statistic = "Mean B-A"</code> . The following comparisons are possible: <ul style="list-style-type: none">• "Mean A-B": Uses the difference between the mean of phase A and the mean of phase B. This is appropriate if a decrease of scores was expected for phase B.

- "Mean B-A": Uses the difference between the mean of phase B and the mean of phase A. This is appropriate if an increase of scores was expected for phase B.
- "Mean |A-B|": Uses the absolute value of the difference between the means of phases A and B.
- "Median A-B": The same as "Mean A-B", but based on the median.
- "Median B-A": The same as "Mean B-A", but based on the median.

number	Sample size of the randomization distribution. The exactness of the p-value can not exceed $1/number$ (i.e., number = 100 results in p-values with an exactness of one percent). Default is number = 500. For faster processing use number = 100. For more precise p-values set number = 1000.
complete	If TRUE, the distribution is based on a complete permutation of all possible starting combinations. This setting overwrites the number Argument. The default setting is FALSE.
limit	Minimal number of data points per phase in the sample. The first number refers to the A-phase and the second to the B-phase (e.g., limit = c(5,3)). If only one number is given, this number is applied to both phases. Default is limit = 5.
startpoints	Alternative to the limit-parameter startpoints exactly defines the possible start points of phase B (e.g., startpoints = 4:9 restricts the phase B start points to measurements 4 to 9. startpoints overruns the limit-parameter.
exclude.equal	If set to exclude.equal = FALSE, which is the default, random distribution values equal to the observed distribution are counted as null-hypothesis conform. That is, they decrease the probability of rejecting the null-hypothesis (increase the p-value). exclude.equal should be set to TRUE if you analyse one single-case design (not a multiple baseline data set) to reach a sufficient power. But be aware, that it increases the chance of an alpha-error.
graph	If graph = TRUE, a histogram of the resulting distribution is plotted. It's FALSE by default.
output	If set to the default output = "C", detailed information is provided. Set output = "p", to only return the resulting p value.
phases	A vector of two characters or numbers indicating the two phases that should be compared. E.g., phases = c("A", "C") or phases = c(2, 4) for comparing the second and the fourth phase. Phases could be combined by providing a list with two elements. E.g., phases = list(A = c(1, 3), B = c(2, 4)) will compare phases 1 and 3 (as A) against 2 and 4 (as B). Default is phases = c("A", "B").
seed	A seed number for the random generator.
...	Further arguments passed to the function.

Value

statistic	Character string from function call (see Arguments above).
N	Number of single-cases.
n1	Number of data points in phase A.

n2	Number of data points in phase B.
limit	Numeric from function call (see Arguments above).
startpoints	A vector defining the start points passed from the function call (see Arguments above).
p.value	P-value of the randomization test for the given data.
number	Sample size of randomization distribution from function call (see Arguments above).
complete	Logical argument from function call (see Arguments above).
observed.statistic	Test statistic observed for the given single-case data. (see statistic in the Arguments above.)
Z	Z-value of observed test statistic.
p.z.single	Probability of z-value.
distribution	Test statistic distribution from randomized data sets.
possible.combinations	Number of possible combinations under the given restrictions.
auto.corrected.number	TRUE indicates that a corrected number of combinations was used. This happens, if the number of possible combinations (under the given restrictions) undercuts the requested number of combinations.

Author(s)

Juergen Wilbert

References

Bulte, I., & Onghena, P. (2009). Randomization tests for multiple-baseline designs: An extension of the SCRT-R package. *Behavior Research Methods*, 41, 477-485.

Bulte, I., & Onghena, P. (2012). *SCRT: Single-Case Randomization Tests*. Available from: <https://CRAN.R-project.org/package=SCRT>

Examples

```
## Compute a randomization test on the first case of the byHeart2011 data and include a graph
rand_test(byHeart2011[1], statistic = "Median B-A", graph = TRUE, seed = 123)
```

```
## Compute a randomization test on the Grosche2011 data using complete permutation
rand_test(Grosche2011, statistic = "Median B-A", complete = TRUE, limit = 4, seed = 123)
```

ranks	<i>Rank-transformation of single-case data files</i>
-------	--

Description

Rank-transformation of single-case data files

Usage

```
ranks(data, var, grand = TRUE, ...)
```

```
rankSC(...)
```

Arguments

data	A single-case data frame. See scdf to learn about this format.
var	A string or string vector with the names of the variables to be ranked.
grand	If TRUE, ranks will be calculated across all cases. If FALSE ranks are calculated within each case.
...	Additional paramters passed to the rank function.

Value

An scdf object where the values of the variable(s) are replaced with ranks.

Author(s)

Juergen Wilbert

See Also

Other data manipulation functions: [as.data.frame.scdf\(\)](#), [fill_missing\(\)](#), [outlier\(\)](#), [shift\(\)](#), [smooth_cases\(\)](#), [standardize\(\)](#), [truncate_phase\(\)](#)

Examples

```
Huber2014_rank <- ranks(Huber2014, var = "compliance")  
plot(Huber2014_rank, style = "grid2")
```

rci	<i>Reliable change index</i>
-----	------------------------------

Description

CAUTION! This function is still under development and not ready for use! The `rciSC` function computes three indices of reliable change (Wise, 2004) and corresponding descriptive statistics.

Usage

```
rci(data, dvar, pvar, rel, ci = 0.95, graph = FALSE, phases = c(1, 2))
rciSC(...)
rCi(...)
```

Arguments

<code>data</code>	A single-case data frame. See scdf to learn about this format.
<code>dvar</code>	Character string with the name of the dependent variable. Defaults to the attributes in the <code>scdf</code> file.
<code>pvar</code>	Character string with the name of the phase variable. Defaults to the attributes in the <code>scdf</code> file.
<code>rel</code>	Reliability of the measure, used to compute the standard error.
<code>ci</code>	Width of confidence interval as a decimal. Default is <code>ci = 0.95</code> applying a 95%-confidence interval.
<code>graph</code>	If set TRUE, a box plot of phase A and B scores is displayed. <code>graph = FALSE</code> by default.
<code>phases</code>	A vector of two characters or numbers indicating the two phases that should be compared. E.g., <code>phases = c("A", "C")</code> or <code>phases = c(2, 4)</code> for comparing the second to the fourth phase. Phases could be combined by providing a list with two elements. E.g., <code>phases = list(A = c(1, 3), B = c(2, 4))</code> will compare phases 1 and 3 (as A) against 2 and 4 (as B). Default is <code>phases = c("A", "B")</code> .
<code>...</code>	Further arguments passed to the function.

Author(s)

Juergen Wilbert

References

Christensen, L., & Mendoza, J. L. (1986). A method of assessing change in a single subject: An alteration of the RC index. *Behavior Therapy*, *17*, 305-308.

Hageman, W. J. J., & Arrindell, W. A. (1993). A further refinement of the reliable change (RC) index by improving the pre-post difference score: Introducing RCID. *Behaviour Research and Therapy*, 31, 693-700.

Jacobson, N. S., & Truax, P. (1991). Clinical Significance: A statistical approach to defining meaningful change in psychotherapy research. *Journal of Consulting and Clinical Psychology*, 59, 12-19.

Wise, E. A. (2004). Methods for analyzing psychotherapy outcomes: A review of clinical significance, reliable change, and recommendations for future directions. *Journal of Personality Assessment*, 82, 50 - 59.

Examples

```
## Report the RCIs of the first case from the byHeart data and include a graph
rci(byHeart2011[1], graph = TRUE, rel = 0.8)
```

readSC	<i>Read single-case data from files</i>
--------	---

Description

Use the readSC function to import single-case data from structured .csv or the readSC.excel function for importing excel files.

Usage

```
readSC(
  filename = NULL,
  data = NULL,
  sep = ",",
  dec = ".",
  sort.labels = FALSE,
  cvar = "case",
  pvar = "phase",
  dvar = "values",
  mvar = "mt",
  phase.names = NULL,
  type = "csv",
  ...
)

readSC.excel(...)
```

Arguments

filename	A character string defining the file to be imported (e.g. "SC_Anita.csv". If filename is left empty a dialog box for choosing will be opened.
data	A data frame. As an alternative to filename a dataframe could be directly provided.
sep	The field separator string. Values within rows have to be separated by this string. Default is sep = ", ".
dec	The string used for decimal points in the file. Must be a single character. Default is dec = "."
sort.labels	If set TRUE, the resulting list is sorted by label names (alphabetically increasing).
cvar	Sets the variable name of the "case" variable. Defaults to "case".
pvar	Sets the variable name of the "phase" variable. Defaults to "phase".
dvar	Sets the variable name of the "values" variable. Defaults to "values".
mvar	Sets the variable name of the "mt" variable. Defaults to "mt".
phase.names	A character vector with phase names. Defaults to the phase names provided in the phase variable.
type	Format of the file to be imported. Either "csv" or "excel" is possible.
...	Further arguments passed to the read.table command.

Value

Returns a single-case data frame. See [scdf](#) to learn about the format of these data frames.

Author(s)

Juergen Wilbert

See Also

[read.table](#), [writeSC](#), [scdf](#), [readRDS](#)

Examples

```
## Read SC-data from a file named "study1.csv" in your working directory
# study1 <- readSC("study1.csv")

## Read SC-data from a .csv-file with semicolon as field and comma as decimal separator
# study2 <- readSC("study2.csv", sep = ";", dec = ",")

## writeSc and readSC
filename <- file.path(tempdir(),"test.csv")
writeSC(exampleA1B1A2B2_zvt, filename)
dat <- readSC(filename, cvar = "case", pvar = "part", dvar = "zvt", mvar = "day")
res1 <- describeSC(exampleA1B1A2B2_zvt)$descriptives
```

```
res2 <- describeSC(dat)$descriptives
identical(res1, res2)
```

sample_names	<i>Samples random names</i>
--------------	-----------------------------

Description

Samples random names

Usage

```
sample_names(n = 1, type = "neutral", seed = NULL)
```

Arguments

n	Number of names
type	"neutral", "male", "female", or "mixed"
seed	A seed for the random number generator.

Value

A character vector with random names

Examples

```
sample_names(3)
```

scdf	<i>Single case data frame</i>
------	-------------------------------

Description

The class scdf stores single-case study data with one or more single-cases.

Usage

```

scdf(
  values,
  B.start,
  mt,
  phase,
  phase.design,
  name,
  dvar = "values",
  pvar = "phase",
  mvar = "mt",
  ...
)

```

Arguments

<code>values</code>	A vector containing measurement values of the dependent variable.
<code>B.start</code>	The first measurement of phase B (simple coding if design is strictly AB).
<code>mt</code>	A vector defining measurement times. Default is <code>mt = (1, 2, 3, ..., n)</code> .
<code>phase</code>	A vector defining phase assignments.
<code>phase.design</code>	A vector defining the length and label of each phase. E.g., <code>phase.length = c(A1 = 10, B1 = 10, A2 = 10, B2 = 10)</code> .
<code>name</code>	A name for the case.
<code>dvar</code>	Character string with the name of the dependent variable. Defaults to the attributes in the scdf file.
<code>pvar</code>	Character string with the name of the phase variable. Defaults to the attributes in the scdf file.
<code>mvar</code>	Character string with the name of the measurement time variable. Defaults to the attributes in the scdf file.
<code>...</code>	Additional variables. E.g., <code>teacher = c(0, 1, 0, 1, 0, 0, 1)</code> , <code>lesson = c(1, 3, 4, 5, 2, 3)</code> .

Details

The `scdf` class is a wrapper for a list containing a dataframe for each case.

If the dependent variable is a named vector then the names are extracted to create a phase design (e.g., `values = c(A = 2,3,5,4,3, B = 6,5,4,3)` will create an AB phase design with five and four measurements). An `scdf` contains several attributes: `dvar` The name of the dependent variable. `phase` The name of the phase variable. `mt` The name of the measurement time variable. `author` Information on the author of the data. `info` Further information on the data. E.g., a publication. `dvar`, `phase`, and `mt` are the defaults most of the scan function use. You can change the values of the attributes with the `scdf_attr` function (e.g., `scdf_attr(exampleAB_add, "dvar") <- "depression"` defines depression as the dependent variable. Please notice that all scan functions have arguments to define `dvar`, `phase`, and `mt` for a given analysis.

Value

Returns a single-case data frame `scdf` suitable for all functions of the `scan` package. Multiple data sets (e.g. from Multiple Baseline Designs) can be listed.

Author(s)

Juergen Wilbert

Examples

```
## Scores on a letter naming task were collected on eleven days in a row. The intervention
## started after the fifth measurement, so the first B phase measurement was 6 (B.start = 6).
klaas <- scdf(
  c(5, 7, 8, 5, 7, 12, 16, 18, 15, 14, 19),
  B.start = 6, name = "Klaas"
)
plot(klaas)

# Alternative coding 1:
klaas <- scdf(
  c(A = 5, 7, 8, 5, 7, B = 12, 16, 18, 15, 14, 19),
  name = "Klaas"
)

# Alternative coding 2:
klaas <- scdf(
  c(5, 7, 8, 5, 7, 12, 16, 18, 15, 14, 19),
  phase.design = c(A = 5, B = 6), name = "Klaas"
)

## Unfortunately in a similar SCDR there were no data collected on days 3 and 9. Use NA to
## pass them to the package.
emmi <- scdf(c(5, 7, NA, 5, 7, 12, 16, 18, NA, 14, 19),
  phase.design = c(A = 5, B = 6), name = "Emmi"
)
describeSC(emmi)

## In a MBD over three persons, data were again collected eleven days in a row. Intervention
## starting points differ between subjects as they were randomly assigned. The three SCDFs
## are then combined in a list for further conjoined analyses.
charlotte <- scdf(c(A = 5, 7, 10, 5, 12, B = 7, 10, 18, 15, 14, 19))
theresa <- scdf(c(A = 3, 4, 3, 5, B = 7, 4, 7, 9, 8, 10, 12))
antonia <- scdf(c(A = 9, 8, 8, 7, 5, 7, B = 6, 14, 15, 12, 16))
mbd <- c(charlotte, theresa, antonia)
names(mbd) <- c("Charlotte", "Theresa", "Antonia")
overlapSC(mbd)

## In a classroom-based intervention it was not possible to measure outcomes every day, but
## only on schooldays. The sequence of measurements is passed to the package by using a
## vector of measurement times.
frida <- scdf(
```



```
c(A = 3, 2, 4, 2, 2, 3, 5, 6, B = 8, 10, 8, 12, 14, 13, 12),
  mt = c(1, 2, 3, 4, 5, 8, 9, 10, 11, 12, 14, 15, 16, 17, 18)
)
summary(frida)
plot(frida)
describeSC(frida)

## example with two independent variables and four phases
jim <- scdf(
  zvt = c(47, 58, 76, 63, 71, 59, 64, 69, 72, 77, 76, 73),
  d2 = c(131, 134, 141, 141, 140, 140, 138, 140, 141, 140, 138, 140),
  phase.design = c(A1 = 3, B1 = 3, A2 = 3, B2 = 3), dvar = "zvt"
)
overlapSC(jim, phases = list(c("A1", "A2"), c("B1", "B2")))
```

scdf_attr

Set and get scdf attributes

Description

Set and get scdf attributes

Usage

```
scdf_attr(x, var)
```

```
scdf_attr(x, var) <- value
```

Arguments

x	Variable
var	Attribute
value	set value

Value

Attribute value

select_cases	<i>Select a subset of cases</i>
--------------	---------------------------------

Description

Select a subset of cases

Usage

```
select_cases(scdf, ...)
```

Arguments

scdf	A single-case data frame. See scdf to learn about this format.
...	Selection criteria. Either numeric or as characters.

Value

An scdf with a subset of cases

Examples

```
select_cases(exampleAB, "Johanna", "Karolina")
# or: select_cases(exampleAB, c("Johanna", "Karolina"))
select_cases(exampleAB, 1,2)
# or: select_cases(exampleAB, 1:2)
select_cases(exampleAB, "-Johanna")
# or: select_cases(exampleAB, c("-Johanna", "-Karolina"))
```

select_phases	<i>Select and combine phases for overlap analyses</i>
---------------	---

Description

Useful when working with

Usage

```
select_phases(data, A, B, ...)
```

Arguments

data	A single-case data frame. See scdf to learn about this format.
A	Selection of the A phase
B	Selection of the B phase
...	Further arguments passed to the function.

Value

An scdf with selected phases

Examples

```
exampleA1B1A2B2 %>%  
  select_phases(A = c(1, 3), B = c(2, 4)) %>%  
  overlap()
```

set_vars	<i>Set analysis variables in an scdf</i>
----------	--

Description

Set analysis variables in an scdf

Usage

```
set_vars(data, dvar, mvar, pvar)  
  
set_dvar(data, dvar)  
  
set_mvar(data, mvar)  
  
set_pvar(data, pvar)
```

Arguments

data	A single-case data frame. See scdf to learn about this format.
dvar	Character string. Name of the dependent variable.
mvar	Character string. Name of the measurement-time variable.
pvar	Character string. Name of the phase variable.

Examples

```
exampleAB_add %>%  
  set_dvar("depression") %>%  
  describe()
```

shift	<i>Shift values in a single-case data file</i>
-------	--

Description

Shifting the values might be helpful in cases where the measurement time is given as a time variable (see example below).

Usage

```
shift(data, value, var)
```

```
shiftSC(...)
```

Arguments

data	A single-case data frame. See scdf to learn about this format.
value	Number by which to shift the values
var	Character string with the name of the target variable. Defaults to the measurement time variable.
...	Further arguments passed to the function.

Value

A scdf with shifted data

See Also

Other data manipulation functions: [as.data.frame.scdf\(\)](#), [fill_missing\(\)](#), [outlier\(\)](#), [ranks\(\)](#), [smooth_cases\(\)](#), [standardize\(\)](#), [truncate_phase\(\)](#)

Examples

```
### Shift the measurement time for a better estimation of the intercept
ex <- shift(example_A24, value = -1996)
plm(ex)
```

`smooth_cases`*Smoothing single-case data*

Description

The `smooth_cases` function provides procedures to smooth single-case data (i.e., to eliminate noise). A moving average function (mean- or median-based) replaces each data point by the average of the surrounding data points step-by-step. With a local regression function, each data point is regressed by its surrounding data points.

Usage

```
smooth_cases(data, dvar, mvar, FUN = "movingMedian", intensity = NULL)
```

```
smoothSC(...)
```

Arguments

<code>data</code>	A single-case data frame. See scdf to learn about this format.
<code>dvar</code>	Character string with the name of the dependent variable. Defaults to the attributes in the <code>scdf</code> file.
<code>mvar</code>	Character string with the name of the measurement time variable. Defaults to the attributes in the <code>scdf</code> file.
<code>FUN</code>	Function determining the smoothed scores. Default <code>FUN = "movingMedian"</code> is a moving Median function. Further possible values are: <code>"movingMean"</code> and a non-parametric <code>"localRegression"</code> .
<code>intensity</code>	For <code>FUN = "movingMedian"</code> and <code>"movingMean"</code> it is the lag used for computing the average. Default is <code>intensity = 1</code> . In case of <code>FUN = "localRegression"</code> it is the proportion of surrounding data influencing each data point, which is <code>intensity = 0.2</code> by default.
<code>...</code>	Further arguments passed to the function.

Value

Returns a data frame (for each single-case) with smoothed data points. See [scdf](#) to learn about the format of these data frames.

Author(s)

Juergen Wilbert

See Also

Other data manipulation functions: [as.data.frame.scdf\(\)](#), [fill_missing\(\)](#), [outlier\(\)](#), [ranks\(\)](#), [shift\(\)](#), [standardize\(\)](#), [truncate_phase\(\)](#)

Examples

```
## Use the three different smoothing functions and compare the results
study <- c(
  "Original" = Huber2014$Berta,
  "Moving Median" = smooth_cases(Huber2014$Berta, FUN = "movingMedian"),
  "Moving Mean" = smooth_cases(Huber2014$Berta, FUN = "movingMean"),
  "Local Regression" = smooth_cases(Huber2014$Berta, FUN = "localRegression")
)
plot(study)
```

standardize

Standardize values of an scdf file

Description

This function scales the measured values of an scdf file. It allows for mean centering and standardization based on each single-case data set or a scaling across all cases included in an scdf.

Usage

```
standardize(
  data,
  var,
  center = TRUE,
  scale = FALSE,
  m = 0,
  sd = 1,
  grand = TRUE
)
```

Arguments

data	A single-case data frame. See scdf to learn about this format.
var	A character string or a vector of character strings with variable names that should be scaled.
center	If set TRUE, data are mean centered.
scale	If set TRUE, the standard deviation is set.
m	The target mean for centering.
sd	The target standard deviation for scaling
grand	If set TRUE, scaling is based on the mean and standard deviation of all values across all single-cases within the scdf.

Value

An scdf with the scaled values.

Author(s)

Juergen Wilbert

See Also

Other data manipulation functions: `as.data.frame.scdf()`, `fill_missing()`, `outlier()`, `ranks()`, `shift()`, `smooth_cases()`, `truncate_phase()`

Examples

```
## Standardize a multiple case scdf and compute an hplm
ex_sc <- standardize(exampleAB_50, var = "values", center = TRUE, scale = TRUE)
hplm(ex_sc)
```

style_plot

Create styles for single-case data plots

Description

The `style_plot` function is used to create graphical styles for a single-case plot

Usage

```
style_plot(style = "default", ...)

style_plotSC(...)

style.plotSC(...)
```

Arguments

`style` A character string or a vector of character strings with predefined styles.
`...` Further arguments passed to the plot command.

Details

`style_plot("")` will return a list of predefined styles. Predefined styles can be combined `style_plot(style = c("grid2", "tiny"))` where settings of a latter style overwrite settings of the former. Additional style parameters are set following the style argument and can be combined with those: `style_plot(style = "grid2", fill = "grey50", pch = 18)`.

Value

Returns a list to be provided for the style argument of the `plot.scdf` function.

- `fill` If set, the area under the line is filled with the given color (e.g., `fill = "tomato"`). Use the standard R command `colors()` to get a list of all possible colours. `fill` is empty by default.
- `annotations` A list of parameters defining annotations to each data point. This adds the score of each MT to your plot.
 - `"pos"` Position of the annotations: 1 = below, 2 = left, 3 = above, 4 = right.
 - `"col"` Color of the annotations.
 - `"cex"` Size of the annotations.
 - `"round"` Rounds the values to the specified decimal.
- `annotations = list(pos = 3, col = "brown", round = 1)` adds scores rounded to one decimal above the data point in brown color to the plot.
- `"names"` A list of parameters defining the depiction of phase names (e.g. `names = list(cex = 0.8, col = "red", side = 1)`: `cex` for size, `col` for color, and `side` for position). See `mtext` for more details.
- `"lwd"` Width of the plot line. Default is `lwd = 2`.
- `"pch"` Point type. Default is `pch = 17` (triangles). Other options are for example: 16 (filled circles) or "A" (uses the letter A).
- `"main"` Main title of the plot.
- `"mai"` Sets the margins of the plot.
- `"bty"` Shape of the frame surrounding the inner plot
- `"fill.bg"` Background color of the plot. If a vector is provided, these colors will be assigned to phases (each phase name becomes a color).
- `"grid"` Color of a grid.
- `"text.ABlag"` Text displayed between phases.
- `"cex.axis"` Size of the axis annotations
- `"las"` Orientation of the axis annotations
- `"col.lines"` Color of the lines
- `"col.dots"` Color of the dots
- `"col.seperator"` Color of the phase seperating lines
- `"col.bg"` Color of the outer plot
- `"col"` General color setting for the plot
- `"col.text"` Color of all labels of the plot.

Author(s)

Juergen Wilbert

See Also

[plot.scdf](#)

Examples

```

newstyle <- style_plot(style = "default")
newstyle$text.ABlag <- c("START", "END")
newstyle$col.dots <- ""
newstyle$annotations <- list(cex = 0.6, col = "grey10", offset = 0.4)
newstyle$names <- list(cex = 0.8, col = "blue", side = 1, adj = 1, line = -1, at = 31)
newstyle$fill.bg <- c("grey99", "grey95", "grey90")
plot(exampleABC, style = newstyle, main = "Example Plot")

```

subset.scdf

*Subset cases, rows, and variables***Description**

Subset cases, rows, and variables

Usage

```

## S3 method for class 'scdf'
subset(x, filter, select, cases, ...)

```

Arguments

x	An scdf object.
filter	Logical expression indicating rows to keep: missing values are taken as false.
select	Expression, indicating columns to select from an scdf.
cases	Expression, indicating cases to keep from an scdf.
...	[not implemented]

Value

An scdf.

Examples

```

subset(exampleAB, (values < 60 & phase == "A") | (values >= 60 & phase == "B"))
subset(exampleAB_add, select = c(-cigarrets, -depression))
subset(exampleAB, cases = c(Karolina, Johanna))
subset(exampleA1B1A2B2, phase %in% c("A1", "B2"), cases = Pawel:Moritz)

```

summary.scdf	<i>Summary functio for a scdf</i>
--------------	-----------------------------------

Description

Summary functio for a scdf

Usage

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

Arguments

object	scdf
...	not in use

tau_u	<i>Tau-U for single-case data</i>
-------	-----------------------------------

Description

This function calculates indices of the Tau-U family as proposed by Parker et al. (2011).

Usage

```
tau_u(
  data,
  dvar,
  pvar,
  tau_method = "b",
  method = "complete",
  phases = c(1, 2),
  meta_method = "random",
  continuity_correction = FALSE
)

tauUSC(...)
```

Arguments

data	A single-case data frame. See scdf to learn about this format.
dvar	Character string with the name of the dependent variable. Defaults to the attributes in the scdf file.
pvar	Character string with the name of the phase variable. Defaults to the attributes in the scdf file.
tau_method	Character with values "a" or "b" (default) indicating whether Kendall Tau A or Kendall Tau B is applied.
method	"complete" (default) or "parker". The latter calculates the number of possible pairs as described in Parker et al. (2011) which might lead to tau-U values greater than 1.
phases	A vector of two characters or numbers indicating the two phases that should be compared. E.g., phases = c("A", "C") or phases = c(2, 4) for comparing the second to the fourth phase. Phases could be combined by providing a list with two elements. E.g., phases = list(A = c(1, 3), B = c(2, 4)) will compare phases 1 and 3 (as A) against 2 and 4 (as B). Default is phases = c("A", "B").
meta_method	Character string. If set "random", a random-effect meta-analysis is calculated. If set "fixed", a fixed-effect meta-analysis is calculated.
continuity_correction	If TRUE, a continuity correction is applied for calculating p-values of correlations. This parameter is not yet implemented.
...	Further arguments passed to the function.

Details

Tau-U is an inconsistently operationalized construct. Parker et al. (2011) describe a method which may result in Tau-U larger than 1. A different implementation of the method (provided at <http://www.singlecaseresearch.org/calculators/tau-u>) uses tau-b (instead of tau-a as in the original formulation by Parker). Bossart et. al (2018) describe inconsistencies in the results from this implementation as well. Another problems lies in the calculation in overall Tau-U values from several single cases. The function presented here applies a metaanalyzes to gain the overall values. Each tau value is weighted by the inverse of the variance (ie. the tau standard error). Tau values are not converted to Pearson r values. The argument "meta_method" calculates a random-effect model ("random") or a fixed effect model ("fixed").

Value

table	A data frame containing statistics from the Tau-U family, including: Pairs, positive and negative comparisons, S, and Tau
matrix	The matrix of comparisons used for calculating the statistics.
tau_u	Tau-U value.

Author(s)

Juergen Wilbert

References

- Brossart, D. F., Laird, V. C., & Armstrong, T. W. (2018). Interpreting Kendall's Tau and Tau-U for single-case experimental designs. *Cogent Psychology*, *5*(1), 1–26. <https://doi.org/10.1080/23311908.2018.1518687>.
- Parker, R. I., Vannest, K. J., Davis, J. L., & Sauber, S. B. (2011). Combining Nonoverlap and Trend for Single-Case Research: Tau-U. *Behavior Therapy*, *42*, 284-299.

See Also

Other overlap functions: [corrected_tau\(\)](#), [nap\(\)](#), [overlap\(\)](#), [pand\(\)](#), [pem\(\)](#), [pet\(\)](#), [pnd\(\)](#)

Examples

```
tau_u(Grosche2011$Eva)

## Replicate tau-U calculation from Parker et al. (2011)
bob <- scdf(c(A = 2, 3, 5, 3, B = 4, 5, 5, 7, 6), name = "Bob")
res <- tau_u(bob, method = "parker", tau_method = "a")
print(res, complete = TRUE)

## Request tau-U for all single-cases from the Grosche2011 data set
tau_u(Grosche2011)
```

trend

Trend analysis for single-cases data

Description

The trend function provides an overview of linear trends in single-case data. By default, it gives you the intercept and slope of a linear and a squared regression of measurement-time on scores. Models are computed separately for each phase and across all phases. For a more advanced application, you can add regression models using the R specific formula class.

Usage

```
trend(data, dvar, pvar, mvar, offset = -1, model = NULL)
```

```
trendSC(...)
```

Arguments

- | | |
|------|--|
| data | A single-case data frame. See scdf to learn about this format. |
| dvar | Character string with the name of the dependent variable. Defaults to the attributes in the scdf file. |
| pvar | Character string with the name of the phase variable. Defaults to the attributes in the scdf file. |

mvar	Character string with the name of the measurement time variable. Defaults to the attributes in the scdf file.
offset	An offset for the first measurement-time of each phase (MT). If <code>offset = 0</code> , the phase measurement is handled as MT 1. Default is <code>offset = -1</code> , setting the first value of MT to 0.
model	A string or a list of (named) strings each depicting one regression model. This is a formula expression of the standard R class. The parameters of the model are <code>values</code> , <code>mt</code> and <code>phase</code> .
...	Further arguments passed to the function.

Value

trend	A matrix containing the results (Intercept, B and beta) of separate regression models for phase A, phase B, and the whole data.
offset	Numeric argument from function call (see Arguments section).

Author(s)

Juergen Wilbert

See Also

[describe](#), [autocorr](#), [plm](#)

Examples

```
## Compute the linear and squared regression for a random single-case
design <- design_rSC(slope = 0.5)
matthea <- rSC(design)
trend(matthea)

## Besides the linear and squared regression models compute two custom models:
## a) a cubic model, and b) the values predicted by the natural logarithm of the
## measurement time.
design <- design_rSC(slope = 0.3)
ben <- rSC(design)
trend(ben, offset = 0, model = c("Cubic" = values ~ I(mt^3), "Log Time" = values ~ log(mt)))
```

truncate_phase

Truncate single-case data

Description

This function truncates data points at the beginning and / or end of each phase in each case.

Usage

```
truncate_phase(
  data,
  dvar,
  pvar,
  truncate = list(A = c(0, 0), B = c(0, 0)),
  na = TRUE
)

truncateSC(...)
```

Arguments

<code>data</code>	A single-case data frame. See scdf to learn about this format.
<code>dvar</code>	Character string with the name of the dependent variable. Defaults to the attributes in the scdf file.
<code>pvar</code>	Character string with the name of the phase variable. Defaults to the attributes in the scdf file.
<code>truncate</code>	A list with a vector of two (beginning and end) values for each phase defining the number of data points to be deleted. For lists of single-case data frames, the truncation is adapted to the length of each phase for each single case.
<code>na</code>	If FALSE, the truncated measurement times are deleted. If TRUE, NAs are set for the dependent variable.
<code>...</code>	Further arguments passed to the function.

Value

A truncated data frame (for each single-case).

Author(s)

Juergen Wilbert

See Also

Other data manipulation functions: [as.data.frame.scdf\(\)](#), [fill_missing\(\)](#), [outlier\(\)](#), [ranks\(\)](#), [shift\(\)](#), [smooth_cases\(\)](#), [standardize\(\)](#)

Examples

```
# Truncate the first two data points of both phases and compare the two data sets
study <- c(
  "Original" = byHeart2011[1],
  "Selected" = truncate_phase(byHeart2011[1], truncate = list(A = c(2, 0), B = c(2, 0)))
)
plot(study)
```

`writeSC`*Export data into a .csv-file*

Description

This function restructures and exports single-case data into a .csv-file.

Usage

```
writeSC(data, filename = NULL, sep = ",", dec = ".", ...)
```

Arguments

<code>data</code>	A single-case data frame. See scdf to learn about this format.
<code>filename</code>	A character string defining the output file name (e.g. "SC_data.csv").
<code>sep</code>	The field separator string. Values within rows will be separated by this string. Default is <code>sep = ","</code> .
<code>dec</code>	The string used for decimal points. Must be a single character. Default is <code>dec = "."</code> .
<code>...</code>	Further arguments passed to <code>write.table</code> .

Author(s)

Juergen Wilbert

See Also

[write.table](#), [readSC](#), [saveRDS](#)

Examples

```
## Write single-case data to a .csv-file
filename <- tempfile()
jessica <- rSC(design_rSC(level = .5))
writeSC(jessica, filename)

## Write multiple cases to a .csv-file with semicolon as field and comma as decimal separator
writeSC(Grosche2011, filename, sep = ";", dec = ",")

## writeSC and readSC
writeSC(exampleA1B1A2B2_zvt, filename)
dat <- readSC(filename, cvar = "case", pvar = "part", dvar = "zvt", mvar = "day")
res1 <- describe(exampleA1B1A2B2_zvt)$descriptives
res2 <- describe(dat)$descriptives
all.equal(res1, res2)
```

<code>\$.scdf</code>	<i>Select a scdf</i>
----------------------	----------------------

Description

Select a scdf

Usage

```
## S3 method for class 'scdf'  
x$i  
  
## S3 method for class 'scdf'  
x[i]
```

Arguments

<code>x</code>	A scdf object
<code>i</code>	A case name from x

Value

A scdf

<code>%>%</code>	<i>Pipe</i>
---------------------	-------------

Description

Several functions in scan are designed to work with pipes %>%. This pipe function is directly imported from magrittr.

Details

Since R 4.1 a pipe operator `|>` is included in base R. This pipe operator can be used along with scan perfectly fine.

Index

- * **Autocorrelation**
 - autocorr, 6
- * **Serial correlation**
 - autocorr, 6
- * **data manipulation functions**
 - as.data.frame.scdf, 5
 - fill_missing, 19
 - outlier, 26
 - ranks, 50
 - shift, 60
 - smooth_cases, 61
 - standardize, 62
 - truncate_phase, 69
- * **data manipulation function**
 - scdf, 54
- * **datagen**
 - random, 44
- * **datasets**
 - Beretvas2008, 7
- * **manip**
 - as.data.frame.scdf, 5
 - fill_missing, 19
 - outlier, 26
 - readSC, 52
 - smooth_cases, 61
 - truncate_phase, 69
 - writeSC, 71
- * **overlap functions**
 - corrected_tau, 13
 - nap, 25
 - overlap, 27
 - pand, 29
 - pem, 31
 - pet, 32
 - pnd, 39
 - tau_u, 66
- * **package**
 - scan-package, 3
- * **regression functions**
 - corrected_tau, 13
 - hplm, 21
 - mplm, 23
 - plm, 34
 - .inheritParams, 3
 - [.scdf (\$.scdf), 72
 - \$.scdf, 72
 - %>%, 72
 - acf, 7
 - add_l2, 4
 - as.data.frame.scdf, 5, 20, 27, 50, 60, 61, 63, 70
 - as.scdf (scdf), 54
 - as_scdf, 6
 - autocorr, 6, 69
 - autocorrSC (autocorr), 6
 - Beretvas2008, 7
 - Borckardt2014 (Beretvas2008), 7
 - byHeart2011 (Beretvas2008), 7
 - c.scdf (combine), 11
 - cdc, 9
 - check_scdf, 11
 - combine, 11
 - convert, 12
 - corrected_tau, 13, 22, 24, 26, 29, 30, 32, 33, 35, 40, 68
 - corrected_tauSC (corrected_tau), 13
 - describe, 14, 69
 - describeSC, 38
 - describeSC (describe), 14
 - design_rSC (random), 44
 - estimate_design, 15
 - example_A24 (Beretvas2008), 7
 - exampleA1B1A2B2 (Beretvas2008), 7
 - exampleA1B1A2B2_zvt (Beretvas2008), 7
 - exampleAB (Beretvas2008), 7

- exampleAB_50 (Beretvas2008), 7
- exampleAB_add (Beretvas2008), 7
- exampleAB_decreasing (Beretvas2008), 7
- exampleAB_mpd (Beretvas2008), 7
- exampleAB_simple (Beretvas2008), 7
- exampleABAB (Beretvas2008), 7
- exampleABC (Beretvas2008), 7
- exampleABC_150 (Beretvas2008), 7
- exampleABC_50 (Beretvas2008), 7
- exampleABC_outlier (Beretvas2008), 7
- export, 17
- fill_missing, 5, 19, 27, 50, 60, 61, 63, 70
- fillmissingSC (fill_missing), 19
- Grosche2011 (Beretvas2008), 7
- Grosche2014 (Beretvas2008), 7
- GruenkeWilbert2014 (Beretvas2008), 7
- hplm, 14, 21, 24, 35
- Huber2014 (Beretvas2008), 7
- Huitema2000 (Beretvas2008), 7
- is.scdf, 23
- Leidig2018 (Beretvas2008), 7
- Leidig2018_l2 (Beretvas2008), 7
- Lenz2013 (Beretvas2008), 7
- longSCDF (as.data.frame.scdf), 5
- mblm, 13
- mplm, 14, 22, 23, 35
- mtext, 64
- nap, 14, 25, 29, 30, 32, 33, 39, 40, 68
- outlier, 5, 20, 26, 50, 60, 61, 63, 70
- outlierSC (outlier), 26
- overlap, 14, 15, 26, 27, 30, 32, 33, 40, 68
- overlapSC, 20, 38
- overlapSC (overlap), 27
- pand, 14, 26, 29, 29, 32, 33, 39, 40, 68
- Parker2011 (Beretvas2008), 7
- pem, 14, 26, 29, 30, 31, 33, 40, 68
- pet, 14, 26, 29, 30, 32, 32, 40, 68
- plm, 7, 14, 22, 24, 34, 40, 41, 69
- plot.scdf, 15, 36, 64
- plotSC (plot.scdf), 36
- pnd, 14, 26, 29, 30, 32, 33, 39, 68
- power_test, 40
- power_testSC (power_test), 40
- print.sc, 41
- print.sc_ac (print.sc), 41
- print.sc_bctau (print.sc), 41
- print.sc_cdc (print.sc), 41
- print.sc_desc (print.sc), 41
- print.sc_design (print.sc), 41
- print.sc_hplm (print.sc), 41
- print.sc_mplm (print.sc), 41
- print.sc_nap (print.sc), 41
- print.sc_outlier (print.sc), 41
- print.sc_overlap (print.sc), 41
- print.sc_pand (print.sc), 41
- print.sc_pem (print.sc), 41
- print.sc_pet (print.sc), 41
- print.sc_plm (print.sc), 41
- print.sc_pnd (print.sc), 41
- print.sc_power (print.sc), 41
- print.sc_rand (print.sc), 41
- print.sc_rci (print.sc), 41
- print.sc_tauu (print.sc), 41
- print.sc_trend (print.sc), 41
- print.scdf, 43
- rand.test (rand_test), 47
- rand_test, 47
- random, 44
- randSC, 20, 40, 41
- randSC (rand_test), 47
- rank, 50
- ranks, 5, 20, 27, 50, 60, 61, 63, 70
- rankSC (ranks), 50
- rCi (rci), 51
- rci, 51
- rciSC (rci), 51
- read.table, 53
- readRDS, 53
- readSC, 52, 71
- rSC (random), 44
- sample_names, 54
- saveRDS, 71
- scan (scan-package), 3
- scan-package, 3
- scdf, 4, 7, 9, 12–14, 16, 20, 21, 24–26, 28, 29, 31, 33, 34, 37, 39, 46, 47, 50, 51, 53, 54, 58–62, 67, 68, 70, 71
- scdf-class (scdf), 54

scdf_attr, [57](#)
scdf_attr<- (scdf_attr), [57](#)
select_cases, [58](#)
select_phases, [58](#)
set_dvar (set_vars), [59](#)
set_mvar (set_vars), [59](#)
set_pvar (set_vars), [59](#)
set_vars, [59](#)
shift, [5](#), [20](#), [27](#), [50](#), [60](#), [61](#), [63](#), [70](#)
shiftSC (shift), [60](#)
smooth_cases, [5](#), [20](#), [27](#), [50](#), [60](#), [61](#), [63](#), [70](#)
smoothSC (smooth_cases), [61](#)
SSDforR2017 (Beretvas2008), [7](#)
standardize, [5](#), [20](#), [27](#), [50](#), [60](#), [61](#), [62](#), [70](#)
style.plotSC (style_plot), [63](#)
style_plot, [38](#), [63](#)
style_plotSC (style_plot), [63](#)
subset.scdf, [65](#)
summary.scdf, [66](#)

tau_u, [14](#), [26](#), [29](#), [30](#), [32](#), [33](#), [40](#), [66](#)
tauUSC (tau_u), [66](#)
trend, [7](#), [68](#)
trendSC (trend), [68](#)
truncate_phase, [5](#), [20](#), [27](#), [50](#), [60](#), [61](#), [63](#), [69](#)
truncateSC (truncate_phase), [69](#)

Waddell2011 (Beretvas2008), [7](#)
write.table, [71](#)
writeSC, [53](#), [71](#)