

Package: SimplyAgree (via r-universe)

October 5, 2026

Type Package

Title Flexible and Robust Agreement and Reliability Analyses

Version 0.3.1

Maintainer Aaron Caldwell <arcaldwell149@gmail.com>

Description Reliability and agreement analyses often have limited software support. Therefore, this package was created to make agreement and reliability analyses easier for the average researcher. The functions within this package include simple tests of agreement, agreement analysis for nested and replicate data, and provide robust analyses of reliability. In addition, this package contains a set of functions to help when planning studies looking to assess measurement agreement.

URL <https://aaroncaldwell.us/SimplyAgree/>

BugReports <https://github.com/arcaldwell149/SimplyAgree/issues>

License GPL (>= 3)

Encoding UTF-8

LazyData true

RoxygenNote 7.3.3

Imports ggplot2, emmeans, lme4, boot, stats, dplyr, magrittr, tidyselect, tidyr, stringr, jmvcore, quantreg, patchwork, insight, nlme, purrr, lifecycle, scales

Suggests knitr, rmarkdown, testthat, readr, covr, mgcv, ggeffects, deming, pbkrtest

VignetteBuilder knitr

Depends R (>= 3.6)

Roxygen list(markdown = TRUE)

Config/pak/sysreqs cmake make libicu-dev

Repository <https://arcaldwell149.r-universe.dev>

Date/Publication 2026-10-05 16:43:39 UTC

RemoteUrl <https://github.com/arcaldwell149/simplyagree>

RemoteRef HEAD

RemoteSha d1a3c95843e83a421ea09fbde1b7910d38234517

Contents

agree_assurance	3
agree_coef	6
agree_expected_half	7
agree_nest	10
agree_np	12
agree_reps	13
agree_test	15
agreement_limit	16
blandPowerCurve	19
ccc_test	20
dem_reg	22
deming_power	25
deming_sample_size	27
jmvagree	29
jmvagreemulti	30
jmvdeming	31
jmvreli	33
joint_test	33
loa-methods	35
loa_lme	36
loa_mermod-methods	37
loa_mixed	38
loa_mixed_bs-methods	40
pb_reg	41
power_agreement_exact	45
powerCurve-methods	47
reli_stats	48
reps	51
simple_agree-methods	52
simple_eiv-methods	53
simple_reli-methods	55
temps	56
tolerance_delta-methods	57
tolerance_limit	58

Index

65

 agree_assurance

Assurance Probability for Limits of Agreement

Description

[Maturing]

Usage

```
agree_assurance(
  conf.level = 0.95,
  assurance = 0.9,
  omega = NULL,
  pstar = 0.95,
  sigma = 1,
  n = NULL
)
```

Arguments

conf.level	confidence level for the range of agreement (1 - alpha). The confidence level of the confidence interval of the range of agreement (tolerance interval). Default is 0.95.
assurance	target lower bound of the assurance probability (1 - gamma). The assurance probability is the probability that the study half-width will be less than omega. Common values are 0.80, 0.90, or 0.95.
omega	upper bound of assurance half-width. The sample size guarantees (assures) that 100(1 - gamma)% of interval half-widths will be less than this value. Can be specified in standard deviation units.
pstar	central proportion of the data distribution covered (P*). It is the proportion of observations that fall between the limits. For example, a value of 0.95 indicates that 95% of the variable's values fall between the limits. Must be between 0 and 1. Common values are 0.90 or 0.95.
sigma	population standard deviation of the paired differences. If the true value is unknown, omega can be specified in standard deviation units by setting sigma = 1.
n	sample size (optional). If provided, the function will solve for a different parameter rather than sample size.

Details

Calculate the sample size necessary for a confidence interval of the Bland-Altman range of agreement when the underlying data distribution is normal. This function uses the assurance probability criterion to determine the optimum sample size, based on the exact confidence interval method of Jan and Shieh (2018), which has been shown to be superior to approximate methods.

Overview:

This function implements the exact method for determining sample size based on assurance probability for Bland-Altman limits of agreement, as described in Jan and Shieh (2018). The assurance probability criterion determines an N that guarantees with specified probability ($1 - \gamma$) that the confidence interval half-width will be no more than a boundary value ω .

Technical Details:

Suppose a study involves paired differences ($X - Y$) whose distribution is approximately $N(\mu, \sigma^2)$. The range of agreement is defined as a confidence interval of the central portion of these differences, specifically the area between the $100(1-p)$ th and $100p$ -th percentiles, where $p^* = 2p - 1$.

The exact two-sided, $100(1 - \alpha)\%$ confidence interval for the range of agreement is defined as:

$$\Pr(\theta_{(1-p)} < \hat{\theta}_{(1-p)} \text{ and } \hat{\theta}_p < \theta_p) = 1 - \alpha$$

The equal-tailed tolerance interval recommended by Jan and Shieh (2018) is:

$$(\bar{X} - d, \bar{X} + d)$$

where $d = g * S$, g is the Odeh-Owen tolerance factor (tabulated as g'' in Odeh and Owen (1980)), and S is the sample standard deviation.

Sample Size Determination:

The sample size N is selected to satisfy: $\Pr(H \leq \omega) \geq 1 - \gamma$

This leads to the expression:

$$\psi(\eta) \geq 1 - \gamma$$

where $\psi()$ is the CDF of a chi-square distribution with $N-1$ degrees of freedom and $\eta = (N-1) * (\omega / (g * \sigma))^2$.

The method uses equal-tailed tolerance intervals based on the noncentral t-distribution to construct exact confidence intervals for the range of agreement. The tolerance factor g is calculated such that the interval maintains the specified confidence level under normality.

Jan and Shieh (2018) demonstrated through extensive simulations that this exact method should be adopted rather than the classical Bland-Altman approximate method.

Interpreting Results:

Each subject produces two measurements (one for each method being compared). The sample size n returned is the number of subject pairs needed. The actual assurance probability may differ slightly from the target due to the discrete nature of sample size.

For dropout considerations, inflate the sample size using: $N' = N / (1 - \text{dropout_rate})$, always rounding up.

Value

An object of class "power.htest" containing the following components:

- `n`: The required sample size (number of subject pairs)
- `conf.level`: The confidence level ($1 - \alpha$)
- `assurance`: The target assurance probability ($1 - \gamma$)
- `actual.assurance`: The actual assurance probability achieved (may differ slightly from target due to discrete nature of n)

- omega: The upper bound of assurance half-width
- pstar: The central proportion covered (P^*)
- sigma: The population standard deviation
- g.factor: The Odeh-Owen factor (g'') used to construct the tolerance interval, tabulated in Odeh and Owen (1980)
- method: Description of the method used
- note: Additional notes about the analysis

Assumptions

- The paired differences are normally distributed
- The variance is constant across the range of measurement
- Pairs are independent

References

Jan, S.L. and Shieh, G. (2018). The Bland-Altman range of agreement: Exact interval procedure and sample size determination. *Computers in Biology and Medicine*, **100**, 247-252. doi:10.1016/j.combiomed.2018.06.020

Odeh, R.E. and Owen, D.B. (1980). *Tables for Normal Tolerance Limits, Sampling Plans, and Screening*. Marcel Dekker, Inc., New York.

See Also

[agree_expected_half\(\)](#) for sample size determination using expected half-width criterion, [power_agreement_exact\(\)](#) for power analysis of agreement tests.

Examples

```
# Example: Planning a method comparison study
# Researchers want 95% confidence, 90% assurance that half-width
# will be within 2.5 SD units, covering central 95% of differences
agree_assurance(
  conf.level = 0.95,
  assurance = 0.90,
  omega = 2.5,
  pstar = 0.95,
  sigma = 1
)
```

agree_coef

*Agreement Coefficients***Description****[Maturing]**

agree_coef produces inter-rater reliability or "agreement coefficients" as described by Gwet.

Usage

```
agree_coef(
  wide = TRUE,
  col.names = NULL,
  measure,
  item,
  id,
  data,
  weighted = FALSE,
  conf.level = 0.95
)
```

Arguments

wide	Logical value (TRUE or FALSE) indicating if data is in a "wide" format. Default is TRUE.
col.names	If wide is equal to TRUE then col.names is a list of the column names containing the measurements for reliability analysis.
measure	Name of column containing the measurement of interest.
item	Name of column containing the items. If this is an inter-rater reliability study then this would indicate the rater (e.g., rater1, rater2, rater3, etc).
id	Column with subject identifier.
data	Data frame with all data.
weighted	Logical value (TRUE or FALSE) indicating whether to weight the responses. If TRUE (default is FALSE) then quadratic weights are utilized. This option should be set to TRUE for ordinal or continuous responses.
conf.level	the confidence level required. Default is 95%.

Value

Returns single data frame of inter-rater reliability coefficients.

References

Gwet, K.L. (2014, ISBN:978-0970806284). "Handbook of Inter-Rater Reliability", 4th Edition. Advanced Analytics, LLC. Gwet, K. L. (2008). "Computing inter-rater reliability and its variance in the presence of high agreement," British Journal of Mathematical and Statistical Psychology, 61, 29-48.

Examples

```
data('reps')
agree_coef(data = reps, wide = TRUE, col.names = c("x", "y"), weighted = TRUE)
```

agree_expected_half	<i>Sample Size for Limits of Agreement Using Expected Half-Width</i>
---------------------	--

Description**[Maturing]****Usage**

```
agree_expected_half(
  conf.level = 0.95,
  delta = NULL,
  pstar = 0.95,
  sigma = 1,
  n = NULL
)
```

Arguments

conf.level	confidence level for the range of agreement ($1 - \alpha$). The confidence level of the confidence interval of the range of agreement (tolerance interval). Default is 0.95.
delta	target upper bound of expected half-width (delta). The sample size guarantees that the expected half-width of the confidence interval will be no more than this value. Can be specified in standard deviation units.
pstar	central proportion of the data distribution covered (P^*). It is the proportion of observations that fall between the limits. For example, a value of 0.95 indicates that 95% of the variable's values fall between the limits. Must be between 0 and 1. Common values are 0.90 or 0.95.
sigma	population standard deviation of the paired differences. If the true value is unknown, delta can be specified in standard deviation units by setting $\sigma = 1$.
n	sample size (optional). If provided, the function will solve for a different parameter rather than sample size.

Details

Calculate the sample size necessary for a confidence interval of the Bland-Altman range of agreement when the underlying data distribution is normal. This function uses the expected half-width criterion to determine the optimum sample size, based on the exact confidence interval method of Jan and Shieh (2018), which has been shown to be superior to approximate methods.

Overview:

This function implements the exact method for determining sample size based on expected half-width for Bland-Altman limits of agreement, as described in Jan and Shieh (2018). The expected half-width criterion determines an N that guarantees that the expected half-width of the confidence interval is less than a boundary value δ .

Technical Details:

Suppose a study involves paired differences $(X - Y)$ whose distribution is approximately $N(\mu, \sigma^2)$. The range of agreement is defined as a confidence interval of the central portion of these differences, specifically the area between the $100(1-p)$ th and $100p$ -th percentiles, where $p^* = 2p - 1$.

The exact two-sided, $100(1 - \alpha)\%$ confidence interval for the range of agreement is defined as:

$$\Pr(\theta_{(1-p)} < \theta_{\hat{p}} \text{ and } \theta_{\hat{p}} < \theta_{(p)}) = 1 - \alpha$$

The equal-tailed tolerance interval recommended by Jan and Shieh (2018) is:

$$(\bar{X} - d, \bar{X} + d)$$

where $d = g * S$, g is the Odeh-Owen tolerance factor (tabulated as g'' in Odeh and Owen (1980) and Hahn and Meeker (1991)), and S is the sample standard deviation.

Sample Size Determination:

The sample size N is selected to satisfy: $E(H) \leq \delta$

where H is the half-width of the confidence interval. This leads to the expression:

$$g(P^*, 1-\alpha, N-1) / c \leq \delta / \sigma$$

where c is a bias correction factor:

$$c = (\Gamma((N-1)/2) * \sqrt{(N-1)/2}) / \Gamma(N/2)$$

The expected half-width is $E(H) = (g/c) * \sigma$. This accounts for the fact that the sample standard deviation S is a biased estimator of σ , requiring correction when computing expected values.

The method uses equal-tailed tolerance intervals based on the noncentral t -distribution to construct exact confidence intervals for the range of agreement. The tolerance factor g is calculated such that the interval maintains the specified confidence level under normality.

Comparison with Assurance Probability Method:

This function uses the expected half-width criterion, which ensures that $E(H) \leq \delta$. An alternative approach is the assurance probability criterion (see [agree_assurance\(\)](#)), which ensures that $\Pr(H \leq \omega) \geq 1 - \gamma$.

The expected half-width criterion:

- Controls the average half-width across repeated sampling
- Generally requires smaller sample sizes than assurance probability
- Is appropriate when the average performance is of primary interest

The assurance probability criterion:

- Provides a probability guarantee about the half-width
- Generally requires larger sample sizes
- Is appropriate when a stronger guarantee is needed for planning purposes

Interpreting Results:

Each subject produces two measurements (one for each method being compared). The sample size n returned is the number of subject pairs needed. The actual expected half-width may differ slightly from the target due to the discrete nature of sample size.

For dropout considerations, inflate the sample size using: $N' = N / (1 - \text{dropout_rate})$, always rounding up.

Value

An object of class "power.htest" containing the following components:

- `n`: The required sample size (number of subject pairs)
- `conf.level`: The confidence level ($1 - \alpha$)
- `delta.target`: The target upper bound of expected half-width
- `delta.actual`: The actual upper bound of expected half-width achieved (may differ slightly from target due to discrete nature of n)
- `pstar`: The central proportion covered (P^*)
- `sigma`: The population standard deviation
- `g.factor`: The Odeh-Owen factor (g'') used to construct the tolerance interval, tabulated in Odeh and Owen (1980)
- `c.factor`: The bias correction factor used in the expected half-width calculation
- `method`: Description of the method used
- `note`: Additional notes about the analysis

Assumptions

- The paired differences are normally distributed
- The variance is constant across the range of measurement
- Pairs are independent

References

- Jan, S.L. and Shieh, G. (2018). The Bland-Altman range of agreement: Exact interval procedure and sample size determination. *Computers in Biology and Medicine*, **100**, 247-252. doi:10.1016/j.combiomed.2018.06.020
- Bland, J.M. and Altman, D.G. (1986). Statistical methods for assessing agreement between two methods of clinical measurement. *The Lancet*, **327**(8476), 307-310.
- Hahn, G.J. and Meeker, W.Q. (1991). *Statistical Intervals: A Guide for Practitioners*. John Wiley & Sons, New York.
- Odeh, R.E. and Owen, D.B. (1980). *Tables for Normal Tolerance Limits, Sampling Plans, and Screening*. Marcel Dekker, Inc., New York.

See Also

`agree_assurance()` for sample size determination using assurance probability criterion, `power_agreement_exact()` for power analysis of agreement tests.

Examples

```
# Example 1: Reproduce Jan and Shieh (2018), page 251
# Expected half-width criterion with P* = 0.95
agree_expected_half(
  conf.level = 0.95,
  delta = 2.25 * 19.61,
  pstar = 0.95,
  sigma = 19.61
)
# Expected result: n = 155

# Example 2: Planning a method comparison study
# Researchers want 95% confidence with expected half-width
# no more than 2.4 SD units, covering central 90% of differences
agree_expected_half(
  conf.level = 0.95,
  delta = 2.4,
  pstar = 0.90,
  sigma = 1
)
)
```

agree_nest

Tests for Absolute Agreement with Nested Data

Description

[Superseded]

Development on `agree_nest()` is complete, and for new code we recommend switching to `agreement_limit()`, which is easier to use, has more features, and still under active development.

`agree_nest` produces an absolute agreement analysis for data where there is multiple observations per subject but the mean varies within subjects as described by Zou (2013). Output mirrors that of `agree_test` but CCC is calculated via U-statistics.

Usage

```
agree_nest(
  x,
  y,
  id,
  data,
  delta,
  agree.level = 0.95,
  conf.level = 0.95,
  TOST = TRUE,
  prop_bias = FALSE,
  ccc = TRUE
)
```

Arguments

x	Name of column with first measurement
y	Name of other column with the other measurement to compare to the first.
id	Column with subject identifier
data	Data frame with all data
delta	The threshold below which methods agree/can be considered equivalent, can be in any units. Equivalence Bound for Agreement.
agree.level	the agreement level required. Default is 95%. The proportion of data that should lie between the thresholds, for 95% limits of agreement this should be 0.95.
conf.level	the confidence level required. Default is 95%.
TOST	Logical indicator (TRUE/FALSE) of whether to use two one-tailed tests for the limits of agreement. Default is TRUE.
prop_bias	Logical indicator (TRUE/FALSE) of whether proportional bias should be considered for the limits of agreement calculations.
ccc	Calculate concordance correlation coefficient.

Value

Returns single `simple_agree` class object with the results of the agreement analysis.

- `loa`: A data frame of the limits of agreement including the average difference between the two sets of measurements, the standard deviation of the difference between the two sets of measurements and the lower and upper confidence limits of the difference between the two sets of measurements.
- `h0_test`: Decision from hypothesis test.
- `ccc.xy`: Lin's concordance correlation coefficient and confidence intervals using U-statistics. Warning: if underlying value varies this estimate will be inaccurate.
- `call`: the matched call.
- `var_comp`: Table of Variance Components.
- `class`: The type of `simple_agree` analysis.

References

- Zou, G. Y. (2013). Confidence interval estimation for the Bland–Altman limits of agreement with multiple observations per individual. *Statistical methods in medical research*, 22(6), 630-642.
- King, TS and Chinchilli, VM. (2001). A generalized concordance correlation coefficient for continuous and categorical data. *Statistics in Medicine*, 20, 2131:2147.
- King, TS; Chinchilli, VM; Carrasco, JL. (2007). A repeated measures concordance correlation coefficient. *Statistics in Medicine*, 26, 3095:3113.
- Carrasco, JL; Phillips, BR; Puig-Martinez, J; King, TS; Chinchilli, VM. (2013). Estimation of the concordance correlation coefficient for repeated measures using SAS and R. *Computer Methods and Programs in Biomedicine*, 109, 293-304.

Examples

```
data('reps')
agree_nest(x = "x", y = "y", id = "id", data = reps, delta = 2)
```

agree_np

Nonparametric Test for Limits of Agreement

Description

[Stable]

agree_np A non-parametric approach to limits of agreement. The hypothesis test is based on binomial proportions within the maximal allowable differences, and the limits are calculated with quantile regression.

Usage

```
agree_np(
  x,
  y,
  id = NULL,
  data,
  delta = NULL,
  prop_bias = FALSE,
  TOST = TRUE,
  agree.level = 0.95,
  conf.level = 0.95
)
```

Arguments

x	Name of column with first measurement.
y	Name of other column with the other measurement to compare to the first.
id	Column with subject identifier with samples are taken in replicates.
data	Data frame with all data.
delta	The threshold below which methods agree/can be considered equivalent and this argument is required. Equivalence Bound for Agreement or Maximal Allowable Difference.
prop_bias	Logical indicator (TRUE/FALSE) of whether proportional bias should be considered for the limits of agreement calculations.
TOST	Logical indicator (TRUE/FALSE) of whether to use two one-tailed tests for the limits of agreement. Default is TRUE.
agree.level	the agreement level required. Default is 95%. The proportion of data that should lie between the thresholds, for 95% limits of agreement this should be 0.95.
conf.level	the confidence level required. Default is 95%.

Value

Returns `simple_agree` object with the results of the agreement analysis.

- `loa`: A data frame of the limits of agreement.
- `agree`: A data frame of the binomial proportion of results in agreement.
- `h0_test`: Decision from hypothesis test.
- `qr_mod`: The quantile regression model.
- `call`: The matched call

References

Bland, J. M., & Altman, D. G. (1999). Measuring agreement in method comparison studies. In *Statistical Methods in Medical Research* (Vol. 8, Issue 2, pp. 135–160). SAGE Publications. [doi:10.1177/096228029900800204](https://doi.org/10.1177/096228029900800204)

Examples

```
data('reps')
agree_np(x = "x", y = "y", id = "id", data = reps, delta = 2)
```

agree_reps

Tests for Absolute Agreement with Replicates

Description**[Superseded]**

Development on `agree_reps()` is complete, and for new code we recommend switching to `agreement_limit()`, which is easier to use, has more features, and still under active development.

`agree_nest` produces an absolute agreement analysis for data where there is multiple observations per subject but the mean does not vary within subjects as described by Zou (2013). Output mirrors that of `agree_test` but CCC is calculated via U-statistics.

Usage

```
agree_reps(
  x,
  y,
  id,
  data,
  delta,
  agree.level = 0.95,
  conf.level = 0.95,
  prop_bias = FALSE,
  TOST = TRUE,
  ccc = TRUE
)
```

Arguments

<code>x</code>	Name of column with first measurement
<code>y</code>	Name of other column with the other measurement to compare to the first.
<code>id</code>	Column with subject identifier
<code>data</code>	Data frame with all data
<code>delta</code>	The threshold below which methods agree/can be considered equivalent, can be in any units. Equivalence Bound for Agreement.
<code>agree.level</code>	the agreement level required. Default is 95%. The proportion of data that should lie between the thresholds, for 95% limits of agreement this should be 0.95.
<code>conf.level</code>	the confidence level required. Default is 95%.
<code>prop_bias</code>	Logical indicator (TRUE/FALSE) of whether proportional bias should be considered for the limits of agreement calculations.
<code>TOST</code>	Logical indicator (TRUE/FALSE) of whether to use two one-tailed tests for the limits of agreement. Default is TRUE.
<code>ccc</code>	Calculate concordance correlation coefficient.

Value

Returns single list with the results of the agreement analysis.

- `loa`: a data frame of the limits of agreement including the average difference between the two sets of measurements, the standard deviation of the difference between the two sets of measurements and the lower and upper confidence limits of the difference between the two sets of measurements.
- `h0_test`: Decision from hypothesis test.
- `ccc.xy`: Lin's concordance correlation coefficient and confidence intervals using U-statistics.
- `call`: The matched call.
- `var_comp`: Table of Variance Components.
- `class`: The type of `simple_agree` analysis.

References

- Zou, G. Y. (2013). Confidence interval estimation for the Bland–Altman limits of agreement with multiple observations per individual. *Statistical methods in medical research*, 22(6), 630-642.
- King, TS and Chinchilli, VM. (2001). A generalized concordance correlation coefficient for continuous and categorical data. *Statistics in Medicine*, 20, 2131:2147.
- King, TS; Chinchilli, VM; Carrasco, JL. (2007). A repeated measures concordance correlation coefficient. *Statistics in Medicine*, 26, 3095:3113.
- Carrasco, JL; Phillips, BR; Puig-Martinez, J; King, TS; Chinchilli, VM. (2013). Estimation of the concordance correlation coefficient for repeated measures using SAS and R. *Computer Methods and Programs in Biomedicine*, 109, 293-304.

Examples

```
data('reps')
agree_reps(x = "x", y = "y", id = "id", data = reps, delta = 2)
```

 agree_test

Tests for Absolute Agreement

Description

[Superseded]

Development on `agree_test()` is complete, and for new code we recommend switching to `agreement_limit()`, which is easier to use, has more features, and still under active development.

The `agree_test` function calculates a variety of agreement statistics. The hypothesis test of agreement is calculated by the method described by Shieh (2019). Bland-Altman limits of agreement, and confidence intervals, are also provided (Bland & Altman 1999; Bland & Altman 1986). In addition, the concordance correlation coefficient (CCC; Lin 1989) is additional part of the output.

Usage

```
agree_test(
  x,
  y,
  delta,
  conf.level = 0.95,
  agree.level = 0.95,
  TOST = TRUE,
  prop_bias = FALSE
)
```

Arguments

<code>x</code>	Vector with first measurement
<code>y</code>	Vector with second measurement
<code>delta</code>	The threshold below which methods agree/can be considered equivalent, can be in any units. Often referred to as the "Equivalence Bound for Agreement" or "Maximal Allowable Difference".
<code>conf.level</code>	the confidence level required. Default is 95%.
<code>agree.level</code>	the agreement level required. Default is 95%. The proportion of data that should lie between the thresholds, for 95% limits of agreement this should be 0.95.
<code>TOST</code>	Logical indicator (TRUE/FALSE) of whether to use two one-tailed tests for the limits of agreement. Default is TRUE.
<code>prop_bias</code>	Logical indicator (TRUE/FALSE) of whether proportional bias should be considered for the limits of agreement calculations.

Value

Returns single list with the results of the agreement analysis.

- `shieh_test`: The TOST hypothesis test as described by Shieh.

- `ccc.xy`: Lin's concordance correlation coefficient and confidence intervals.
- `s.shift`: Scale shift from x to y.
- `l.shift`: Location shift from x to y.
- `bias`: a bias correction factor that measures how far the best-fit line deviates from a line at 45 degrees. No deviation from the 45 degree line occurs when `bias = 1`. See Lin 1989, page 258.
- `loa`: Data frame containing the limits of agreement calculations
- `h0_test`: Decision from hypothesis test.
- `call`: the matched call

References

Shieh (2019). Assessing Agreement Between Two Methods of Quantitative Measurements: Exact Test Procedure and Sample Size Calculation, Statistics in Biopharmaceutical Research, doi:10.1080/19466315.2019.1677495

Bland, J. M., & Altman, D. G. (1999). Measuring agreement in method comparison studies. Statistical methods in medical research, 8(2), 135-160.

Bland, J. M., & Altman, D. (1986). Statistical methods for assessing agreement between two methods of clinical measurement. The lancet, 327(8476), 307-310.

Lawrence, I., & Lin, K. (1989). A concordance correlation coefficient to evaluate reproducibility. Biometrics, 255-268.

Examples

```
data('reps')
agree_test(x=reps$x, y=reps$y, delta = 2)
```

agreement_limit

Limits of Agreement

Description

[Maturing]

A function for calculating for Bland-Altman limits of agreement based on the difference between two measurements (difference = $x - y$). Please note that the package developer recommends reporting/using tolerance limits (see "`tolerance_limit`" function).

Usage

```
agreement_limit(
  x,
  y,
  id = NULL,
  data,
```

```

data_type = c("simple", "nest", "reps"),
loa_calc = c("mover", "blandaltman"),
bound_type = c("iu", "joint"),
agree.level = 0.95,
alpha = 0.05,
prop_bias = FALSE,
log_tf = FALSE,
log_tf_display = c("ratio", "sympercent"),
lmer_df = c("satterthwaite", "asymptotic"),
lmer_limit = 3000
)

```

Arguments

x	Name of column with first measurement
y	Name of other column with the other measurement to compare to the first.
id	Column with subject identifier. Default is "id" if no entry is provided.
data	Data frame with all data.
data_type	The type of data structure. Options include "simple" (all independent data points), "nest" (nested data) and "reps" (replicated data points).
loa_calc	The method by which the limits of agreement confidence intervals are calculated. Options are "mover" (Methods of Recovering Variances method) or "blandlatman" (Bland-Altman method).
bound_type	Which claim the confidence bounds of the limits of agreement (<code>lower_loa_ci/upper_loa_ci</code>) support. "iu" (default) gives a one-sided $1 - \alpha$ bound on each limit (the outer ends of $1 - 2 * \alpha$ two-sided intervals). These give an intersection-union test, at level α , of whether both limits of agreement lie within a maximal allowable difference, but are not a joint $1 - \alpha$ interval for the limits. "joint" gives a one-sided $1 - \alpha/2$ bound on each limit (the outer ends of $1 - \alpha$ two-sided intervals, the Bland-Altman convention), with at least $1 - \alpha$ joint confidence for both limits. See details.
agree.level	the agreement level required. Default is 95%. The proportion of data that should lie between the thresholds, for 95% limits of agreement this should be 0.95.
alpha	The alpha-level for confidence levels.
prop_bias	Logical indicator (TRUE/FALSE) of whether proportional bias should be considered for the limits of agreement calculations. Note that a non-zero slope of the differences on the average can appear without any true proportional bias whenever the two methods have unequal measurement error variances, because $\text{cov}(\text{difference}, \text{average}) = (\text{var}(x) - \text{var}(y)) / 2$ (Bland & Altman, 1999); errors-in-variables methods such as <code>dem_reg()</code> or <code>pb_reg()</code> are better suited to assessing proportional bias.
log_tf	Calculate limits of agreement using log-transformed data.
log_tf_display	The type of presentation for log-transformed results. The differences between methods can be displayed as a "ratio" or "sympercent".
lmer_df	Degrees of freedom method, only matters for if <code>data_type</code> is "nest". Default is "satterthwaite". The "asymptotic" method is faster but more liberal.

`lmer_limit` Sample size limit for degrees of freedom method. If number of observations exceeds this limit, then the "asymptotic" method is utilized.

Details

The limits of agreement (LoA) are calculated in this function are based on the method originally detailed by Bland & Altman (1986 & 1999). The `loa_calc` allow users to specify the calculative method for the LoA which can be based on Bland-Altman (1999) (`loa_calc = "blandaltman"`), or by the more accurate MOVER method of Zou (2013) and Donner & Zou (2012) (`loa_calc = "mover"`).

The confidence bounds on the limits of agreement answer one of two questions, set by `bound_type`:

- "iu" (default): "Can I conclude agreement within plus or minus delta?" If `lower_loa_ci > -delta` and `upper_loa_ci < delta`, reject at level α that either limit of agreement lies outside plus or minus delta. `conf.level` ($1 - \alpha$) applies to each side separately; both bounds hold together only about $1 - 2 * \alpha$ of the time.
- "joint": "Where are the limits of agreement?" With at least $1 - \alpha$ confidence, both limits of agreement lie between `lower_loa_ci` and `upper_loa_ci`. Used as a test against plus or minus delta, the error rate is at most $\alpha/2$ (conservative).

Because a one-sided bound on the $(1 - \text{agree.level})/2$ percentile is a one-sided tolerance bound, these bounds are equal-tailed tolerance bounds; they target the same quantities as `tolerance_limit(bound_type = "iu")`. The bias confidence interval (`lower.CL/upper.CL`) is always a two-sided $1 - \alpha$ interval.

Value

Returns single `loa` class object with the results of the agreement analysis.

- `loa`: A data frame containing the Limits of Agreement.
- `call`: The matched call.

References

MOVER methods:

Zou, G. Y. (2013). Confidence interval estimation for the Bland–Altman limits of agreement with multiple observations per individual. *Statistical methods in medical research*, 22(6), 630-642.

Donner, A., & Zou, G. Y. (2012). Closed-form confidence intervals for functions of the normal mean and standard deviation. *Statistical Methods in Medical Research*, 21(4), 347-359.

Bland & Altman methods:

Bland, J. M., & Altman, D. (1986). Statistical methods for assessing agreement between two methods of clinical measurement. *The Lancet*, 327(8476), 307-310.

Bland, J. M., & Altman, D. (1999). Measuring agreement in method comparison studies. *Statistical methods in medical research*, 8(2), 135-160.

Bland, J. M., & Altman, D. G. (1996). Statistics notes: measurement error proportional to the mean. *BMJ*, 313(7049), 106.

Examples

```
data('reps')

# Simple
agreement_limit(x = "x", y = "y", data = reps)

# Replicates
agreement_limit(x = "x", y = "y", data = reps, id = "id", data_type = "rep")

# Nested
agreement_limit(x = "x", y = "y", data = reps, id = "id", data_type = "nest")
```

blandPowerCurve

*Power Curve for Bland-Altman Limits of Agreement***Description**

[Maturing] This function calculates the power for the Bland-Altman method under varying parameter settings and for a range of sample sizes.

Usage

```
blandPowerCurve(
  samplesizes = seq(10, 100, 1),
  mu = 0,
  SD,
  delta,
  conf.level = 0.95,
  agree.level = 0.95
)
```

Arguments

<code>samplesizes</code>	vector of samples sizes at which to estimate power.
<code>mu</code>	mean of differences
<code>SD</code>	standard deviation of differences
<code>delta</code>	The threshold below which methods agree/can be considered equivalent, can be in any units. Equivalence Bound for Agreement. More than one delta can be provided.
<code>conf.level</code>	the confidence level(s) required. Default is 95%. More than one confidence level can be provided.
<code>agree.level</code>	the agreement level(s) required. Default is 95%. The proportion of data that should lie between the thresholds, for 95% limits of agreement this should be 0.95. More than one confidence level can be provided.

Value

A dataframe is returned containing the power analysis results. The results can then be plotted with the `plot.powerCurve` function.

References

Lu, M. J., et al. (2016). Sample Size for Assessing Agreement between Two Methods of Measurement by Bland-Altman Method. The international journal of biostatistics, 12(2), [doi:10.1515/ijb20150039](https://doi.org/10.1515/ijb20150039)

Examples

```
powerCurve <- blandPowerCurve(samplesizes = seq(10, 200, 1),
mu = 0,
SD = 3.3,
delta = 8,
conf.level = .95,
agree.level = .95)
# Plot the power curve
plot(powerCurve, type = 1)
# Find at what N power of .8 is achieved
find_n(powerCurve, power = .8)

# If the desired power is not found then
## Sample size range must be expanded
```

ccc_test

Concordance Correlation Coefficient

Description

Computes Lin's Concordance Correlation Coefficient (CCC) and performs a hypothesis test, returning an object of class "htest".

Usage

```
ccc_test(
  x,
  y = NULL,
  id = NULL,
  data = NULL,
  data_type = c("simple", "reps", "nest"),
  conf.level = 0.95,
  null.value = 0,
  alternative = c("two.sided", "greater", "less")
)
```

Arguments

x	Either a numeric vector of measurements for method 1, or a character string naming the column in data containing method 1 measurements.
y	Either a numeric vector of measurements for method 2, or a character string naming the column in data containing method 2 measurements.
id	A character string naming the column in data containing subject identifiers. Required when data_type is "reps" or "nest".
data	An optional data frame containing the variables named in x, y, and id.
data_type	Character string specifying the data structure: "simple" for paired data with one observation per subject (default), "reps" for replicate data where subjects are measured multiple times, "nest" for nested data structures.
conf.level	Confidence level for the interval, default is 0.95.
null.value	The hypothesized value of CCC under the null hypothesis, default is 0.
alternative	Character string specifying the alternative hypothesis: "two.sided" (default), "greater", or "less".

Details

The concordance correlation coefficient measures the agreement between two measurements of the same quantity. It combines measures of precision (Pearson correlation) and accuracy (bias correction factor) to determine how far the observed data deviate from the line of perfect concordance (45-degree line through the origin).

For simple paired data, Lin's original method is used. For replicate or nested data, U-statistics are employed following King, Chinchilli, and Carrasco.

The hypothesis test is performed using Fisher's Z-transformation of the CCC:

$$Z = 0.5 \times \log \left(\frac{1 + CCC}{1 - CCC} \right)$$

Value

A list with class "htest" containing the following components:

statistic	The Z-statistic (Fisher-transformed).
parameter	The sample size (n for simple data, number of subjects for reps/nest).
stderr	The standard errors for CCC and Z-transformed CCC.
p.value	The p-value for the test.
conf.int	A confidence interval for the CCC.
estimate	The estimated CCC.
null.value	The specified hypothesized value of CCC.
alternative	A character string describing the alternative hypothesis.
method	A character string indicating the method used.
data.name	A character string giving the names of the data.

References

- Lin, L. I. (1989). A concordance correlation coefficient to evaluate reproducibility. *Biometrics*, 45(1), 255-268.
- King, T. S., & Chinchilli, V. M. (2001). A generalized concordance correlation coefficient for continuous and categorical data. *Statistics in Medicine*, 20(14), 2131-2147.
- King, T. S., Chinchilli, V. M., & Carrasco, J. L. (2007). A repeated measures concordance correlation coefficient. *Statistics in Medicine*, 26(16), 3095-3113.

Examples

```
# Simple paired data using vectors
x <- c(1, 2, 3, 4, 5)
y <- c(1.1, 2.2, 2.9, 4.1, 5.2)
ccc_test(x, y)

# Using data frame interface
data("reps")
# Simple data
ccc_test(x = "x", y = "y", data = reps)

# Test against specific null value
ccc_test(x = "x", y = "y", data = reps,
         null.value = 0.8, alternative = "greater")

# Replicate data
ccc_test(x = "x", y = "y", id = "id", data = reps, data_type = "reps")

# Nested data
ccc_test(x = "x", y = "y", id = "id", data = reps, data_type = "nest")
```

dem_reg

Deming Regression

Description

[Stable]

A function for fitting a straight line to two-dimensional data (i.e., X and Y) that are measured with error.

Usage

```
dem_reg(
  formula = NULL,
  data,
  id = NULL,
  x = NULL,
```

```

    y = NULL,
    conf.level = 0.95,
    weighted = FALSE,
    weights = NULL,
    error.ratio = 1,
    model = TRUE,
    keep_data = FALSE,
    ...
)

```

Arguments

formula	A formula of the form $y \sim x$ specifying the model. If provided, takes precedence over x and y arguments.
data	Data frame with all data.
id	Column with subject identifier (optional).
x	Name of column with first measurement (deprecated in favor of formula interface).
y	Name of other column with the other measurement to compare to the first (deprecated in favor of formula interface).
conf.level	The confidence level required. Default is 95%.
weighted	Logical indicator (TRUE/FALSE) for whether to use weighted Deming regression. Default is FALSE.
weights	an optional vector of weights to be used in the fitting process. Should be NULL or a numeric vector.
error.ratio	Ratio of the measurement error variance of x to that of y, i.e., $\text{var}(x)/\text{var}(y)$, where $\text{var}()$ denotes the error (not total) variance of each method. Default is 1 (equal error variances). Values greater than 1 indicate that x is measured with more error than y. This argument is ignored if subject identifiers are provided.
model	Logical. If TRUE (default), the model frame is stored in the returned object. This is needed for methods like <code>plot()</code> , <code>fitted()</code> , <code>residuals()</code> , and <code>predict()</code> to work without supplying data. If FALSE, the model frame is not stored (saves memory for large datasets), but these methods will require a data argument.
keep_data	Logical indicator (TRUE/FALSE). If TRUE, the jackknife samples are returned; default is FALSE.
...	Additional arguments (currently unused).

Details

This function provides a Deming regression analysis wherein the sum of distances in both x and y direction is minimized. Deming regression, also known as error-in-variable regression, is useful in situations where both X & Y are measured with error. The use of Deming regression is beneficial when comparing to methods for measuring the same continuous variable.

Currently, the `dem_reg` function covers simple Deming regression and weighted Deming regression. Weighted Deming regression can be used by setting the `weighted` argument to `TRUE`. The weights can be provided by the user or can be calculated within function.

If the data are measured in replicates, then the measurement error can be directly derived from the data. This can be accomplished by indicating the subject identifier with the `id` argument. When the replicates are not available in the data, then the ratio of error variances, $\text{var}(x)/\text{var}(y)$, can be provided with the `error.ratio` argument. When subject identifiers are provided, the error ratio is instead estimated from the replicates as the within-subject variance of x divided by the within-subject variance of y .

Value

The function returns a `simple_eiv` (eiv meaning "error in variables") object with the following components:

- `coefficients`: Named vector of coefficients (intercept and slope).
- `residuals`: Optimized residuals from the fitted model.
- `fitted.values`: Estimated true Y values ($\hat{Y}$).
- `model_table`: Data frame presenting the full results from the Deming regression analysis.
- `vcov`: Variance-covariance matrix for slope and intercept.
- `df.residual`: Residual degrees of freedom.
- `call`: The matched call.
- `terms`: The terms object used.
- `xlevels`: (Only for models with factors) levels of factors.
- `model`: The model frame.
- `x_vals`: Original x values used in fitting.
- `y_vals`: Original y values used in fitting.
- `x_hat`: Estimated true X values.
- `y_hat`: Estimated true Y values.
- `error.ratio`: Error ratio used in fitting.
- `weighted`: Whether weighted regression was used.
- `weights`: Weights used in fitting.
- `conf.level`: Confidence level used.
- `resamples`: List containing resamples from jackknife procedure (if `keep_data = TRUE`).

Interface Change

The `x` and `y` arguments are deprecated. Please use the formula interface instead:

- Old: `dem_reg(x = "x_var", y = "y_var", data = df)`
- New: `dem_reg(y_var ~ x_var, data = df)`

References

- Linnet, K. (1990) Estimation of the linear relationship between the measurements of two methods with proportional errors. *Statistics in Medicine*, 9, 1463-1473.
- Linnet, K. (1993). Evaluation of regression procedures for methods comparison studies. *Clinical chemistry*, 39, 424-432.
- Sadler, W.A. (2010). Joint parameter confidence regions improve the power of parametric regression in method-comparison studies. *Accreditation and Quality Assurance*, 15, 547-554.

Examples

```
## Not run:  
# New formula interface (recommended)  
model <- dem_reg(y ~ x, data = mydata)  
  
# Old interface (still works with deprecation warning)  
model <- dem_reg(x = "x", y = "y", data = mydata)  
  
## End(Not run)
```

deming_power

Simulate Deming Regression Power

Description

[Experimental]

Functions for conducting power analysis and sample size determination for Deming regression in method comparison studies. These functions help determine the sample size needed to detect specified biases (proportional and/or constant) between two measurement methods.

Estimates statistical power to detect deviations from the line of identity using simulated data with known properties.

Usage

```
deming_power_sim(  
  n_sims = 1000,  
  sample_size = 50,  
  x_range = c(10, 100),  
  x_dist = "uniform",  
  actual_slope = 1.05,  
  actual_intercept = 0,  
  ideal_slope = 1,  
  ideal_intercept = 0,  
  y_var_params = list(beta1 = 1, beta2 = 0, J = 1, type = "constant"),  
  x_var_params = list(beta1 = 1, beta2 = 0, J = 1, type = "constant"),  
  weighted = FALSE,  
  conf.level = 0.95  
)
```

Arguments

n_sims	Number of simulation iterations. Default is 1000.
sample_size	Sample size (number of paired observations) per simulation.
x_range	Numeric vector of length 2 specifying min and max of X values (e.g., c(10, 100)).
x_dist	Character specifying distribution of X values: "uniform", "central", or "right_skewed".
actual_slope	Actual slope used to generate data (e.g., 1.05 for 5% proportional bias).
actual_intercept	Actual intercept used to generate data.
ideal_slope	Hypothesized slope to test against (typically 1 for identity line).
ideal_intercept	Hypothesized intercept to test against (typically 0 for identity line).
y_var_params	List with Y variance parameters: beta1, beta2, J, type. Type can be "constant", "proportional", or "power". For power function: $\sigma^2 = (\beta_1 + \beta_2 * U)^J$
x_var_params	List with X variance parameters (same structure as y_var_params).
weighted	Logical. Use weighted Deming regression? Default is FALSE.
conf.level	Confidence level for tests. Default is 0.95.

Details**Power Analysis for Deming Regression**

This function generates simulated datasets with specified error characteristics and tests whether confidence intervals and joint confidence regions detect deviations from hypothesized values. The joint confidence region typically provides higher statistical power, especially when the X-range is narrow (high slope-intercept correlation).

The variance functions allow flexible modeling of heteroscedastic errors:

- constant: $\sigma^2 = \beta_1$
- proportional: $\sigma^2 = \beta_1 * U^2$ (constant CV%)
- power: $\sigma^2 = (\beta_1 + \beta_2 * U)^J$

Value

A list of class "deming_power" containing:

power_ci_slope	Power based on slope confidence interval
power_ci_intercept	Power based on intercept confidence interval
power_either_ci	Power when either CI detects difference
power_joint	Power based on joint confidence region
settings	List of simulation settings
advantage	Difference in power between joint region and CIs

References

Sadler, W.A. (2010). Joint parameter confidence regions improve the power of parametric regression in method-comparison studies. *Accreditation and Quality Assurance*, 15, 547-554.

Examples

```
## Not run:
# Simple example: detect 5% proportional bias with constant variance
power_result <- deming_power_sim(
  n_sims = 500,
  sample_size = 50,
  x_range = c(10, 100),
  actual_slope = 1.05,
  ideal_slope = 1.0,
  y_var_params = list(beta1 = 25, beta2 = 0, J = 1, type = "constant"),
  x_var_params = list(beta1 = 20, beta2 = 0, J = 1, type = "constant")
)
print(power_result)

# More complex: heteroscedastic errors
power_result2 <- deming_power_sim(
  n_sims = 500,
  sample_size = 75,
  x_range = c(1, 100),
  actual_slope = 1.03,
  y_var_params = list(beta1 = 0.5, beta2 = 0.05, J = 2, type = "power"),
  x_var_params = list(beta1 = 0.4, beta2 = 0.04, J = 2, type = "power"),
  weighted = TRUE
)

## End(Not run)
```

deming_sample_size

Determine Required Sample Size for Deming Regression

Description

[Experimental]

Automatically determines the minimum sample size needed to achieve target statistical power for detecting specified bias in method comparison studies using Deming regression.

Usage

```
deming_sample_size(
  target_power = 0.9,
  initial_n = 20,
  max_n = 500,
  n_sims = 500,
```

```

    use_joint = TRUE,
    step_size = 5,
    ...
  )

```

Arguments

target_power	Desired statistical power (e.g., 0.80 or 0.90). Default is 0.90.
initial_n	Starting sample size for search. Default is 20.
max_n	Maximum sample size to try. Default is 500.
n_sims	Number of simulations per sample size tested. Default is 500.
use_joint	Logical. If TRUE, optimizes for joint region power; if FALSE, for CI power.
step_size	Step size for sample size increments. Default is 5.
...	Additional arguments passed to deming_power_sim()

Details

This function performs a grid search over sample sizes to find the minimum N needed to achieve the target power. It tests both confidence interval and joint confidence region approaches, allowing comparison of required sample sizes.

Using joint confidence regions typically requires 20-50% fewer samples than confidence intervals when the measurement range is narrow (max:min ratio < 10:1).

Value

A list of class "deming_sample_size" containing:

n_required_ci	Required N for confidence intervals
n_required_joint	Required N for joint confidence region
target_power	Target power level
power_curve	Data frame with N and power for both methods
reduction_n	Sample size reduction using joint method
reduction_pct	Percentage reduction

Examples

```

## Not run:
# Determine N needed for 90% power to detect 5% bias
sample_size_result <- deming_sample_size(
  target_power = 0.90,
  initial_n = 30,
  max_n = 200,
  n_sims = 500,
  x_range = c(20, 200),
  actual_slope = 1.05,
  ideal_slope = 1.0,

```

```

y_var_params = list(beta1 = 1, beta2 = 0.05, J = 2, type = "power"),
x_var_params = list(beta1 = 0.8, beta2 = 0.04, J = 2, type = "power")
)

print(sample_size_result)
plot(sample_size_result)

## End(Not run)

```

jmvagree

Simple Agreement Analysis

Description

Simple Agreement Analysis

Usage

```

jmvagree(
  data,
  method1,
  method2,
  ciWidth = 95,
  agreeWidth = 95,
  testValue = 2,
  CCC = TRUE,
  plotbland = TRUE,
  plotcon = FALSE,
  plotcheck = FALSE,
  prop_bias = FALSE,
  xlabel = "Average of Both Methods",
  ylabel = "Difference between Methods"
)

```

Arguments

data	Data
method1	Name of column containing 1st Vector of data
method2	Name of column containing Vector of data
ciWidth	a number between 50 and 99.9 (default: 95), the width of confidence intervals
agreeWidth	a number between 50 and 99.9 (default: 95), the width of agreement limits
testValue	a number specifying the limit of agreement
CCC	TRUE or FALSE (default), produce CCC table
plotbland	TRUE or FALSE (default), for Bland-Altman plot

plotcon	TRUE or FALSE (default), for Bland-Altman plot
plotcheck	TRUE or FALSE (default), assumptions plots
prop_bias	TRUE or FALSE
xlabel	The label for the x-axis on the BA plot
ylabel	The label for the y-axis on the BA plot

Value

A results object containing:

results\$text	a html
results\$blandtab	a table
results\$ccctab	a table
results\$plotba	an image
results\$plotcon	an image
results\$plotcheck	an image

Tables can be converted to data frames with asDF or [as.data.frame](#). For example:

```
results$blandtab$asDF
as.data.frame(results$blandtab)
```

jmvagreemulti	<i>Nested/Replicate Data Agreement Analysis</i>
---------------	---

Description

Nested/Replicate Data Agreement Analysis

Usage

```
jmvagreemulti(
  data,
  method1,
  method2,
  id,
  ciWidth = 95,
  agreeWidth = 95,
  testValue = 2,
  CCC = TRUE,
  valEq = FALSE,
  plotbland = FALSE,
  plotcon = FALSE,
  prop_bias = FALSE,
  xlabel = "Average of Both Methods",
  ylabel = "Difference between Methods"
)
```

Arguments

data	Data
method1	Name of column containing 1st Vector of data
method2	Name of column containing Vector of data
id	Name of column containing subject identifier
ciWidth	a number between 50 and 99.9 (default: 95), the width of confidence intervals
agreeWidth	a number between 50 and 99.9 (default: 95), the width of agreement limits
testValue	a number specifying the limit of agreement
CCC	TRUE or FALSE (default), produce CCC table
valEq	.
plotbland	TRUE or FALSE (default), for Bland-Altman plot
plotcon	TRUE or FALSE (default), for Line of identity plot
prop_bias	TRUE or FALSE
xlabel	The label for the x-axis on the BA plot
ylabel	The label for the y-axis on the BA plot

Value

A results object containing:

results\$text	a preformatted
results\$blandtab	a table
results\$ccctab	a table
results\$plotba	an image
results\$plotcon	an image

Tables can be converted to data frames with `asDF` or `as.data.frame`. For example:

```
results$blandtab$asDF
```

```
as.data.frame(results$blandtab)
```

jmvdeming

Deming Regression

Description

Deming Regression

Usage

```
jmvdeming(
  data,
  method1,
  method2,
  ciWidth = 95,
  testValue = 1,
  plotcon = FALSE,
  plotcheck = FALSE,
  weighted = FALSE,
  xlabel = "Method: 1",
  ylabel = "Method: 2"
)
```

Arguments

data	Data
method1	Name of column containing 1st Vector of data
method2	Name of column containing Vector of data
ciWidth	a number between 50 and 99.9 (default: 95), the width of confidence intervals
testValue	Ratio of the error variances, Method 1 (x) / Method 2 (y). Default is 1.
plotcon	TRUE or FALSE (default), for Bland-Altman plot
plotcheck	TRUE or FALSE (default), assumptions plots
weighted	TRUE or FALSE
xlabel	The label for the x-axis
ylabel	The label for the y-axis

Value

A results object containing:

results\$text	a html
results\$demtab	a table
results\$plotcon	an image
results\$plotcheck	an image

Tables can be converted to data frames with `asDF` or `as.data.frame`. For example:

```
results$demtab$asDF
as.data.frame(results$demtab)
```

jmvreli*Reliability Analysis*

Description

Reliability Analysis

Usage

```
jmvreli(data, vars, ciWidth = 95, desc = FALSE, plots = FALSE)
```

Arguments

data	the data as a data frame
vars	a list of the column names containing the measurements for reliability analysis.
ciWidth	a number between 50 and 99.9 (default: 95), the width of confidence intervals
desc	TRUE or FALSE (default), provide table of variance components
plots	TRUE or FALSE (default), plot data

Value

A results object containing:

results\$text	a html
results\$icctab	a table
results\$varstab	a table
results\$plots	an image

Tables can be converted to data frames with `asDF` or `as.data.frame`. For example:

```
results$icctab$asDF  
as.data.frame(results$icctab)
```

joint_test*Joint Confidence Region Test for Method Agreement*

Description

Tests whether the estimated intercept and slope jointly fall within a confidence region around specified ideal values (typically intercept=0 and slope=1 for method comparison studies).

Usage

```
joint_test(object, ...)

## S3 method for class 'simple_eiv'
joint_test(
  object,
  ideal_intercept = 0,
  ideal_slope = 1,
  conf.level = 0.95,
  test_method = c("F", "asymptotic"),
  ...
)
```

Arguments

<code>object</code>	A <code>simple_eiv</code> object from <code>dem_reg()</code> or <code>pb_reg()</code> .
<code>...</code>	Additional arguments (currently unused).
<code>ideal_intercept</code>	The hypothesized intercept value (default: 0).
<code>ideal_slope</code>	The hypothesized slope value (default: 1).
<code>conf.level</code>	Confidence level for the test (default: 0.95).
<code>test_method</code>	Reference distribution for the test. "F" (default) uses the finite-sample F(2, n-2) distribution per Sadler (2010). "asymptotic" uses the chi-squared(2) distribution, which is the large-sample approximation. The two converge for large n.

Details

The test computes the Mahalanobis distance between the estimated coefficients and the hypothesized values using the variance-covariance matrix of the estimates. By default, the finite-sample F(2, n-2) reference distribution is used (Sadler, 2010), which is more conservative than the asymptotic chi-squared(2) approximation for small samples.

For Deming regression, the variance-covariance matrix is computed via jackknife. For Passing-Bablok regression, bootstrap resampling must have been performed (i.e., `boot_ci = TRUE` in the original call).

Value

An object of class `htest` containing:

statistic The Mahalanobis distance.

parameter Degrees of freedom. For the F-based test, `df1` and `df2`; for the asymptotic test, `df` (always 2).

p.value The p-value for the test.

conf.int The confidence level used.

estimate Named vector of estimated intercept and slope.

null.value Named vector of hypothesized intercept and slope.

- alternative** Description of the alternative hypothesis.
- method** Description of the test.
- data.name** Name of the input object.

References

Sadler, W.A. (2010). Joint parameter confidence regions improve the power of parametric regression in method-comparison studies. *Accreditation and Quality Assurance*, 15, 547-554.

loa-methods	<i>Methods for loa objects</i>
-------------	--------------------------------

Description

Methods defined for objects returned from the `agreement_limit` function.

Usage

```
## S3 method for class 'loa'
print(x, digits = 4, ...)

## S3 method for class 'loa'
plot(
  x,
  geom = c("geom_point", "geom_bin2d", "geom_density_2d", "geom_density_2d_filled",
    "stat_density_2d"),
  delta = NULL,
  ...
)

## S3 method for class 'loa'
check(x, ...)
```

Arguments

- `x` object of class `loa` as returned from a `agreement_limit` function.
- `digits` The number of digits to print.
- `...` further arguments passed through, see description of return value for details. [agreement_limit](#).
- `geom` String naming the type of geometry to display the data points. Default is "geom_point". Other options include: "geom_bin2d", "geom_density_2d", "geom_density_2d_filled", and "stat_density_2d".
- `delta` The maximal allowable difference.

Value

`print` Prints short summary of the Limits of Agreement.

`plot` Returns a plot of the limits of agreement.

`check` Returns plots testing the assumptions of a Bland-Altman analysis. P-values for the normality and heteroskedascity tests are provided as captions to the plot.

loa_lme

*Limits of Agreement with Linear Mixed Effects***Description****[Stable]**

This function allows for the calculation of (parametric) bootstrapped limits of agreement when there are multiple observations per subject. The package author recommends using `tolerance_limit` as an alternative to this function.

Usage

```
loa_lme(
  diff,
  avg,
  condition = NULL,
  id,
  data,
  type = c("perc", "norm", "basic"),
  conf.level = 0.95,
  agree.level = 0.95,
  replicates = 999,
  prop_bias = FALSE,
  het_var = FALSE
)
```

Arguments

<code>diff</code>	Column name of the data frame that includes the difference between the 2 measurements of interest.
<code>avg</code>	Column name of the data frame that includes the average of the 2 measurements of interest.
<code>condition</code>	Column name indicating different conditions subjects were tested under. This can be left missing if there are no differing conditions to be tested.
<code>id</code>	Column name indicating the subject/participant identifier
<code>data</code>	A data frame containing the variables within the model.
<code>type</code>	A character string representing the type of bootstrap confidence intervals. Only "norm", "basic", and "perc" currently supported. Bias-corrected and accelerated, bca, is the default. See <code>?boot::boot.ci</code> for more details.

<code>conf.level</code>	The confidence level required. Default is 95%.
<code>agree.level</code>	The agreement level required. Default is 95%.
<code>replicates</code>	The number of bootstrap replicates. Passed on to the boot function. Default is 999.
<code>prop_bias</code>	Logical indicator (default is FALSE) of whether proportional bias should be considered for the limits of agreement calculations.
<code>het_var</code>	Logical indicator (default is FALSE) of whether to assume homogeneity of variance in each condition.

Value

Returns single list with the results of the agreement analysis.

- `var_comp`: Table of variance components
- `loa`: A data frame of the limits of agreement including the average difference between the two sets of measurements, the standard deviation of the difference between the two sets of measurements and the lower and upper confidence limits of the difference between the two sets of measurements.
- `call`: The matched call.

References

Parker, R. A., Weir, C. J., Rubio, N., Rabinovich, R., Pinnock, H., Hanley, J., McLoughan, L., Drost, E.M., Mantoani, L.C., MacNee, W., & McKinstry, B. (2016). "Application of mixed effects limits of agreement in the presence of multiple sources of variability: exemplar from the comparison of several devices to measure respiratory rate in COPD patients". PLOS One, 11(12), e0168321. [doi:10.1371/journal.pone.0168321](https://doi.org/10.1371/journal.pone.0168321)

loa_mermod-methods	<i>Methods for loa_mermod objects</i>
--------------------	---------------------------------------

Description

Methods defined for objects returned from the `loa_lme`.

Usage

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

## S3 method for class 'loa_mermod'
plot(
  x,
  x_label = "Average of Both Methods",
  y_label = "Difference Between Methods",
  geom = "geom_point",
```

```
smooth_method = NULL,
smooth_se = TRUE,
...
)

## S3 method for class 'loa_mermod'
check(x, ...)
```

Arguments

x	object of class loa_mermod.
...	further arguments passed through, see description of return value for details. loa_mixed .
x_label	Label for x-axis.
y_label	Label for y-axis.
geom	String naming the type of geometry to display the data points. Default is "geom_point". Other options include: "geom_bin2d", "geom_density_2d", "geom_density_2d_filled", and "stat_density_2d".
smooth_method	Smoothing method (function) to use, accepts either NULL or a character vector, e.g. "lm", "glm", "gam", "loess" or a function. Default is NULL, which will not include a trend line.
smooth_se	Display confidence interval around smooth?

Value

- print Prints short summary of the Limits of Agreement
- plot Returns a plot of the limits of agreement

loa_mixed	<i>Mixed Effects Limits of Agreement</i>
-----------	--

Description

[Deprecated]

loa_mixed() is outdated, and for new code we recommend switching to loa_lme() or tolerance_limit, which are easier to use, have more features, and are still under active development.

This function allows for the calculation of bootstrapped limits of agreement when there are multiple observations per subject.

Usage

```
loa_mixed(
  diff,
  condition,
  id,
  data,
  plot.xaxis = NULL,
  delta,
  conf.level = 0.95,
  agree.level = 0.95,
  replicates = 1999,
  type = "bca"
)
```

Arguments

<code>diff</code>	column name of the data frame that includes the continuous measurement of interest.
<code>condition</code>	column name indicating different conditions subjects were tested under.
<code>id</code>	column name indicating the subject/participant identifier
<code>data</code>	A data frame containing the variables within the model.
<code>plot.xaxis</code>	column name indicating what to plot on the x.axis for the Bland-Altman plots. If this argument is missing or set to NULL then no plot will be produced.
<code>delta</code>	The threshold below which methods agree/can be considered equivalent, can be in any units. Equivalence Bound for Agreement.
<code>conf.level</code>	the confidence level required. Default is 95%.
<code>agree.level</code>	the agreement level required. Default is 95%.
<code>replicates</code>	the number of bootstrap replicates. Passed on to the boot function. Default is 1999.
<code>type</code>	A character string representing the type of bootstrap confidence intervals. Only "norm", "basic", "bca", and "perc" currently supported. Bias-corrected and accelerated, bca, is the default. See <code>?boot::boot.ci</code> for more details.

Value

Returns single list with the results of the agreement analysis.

- `var_comp`: Table of variance components
- `loa`: a data frame of the limits of agreement including the average difference between the two sets of measurements, the standard deviation of the difference between the two sets of measurements and the lower and upper confidence limits of the difference between the two sets of measurements.
- `h0_test`: Decision from hypothesis test.

- `bland_alt.plot`: Simple Bland-Altman plot. Red line are the upper and lower bounds for shieh test; grey box is the acceptable limits (delta). If the red lines are within the grey box then the shieh test should indicate 'reject h_0 ', or to reject the null hypothesis that this not acceptable agreement between x & y .
- `conf.level`: Returned as input.
- `agree.level`: Returned as input.

References

Parker, R. A., Weir, C. J., Rubio, N., Rabinovich, R., Pinnock, H., Hanley, J., McLoughan, L., Drost, E.M., Mantoani, L.C., MacNee, W., & McKinstry, B. (2016). "Application of mixed effects limits of agreement in the presence of multiple sources of variability: exemplar from the comparison of several devices to measure respiratory rate in COPD patients". Plos One, 11(12), e0168321. [doi:10.1371/journal.pone.0168321](https://doi.org/10.1371/journal.pone.0168321)

loa_mixed_bs-methods *Methods for loa_mixed_bs objects*

Description

Methods defined for objects returned from the `loa_mixed` functions.

Usage

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

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

Arguments

`x` object of class `loa_mixed_bs` as returned from `loa_mixed`

`...` further arguments passed through, see description of return value for details. [loa_mixed](#).

Value

`print` Prints short summary of the Limits of Agreement

`plot` Returns a plot of the limits of agreement

Description**[Experimental]**

A robust, nonparametric method for fitting a straight line to two-dimensional data where both variables (X and Y) are measured with error. Particularly useful for method comparison studies.

Usage

```
pb_reg(
  formula,
  data,
  id = NULL,
  method = c("scissors", "symmetric", "invariant"),
  conf.level = 0.95,
  weights = NULL,
  error.ratio = 1,
  replicates = 0,
  se_method = c("bootstrap", "jackknife", "dufey"),
  model = TRUE,
  keep_data = TRUE,
  ...
)
```

Arguments

formula	A formula of the form $y \sim x$ specifying the model.
data	Data frame with all data.
id	Column with subject identifier (optional). If provided, measurement error ratio is calculated from replicate measurements.
method	Method for Passing-Bablok estimation. Options are: "scissors": Scissors estimator (1988) - most robust, scale invariant (default) "symmetric": Original Passing-Bablok (1983) - symmetric around 45-degree line "invariant": Scale-invariant method (1984) - adaptive reference line
conf.level	The confidence level required. Default is 95%.
weights	An optional vector of case weights to be used in the fitting process. Should be NULL or a numeric vector.
error.ratio	Ratio of measurement error variances ($\text{var}(x)/\text{var}(y)$). Default is 1. This argument is ignored if subject identifiers are provided via id.

replicates	Number of resamples for confidence intervals and the variance-covariance matrix. For <code>se_method = "bootstrap"</code> this is the number of bootstrap resamples; for <code>se_method = "jackknife"</code> it is the number of random delete-d subsets. If 0 (default), analytical confidence intervals are used and no variance-covariance matrix is returned (unless <code>se_method = "dufey"</code>). Resampling is recommended for weighted data and 'invariant' or 'scissors' methods.
se_method	Method used to estimate the variance-covariance matrix and confidence intervals of the coefficients. Options are: • "bootstrap": Nonparametric pairs (case) bootstrap with percentile confidence intervals (default). Used when <code>replicates > 0</code>. • "jackknife": Delete-d jackknife with $d = \text{floor}(n/2)$, using replicates random subsets (Shao & Wu, 1989). Used when <code>replicates > 0</code>. • "dufey": [Experimental] Analytic sandwich estimator of Dufey (2020). Does not require resampling (<code>replicates</code> is ignored). Only available for the "scissors" method without case weights and with <code>error.ratio = 1</code>. See Details before using it for joint tests.
model	Logical. If TRUE (default), the model frame is stored in the returned object. This is needed for methods like <code>plot()</code> , <code>fitted()</code> , <code>residuals()</code> , and <code>predict()</code> to work without supplying data. If FALSE, the model frame is not stored (saves memory for large datasets), but these methods will require a data argument.
keep_data	Logical indicator (TRUE/FALSE). If TRUE, intermediate calculations are returned; default is FALSE.
...	Additional arguments (currently unused).

Details

Passing-Bablok regression is a robust nonparametric method that estimates the slope as the shifted median of all possible slopes between pairs of points. The intercept is then calculated as the median of $y - \text{slope} \cdot x$. This method is particularly useful when:

- Both X and Y are measured with error
- You want a robust method not sensitive to outliers
- The relationship is assumed to be linear
- X and Y are highly positively correlated

Methods:

Three Passing-Bablok methods are available:

"scissors" (default): The scissors estimator (1988), most robust and scale-invariant. Uses the median of absolute values of angles.

"symmetric": The original method (1983), symmetric about the $y = x$ line. Uses the line $y = -x$ as the reference for partitioning points.

"invariant": Scale-invariant method (1984). First finds the median angle of slopes below the horizontal, then uses this as the reference line.

Measurement Error Handling:

If the data are measured in replicates, then the measurement error ratio can be directly derived from the data. This can be accomplished by indicating the subject identifier with the `id` argument. When replicates are not available in the data, then the ratio of error variances ($\text{var}(x)/\text{var}(y)$) can be provided with the `error.ratio` argument (default = 1, indicating equal measurement errors). The error ratio affects how pairwise slopes are weighted in the robust median calculation. When `error.ratio = 1`, all pairs receive equal weight. When `error.ratio != 1`, pairs are weighted to account for heterogeneous measurement precision.

Weighting:

Case weights can be provided via the `weights` argument. These are distinct from measurement error weighting (controlled by `error.ratio`). Case weights allow you to down-weight or up-weight specific observations in the analysis.

Standard Errors and Variance-Covariance Matrix:

The analytical (Passing & Bablok, 1983) confidence intervals do not provide a covariance between the intercept and slope, so a variance-covariance matrix (needed by `joint_test()` and `plot_joint()`) is only returned when one of the following is used:

- **Pairs bootstrap** (`se_method = "bootstrap"`, `replicates > 0`): whole (x, y) observations are resampled with replacement and the model is refit. The variance-covariance matrix is the covariance of the bootstrap estimates and the confidence intervals are percentile intervals.
- **Delete-d jackknife** (`se_method = "jackknife"`, `replicates > 0`): the model is refit on replicates random subsets that each leave out $d = \text{floor}(n/2)$ observations. The delete-1 jackknife is inconsistent for median-type estimators such as Passing-Bablok; deleting d observations with $\sqrt{n}/d \rightarrow 0$ restores consistency (Shao & Wu, 1989). Confidence intervals are Wald-type intervals using a $t(n-2)$ quantile.
- **Dufey (2020)** (`se_method = "dufey"`): an analytic, distribution-free estimator for the equivariant ("scissors") Passing-Bablok estimator, based on the U-statistic variance of Kendall's tau and a sandwich estimator for the intercept. The slope interval is formed from order statistics of the pairwise slopes and the intercept interval is Wald-type. **This option is experimental.** In simulations its marginal standard errors were accurate for $n \geq 20$, but joint tests of intercept and slope (`joint_test()`) rejected a true null hypothesis about 7-11% of the time at a nominal 5%, suggesting the intercept-slope covariance is not yet reliable. Prefer the pairs bootstrap or delete-d jackknife for joint tests.

Resampling is particularly useful for:

- Weighted regression (case weights or `error.ratio != 1`)
- Methods 'invariant' and 'scissors' (where analytical CI validity is uncertain)
- Joint tests of intercept and slope

The method automatically:

- Tests for high positive correlation using Kendall's tau
- Tests for linearity using a CUSUM test
- Computes confidence intervals (analytical or bootstrap)

Value

The function returns a `simple_eiv` object with the following components:

- `coefficients`: Named vector of coefficients (intercept and slope).
- `residuals`: Residuals from the fitted model.
- `fitted.values`: Predicted Y values.
- `model_table`: Data frame presenting the full results from the Passing-Bablok regression.
- `vcov`: Variance-covariance matrix for slope and intercept (if resampling or `se_method = "dufey"` is used; otherwise NULL).
- `df.residual`: Residual degrees of freedom.
- `call`: The matched call.
- `terms`: The terms object used.
- `model`: The model frame.
- `x_vals`: Original x values used in fitting.
- `y_vals`: Original y values used in fitting.
- `weights`: Case weights (if provided).
- `error_ratio`: Error ratio used in fitting.
- `conf.level`: Confidence level used.
- `method`: Character string describing the method.
- `method_num`: Numeric method identifier (1, 2, or 3).
- `kendall_test`: Results of Kendall's tau correlation test.
- `cusum_test`: Results of CUSUM linearity test.
- `n_slopes`: Number of slopes used in estimation.
- `boot`: Resampling results (if replicates > 0 and `se_method` is "bootstrap" or "jackknife").
- `se_method`: Method actually used for standard errors ("analytic", "bootstrap", "jackknife", or "dufey").

References

- Passing, H., & Bablok, W. (1983). A New Biometrical Procedure for Testing the Equality of Measurements from Two Different Analytical Methods. Application of linear regression procedures for method comparison studies in Clinical Chemistry, Part I. *Cclm*, 21(11), 709-720. doi: 10.1515/cclm.1983.21.11.709
- Passing, H., & Bablok, W. (1984). Comparison of Several Regression Procedures for Method Comparison Studies and Determination of Sample Sizes Application of linear regression procedures for method comparison studies in Clinical Chemistry, Part II. *Clinical Chemistry and Laboratory Medicine*, 22(6). doi: 10.1515/cclm.1984.22.6.431
- Bablok, W., Passing, H., Bender, R., & Schneider, B. (1988). A General Regression Procedure for Method Transformation. Application of Linear Regression Procedures for Method Comparison Studies in Clinical Chemistry, Part III. *Clinical Chemistry and Laboratory Medicine*, 26(11). doi: 10.1515/cclm.1988.26.11.783
- Dufey, F. (2020). Derivation of Passing-Bablok regression from Kendall's tau. *The International Journal of Biostatistics*, 16(2), 20190157. doi: 10.1515/ijb-2019-0157
- Sen, P. K. (1968). Estimates of the regression coefficient based on Kendall's tau. *Journal of the American Statistical Association*, 63(324), 1379-1389. doi: 10.1080/01621459.1968.10480934
- Shao, J., & Wu, C. F. J. (1989). A general theory for jackknife variance estimation. *The Annals of Statistics*, 17(3), 1176-1197. doi: 10.1214/aos/1176347263

Examples

```
## Not run:
# Basic Passing-Bablok regression (scissors method, default)
model <- pb_reg(method2 ~ method1, data = mydata)

# With known error ratio
model_er <- pb_reg(method2 ~ method1, data = mydata, error.ratio = 2)

# With replicate measurements
model_rep <- pb_reg(method2 ~ method1, data = mydata, id = "subject_id")

# With bootstrap confidence intervals
model_boot <- pb_reg(method2 ~ method1, data = mydata,
                     error.ratio = 1.5, replicates = 1000)

# Delete-d jackknife or analytic (Dufey 2020) variance-covariance matrix
model_jack <- pb_reg(method2 ~ method1, data = mydata,
                     se_method = "jackknife", replicates = 1000)
model_dufey <- pb_reg(method2 ~ method1, data = mydata, se_method = "dufey")
joint_test(model_dufey)

# Symmetric method
model_sym <- pb_reg(method2 ~ method1, data = mydata, method = "symmetric")

# Scale-invariant method
model_inv <- pb_reg(method2 ~ method1, data = mydata, method = "invariant")

# With case weights
model_wt <- pb_reg(method2 ~ method1, data = mydata,
                  weights = mydata$case_weights)

# View results
print(model)
summary(model)
plot(model)

## End(Not run)
```

power_agreement_exact *Power Calculation for Exact Agreement/Tolerance Test*

Description

[Maturing]

Computes sample size, power, or other parameters for the exact method of assessing agreement between two measurement methods, as described in Shieh (2019). This method tests whether the central portion of paired differences falls within specified bounds. This *roughly* equates to the power for tolerance limits.

Usage

```
power_agreement_exact(
  n = NULL,
  delta = NULL,
  mu = 0,
  sigma = NULL,
  p0_star = 0.95,
  power = NULL,
  alpha = 0.05,
  max_iter = 1000
)
```

Arguments

n	Number of subject pairs (sample size)
delta	Maximum allowable difference bound (half-width of tolerance interval)
mu	Mean of paired differences
sigma	Standard deviation of paired differences
p0_star	The coverage proportion (content) of the tolerance interval. Central proportion under null hypothesis (default = 0.95)
power	Target power (probability of rejecting false null)
alpha	Significance level (Type I error rate, default = 0.05, Confidence level = 1-alpha)
max_iter	Maximum iterations for gamma computation (default = 1000)

Details

This function implements the exact agreement test procedure of Shieh (2019) for method comparison studies. The test evaluates whether the central proportion of the distribution of paired differences lies within the interval $[-\delta, \delta]$.

The null hypothesis is: $H_0: \theta_{(1-p)} \leq -\delta$ or $\delta \leq \theta_p$ The alternative is: $H_1: -\delta < \theta_{(1-p)}$ and $\theta_p < \delta$

where $p = (1 + p0_star)/2$, and θ_p represents the 100p-th percentile of the paired differences.

Specify three of: n, delta, power, and sigma. The fourth will be calculated. If mu is not specified, it defaults to 0.

Tolerance Interval Interpretation:

The parameter p0_star represents the tolerance coverage proportion, i.e., the proportion of the population that must fall within the specified bounds $[-\delta, \delta]$ under the null hypothesis. This is conceptually related to tolerance intervals, but formulated as a hypothesis test rather than an estimation problem.

Note: This differs from Bland-Altman's "95% limits of agreement," which are confidence intervals for the 2.5th and 97.5th percentiles, *not* tolerance intervals.

Value

An object of class "power.htest", a list with components:

n	Sample size
delta	c
mu	Mean of differences
sigma	Standard deviation of differences
p0_star	Central proportion (null hypothesis)
p1_star	Central proportion (alternative hypothesis)
alpha	Significance level
power	Power of the test
critical_value	Critical value for test statistic
method	Description of the method
note	Additional notes

References

Shieh, G. (2019). Assessing Agreement Between Two Methods of Quantitative Measurements: Exact Test Procedure and Sample Size Calculation. *Statistics in Biopharmaceutical Research*, 12(3), 352-359. <https://doi.org/10.1080/19466315.2019.1677495>

Examples

```
# Example 1: Find required sample size
power_agreement_exact(delta = 7, mu = 0.5, sigma = 2.5,
                      p0_star = 0.95, power = 0.80, alpha = 0.05)

# Example 2: Calculate power for given sample size
power_agreement_exact(n = 15, delta = 0.1, mu = 0.011,
                      sigma = 0.044, p0_star = 0.80, alpha = 0.05)

# Example 3: Find required delta for given power and sample size
power_agreement_exact(n = 50, mu = 0, sigma = 2.5,
                      p0_star = 0.95, power = 0.90, alpha = 0.05)
```

powerCurve-methods	<i>Methods for powerCurve objects</i>
--------------------	---------------------------------------

Description

Methods defined for objects returned from the powerCurve function.

Usage

```
find_n(x, power = 0.8)

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

Arguments

x object of class `powerCurve`

power Level of power (value between 0 and 1) for `find_n` to find the sample size.

... further arguments passed through, see description of return value for details. [blandPowerCurve](#).

Value

`plot` Returns a plot of the limits of agreement (type = 1) or concordance plot (type = 2)

`find_n` Find sample size at which desired power is achieved

reli_stats

Reliability Statistics

Description**[Stable]**

The `reli_stats` and `reli_aov` functions produce reliability statistics described by Weir (2005). This includes intraclass correlation coefficients, the coefficient of variation, and the standard MSE of measurement.

Usage

```
reli_stats(
  measure,
  item,
  id,
  data,
  wide = FALSE,
  col.names = NULL,
  se_type = c("MSE", "ICC1", "ICC2", "ICC3", "ICC1k", "ICC2k", "ICC3k"),
  cv_calc = c("MSE", "residuals", "SEM"),
  conf.level = 0.95,
  other_ci = FALSE,
  type = c("chisq", "perc", "norm", "basic"),
  replicates = 1999
)

reli_aov(
```

```

    measure,
    item,
    id,
    data,
    wide = FALSE,
    col.names = NULL,
    se_type = c("MSE", "ICC1", "ICC2", "ICC3", "ICC1k", "ICC2k", "ICC3k"),
    cv_calc = c("MSE", "residuals", "SEM"),
    conf.level = 0.95,
    other_ci = FALSE,
    type = c("chisq", "perc", "norm", "basic"),
    replicates = 1999
  )

```

Arguments

<code>measure</code>	Name of column containing the measurement of interest.
<code>item</code>	Name of column containing the items. If this is a test-retest reliability study then this would indicate the time point (e.g., time1,time2, time3, etc.).
<code>id</code>	Column with subject identifier.
<code>data</code>	Data frame with all data.
<code>wide</code>	Logical value (TRUE or FALSE) indicating if data is in a "wide" format. Default is TRUE.
<code>col.names</code>	If wide is equal to TRUE then col.names is a list of the column names containing the measurements for reliability analysis.
<code>se_type</code>	Type of standard error calculation. The default is to use the mean square error (MSE). Otherwise, the total sums of squares and the ICC are utilized to estimate the SEM, SEE, and SEP.
<code>cv_calc</code>	Coefficient of variation (CV) calculation. This function allows for 3 versions of the CV. "MSE" is the default.
<code>conf.level</code>	the confidence level required. Default is 95%.
<code>other_ci</code>	Logical value (TRUE or FALSE) indicating whether to calculate confidence intervals for the CV, SEM, SEP, and SEE. Note: this will dramatically increase the computation time.
<code>type</code>	A character string representing the type of confidence intervals for the CV, SEM, SEP, and SEE. Only "norm", "basic", and "perc" currently supported for parametric bootstrap CI. An approximate method, chisq, is the default. See ?boot::boot.ci for more details on bootstrap methods.
<code>replicates</code>	The number of bootstrap replicates. Passed on to the boot function. Default is 1999.

Details

These functions return intraclass correlation coefficients and other measures of reliability (CV, SEM, SEE, and SEP). The estimates of variances for any of the measures are derived from linear

mixed models. When `other_ci` is set to `TRUE`, then a parametric bootstrap approach to calculating confidence intervals is used for the CV, SEM, SEE, and SEP.

`reli_stats` uses a linear mixed model to estimate variance components. In some cases there are convergence issues. When this occurs it is prudent to use `reli_aov` which instead utilizes sums of squares approach. The results may differ slightly between the functions. If `reli_aov` is used then rows with missing observations (e.g., if a participant has a missing observation) will be dropped.

The CV calculation has 3 versions. The "MSE" uses the "mean squared error" from the linear mixed model used to calculate the ICCs. The "SEM" option instead uses the SEM calculation and expresses CV as a ratio of the SEM to the overall mean. The "residuals" option uses the model residuals to calculate the root mean square error which is then divided by the grand mean.

The CV, SEM, SEE, and SEP values can have confidence intervals produced if the `other_ci` argument is set to `TRUE`. For the CV, the default method (`type = "chisq"`) is Vangal's modification of the McKay approximation. For the other measures, a simple chi-squared approximation is utilized (Hann & Meeker, 1991). All other methods are bootstrapping based methods (see `?boot::boot`). The `reli_stats` functions utilizes a parametric bootstrap while the `reli_aov` function utilizes an ordinary (non-parametric) bootstrap method.

Value

Returns single list with the results of the agreement analysis.

- `icc`: Table of ICC results
- `lmer`: Linear mixed model from `lme4`
- `anova`: Analysis of Variance table
- `var_comp`: Table of Variance Components
- `n.id`: Number of subjects/participants
- `n.items`: Number of items/time points
- `cv`: Coefficient of Variation
- `SEM`: List with Standard MSE of Measurement estimate (est)
- `SEE`: List with Standard MSE of the Estimate estimate (est)
- `SEP`: List with Standard MSE of Predictions (est)
- `call`: the matched call

References

- Weir, J. P. (2005). Quantifying test-retest reliability using the intraclass correlation coefficient and the SEM. *The Journal of Strength & Conditioning Research*, 19(1), 231-240.
- Shrout, P.E. and Fleiss, J.L. (1976). Intraclass correlations: uses in assessing rater reliability. *Psychological Bulletin*, 86, 420-428.
- McGraw, K. O. and Wong, S. P. (1996). Forming inferences about some intraclass correlation coefficients. *Psychological Methods*, 1, 30-46. See errata on page 390 of same volume.
- Hahn, G. J., & Meeker, W. Q. (2011). *Statistical intervals: a guide for practitioners* (Vol. 92). John Wiley & Sons. pp. 55-56.
- Vangel, M. G. (1996). Confidence intervals for a normal coefficient of variation. *The American Statistician*, 50(1), 21-26.

Examples

```
data('reps')
reli_stats(data = reps, wide = TRUE, col.names = c("x", "y"))
```

reps	<i>reps</i>
------	-------------

Description

A fake data set of a agreement study where both measures have replicates.
 The data set published in the original Bland & Altman paper on agreement.

Usage

```
reps
ba1986
```

Format

A data frame with 20 rows with 3 variables

id Subject identifier
x X measurement
y Y measurement

A data frame with 17 rows with 5 variables

id Subject identifier
wright1 PERF measurement #1 using Wright device
wright2 PERF measurement #2 using Wright device
mini1 PERF measurement #1 using Mini device
mini2 PERF measurement #2 using Mini device

References

Bland, J. M., & Altman, D. (1986). Statistical methods for assessing agreement between two methods of clinical measurement. The Lancet, 327(8476), 307-310.

Description

Methods defined for objects returned from the agree functions.

Usage

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

## S3 method for class 'simple_agree'
plot(
  x,
  type = 1,
  x_name = "x",
  y_name = "y",
  geom = c("geom_point", "geom_bin2d", "geom_density_2d", "geom_density_2d_filled",
    "stat_density_2d"),
  smooth_method = NULL,
  smooth_se = TRUE,
  ...
)

check(x, ...)
```

```
## S3 method for class 'simple_agree'
check(x, ...)
```

Arguments

x	object of class simple_agree as returned from a function starting with 'agree'
...	further arguments passed through, see description of return value for details. agree_test .
type	Type of plot to output. Default (1) is Bland-Altman plot while type=2 will produce a line-of-identity plot.
x_name	Name/label for x values (first measurement)
y_name	Name/label for y values (second measurement)
geom	String naming the type of geometry to display the data points. Default is "geom_point". Other options include: "geom_bin2d", "geom_density_2d", "geom_density_2d_filled", and "stat_density_2d".
smooth_method	Smoothing method (function) to use, accepts either NULL or a character vector, e.g. "lm", "glm", "gam", "loess" or a function. Default is NULL, which will not include a trend line.
smooth_se	Display confidence interval around smooth?

Value

`print` Prints short summary of the Limits of Agreement

`plot` Returns a plot of the limits of agreement (type = 1) or concordance plot (type = 2)

`check` Returns 2 plots, `p_norm` and `p_het`, testing the assumptions of a Bland-Altman analysis. P-values for the normality and heteroskedasticity tests are provided as captions to the plot.

simple_eiv-methods *Methods for simple_eiv objects*

Description

Methods defined for objects returned from error-in-variables models (e.g., `dem_reg`, `pb_reg`).

Usage

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

## S3 method for class 'simple_eiv'
formula(x, ...)

## S3 method for class 'simple_eiv'
model.frame(formula, data = NULL, na.action = na.pass, ...)

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

## S3 method for class 'simple_eiv'
confint(object, parm, level = NULL, ...)

## S3 method for class 'simple_eiv'
plot(
  x,
  x_name = NULL,
  y_name = NULL,
  interval = c("none", "confidence"),
  level = NULL,
  n_points = 100,
  data = NULL,
  ...
)

## S3 method for class 'simple_eiv'
check(x, data = NULL, ...)

plot_joint(x, ...)
```

```

## S3 method for class 'simple_eiv'
plot_joint(
  x,
  ideal_slope = 1,
  ideal_intercept = 0,
  show_intervals = TRUE,
  n_points = 100,
  test_method = c("F", "asymptotic"),
  ...
)

## S3 method for class 'simple_eiv'
vcov(object, ...)

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

## S3 method for class 'simple_eiv'
fitted(object, type = c("y", "x", "both"), data = NULL, ...)

## S3 method for class 'simple_eiv'
residuals(object, type = c("optimized", "x", "y", "raw_y"), data = NULL, ...)

## S3 method for class 'simple_eiv'
predict(
  object,
  newdata = NULL,
  interval = c("none", "confidence"),
  level = NULL,
  se.fit = FALSE,
  data = NULL,
  ...
)

```

Arguments

...	further arguments passed through.
formula	A simple_eiv object (for model.frame method)
data	Optional data frame. Required if model was fitted with model = FALSE and newdata is NULL.
na.action	Function for handling NA values (default: na.pass)
object, x	object of class simple_eiv from dem_reg or pb_reg function.
parm	A specification of which parameters are to be given confidence intervals, either a vector of numbers or a vector of names. If missing, all parameters are considered.
level	Confidence level for intervals (default uses the model's conf.level).

x_name	Name for x-axis label (optional).
y_name	Name for y-axis label (optional).
interval	Type of interval calculation. Can be "none" (default), or "confidence"
n_points	Number of points to use for drawing the joint confidence region (default = 100).
ideal_slope	The hypothesized slope value to test against (default = 1)
ideal_intercept	The hypothesized intercept value to test against (default = 0)
show_intervals	Logical. If TRUE, shows individual confidence intervals as well.
test_method	Reference distribution for the ellipse and test. "F" (default) uses the finite-sample F(2, n-2) distribution. "asymptotic" uses chi-squared(2).
type	Type of residuals to return. Options are "optimized" (default), "x", "y", or "raw_y".
newdata	An optional data frame containing values of X at which to predict. If omitted, the fitted values are returned.
se.fit	Logical. If TRUE, standard errors of predictions are returned. Note: For Passing-Bablok regression without bootstrap, standard errors are not available and this argument is ignored with a warning.

Value

print Prints short summary of the EIV regression model.
 summary Prints detailed summary.
 plot Returns a plot of the regression line and data.
 check Returns plots of residuals.
 plot_joint Returns plot of joint confidence region (Deming only).
 predict Predicts Y values for new X values.
 fitted Extracts fitted values.
 residuals Extracts residuals.
 coef Extracts model coefficients.
 vcov Extracts variance-covariance matrix.

simple_reli-methods *Methods for simple_reli objects*

Description

Methods defined for objects returned from the agree functions.

Usage

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

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

## S3 method for class 'simple_reli'
check(x, ...)
```

Arguments

x object of class `simple_reli` as returned from the `reli_stats` function

... further arguments passed through, see description of return value for details.
[reli_stats](#).

Value

print Prints short summary of the Limits of Agreement

plot Returns a plot of the data points used in the reliability analysis

temps

Data

Description

A dataset from a study on the reliability of human body temperature at different times of day before and after exercise.

Usage

```
temps

recpre_long
```

Format

A data frame with 60 rows and 10 variables:

id Subject identifier

trial_num order in which the experimental trial was completed

trial_condition Environmental condition and metabolic heat production

tod Time of Day

trec_pre Rectal temperature before the beginning of the trial

trec_post Rectal temperature at the end of the trial

trec_delta Change in rectal temperature

teso_pre Esophageal temperature before the beginning of the trial

teso_post Esophageal temperature at the end of the trial

teso_delta Change in esophageal temperature

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 30 rows and 6 columns.

Source

Ravanelli N, Jay O. The Change in Core Temperature and Sweating Response during Exercise Are Unaffected by Time of Day within the Wake Period. *Med Sci Sports Exerc.* 2020 Dec 1. doi: 10.1249/MSS.0000000000002575. Epub ahead of print. PMID: 33273272.

tolerance_delta-methods

Methods for tolerance_delta objects

Description

Methods defined for objects returned from the `tolerance_delta` function(s).

Usage

```
## S3 method for class 'tolerance_delta'
print(x, digits = 4, ...)

## S3 method for class 'tolerance_delta'
plot(
  x,
  geom = c("geom_point", "geom_bin2d", "geom_density_2d", "geom_density_2d_filled",
    "stat_density_2d"),
  delta = NULL,
  ...
)

## S3 method for class 'tolerance_delta'
check(x, ...)
```

Arguments

<code>x</code>	object of class <code>tolerance_delta</code> as returned from a <code>agreement_limit</code> function.
<code>digits</code>	The number of digits to print.
<code>...</code>	further arguments passed through, see description of return value for details. tolerance_limit .
<code>geom</code>	String naming the type of geometry to display the data points. Default is <code>"geom_point"</code> . Other options include: <code>"geom_bin2d"</code> , <code>"geom_density_2d"</code> , <code>"geom_density_2d_filled"</code> , and <code>"stat_density_2d"</code> .
<code>delta</code>	The maximal allowable difference.

Value

- `print` Prints short summary of the tolerance limits, with the SD of the differences (coefficient of variation when log-transformed). If the SD differs between rows of the limits (e.g., by condition), it is shown as a column instead. For clustered models, the between- and within-subject SDs are also printed.
- `plot` Returns a plot of the tolerance limits.
- `check` Returns plots testing the assumptions of the model. P-values for the normality and heteroskedasticity tests are provided as captions to the plot.

tolerance_limit	<i>Tolerance Limits from an Agreement Study</i>
-----------------	---

Description**[Maturing]**

A function for calculating tolerance limits for the difference between two measurements (difference = x-y). This is a procedure that should produce results similar to the Bland-Altman limits of agreement. See vignettes for more details.

Usage

```
tolerance_limit(
  data,
  x,
  y,
  id = NULL,
  condition = NULL,
  time = NULL,
  pred_level = 0.95,
  tol_level = 0.95,
  conf_level = 0.95,
  tol_method = c("analytic", "boot_cal"),
  bound_type = c("joint", "iu"),
  prop_bias = FALSE,
  log_tf = FALSE,
  log_tf_display = c("ratio", "sympercent"),
  model = c("gls", "lme"),
  cor_type = c("sym", "car1", "ar1", "none"),
  correlation = NULL,
  weights = NULL,
  keep_model = TRUE,
  replicates = 999
)
```

Arguments

data	A data frame containing the variables.
x	Name of the column for the first measurement.
y	Name of the column for the second measurement.
id	Name of the column for the subject ID. With <code>model = "lme"</code> , two column names can be given for nested random intercepts, outer level first (e.g., <code>c("golfer", "club")</code>) for clubs within golfers; see "Model assumptions" in details.
condition	Name of the column indicating different conditions subjects were tested under. This can be left missing if there are no differing conditions to be tested. Supplying a condition adds a separate residual variance for each condition (<code>nlme::varIdent</code>); see "Model assumptions" in details.
time	Name of the column indicating the time points. Only necessary if the data is from time series or repeated measures collection.
pred_level	Prediction level for the prediction interval, which is also the content (the proportion of differences to be covered, β) for the tolerance limits. Default is 95%.
tol_level	Confidence level (γ) for the tolerance limits. Default is 95%. See <code>bound_type</code> for what this confidence refers to.
conf_level	Confidence level for the confidence interval of the bias (<code>lower.CL/upper.CL</code>). Default is 95%. This does not affect the prediction or tolerance limits. For a two one-sided tests (TOST) procedure on the bias at level α , use <code>conf_level = 1 - 2 * alpha</code> (e.g., 0.90).
tol_method	Method for calculating the tolerance limits. Options are "analytic" (default) for closed-form limits and "boot_cal" (experimental) for the closed-form limits calibrated by a parametric bootstrap. The analytic limits are recommended; see details. Both target the interval set by <code>bound_type</code> . The previous names, "approx" and "perc", are deprecated aliases for "analytic" and "boot_cal".
bound_type	Which claim the tolerance limits (<code>lower.TL/upper.TL</code>) support. "joint" (default) gives a β -content, γ -confidence tolerance interval: with confidence <code>tol_level</code> , at least <code>pred_level</code> of all differences lie within the limits. "iu" gives equal-tailed bounds: a one-sided <code>tol_level</code> confidence bound on each of the $(1 - \text{pred_level})/2$ and $(1 + \text{pred_level})/2$ percentiles of the differences. The "iu" bounds are intended for an intersection-union test of agreement against a maximal allowable difference and are not a joint <code>tol_level</code> interval. See details.
prop_bias	Whether to include a proportional bias term in the model. Determines whether proportional bias should be considered for the prediction/tolerance limits calculations. Note that a slope of the differences on the average can appear without any true proportional bias when the two methods have unequal measurement error variances; see "Model assumptions" in details.
log_tf	Calculate limits of agreement using log-transformed data.
log_tf_display	The type of presentation for log-transformed results. The differences between methods can be displayed as a "ratio" or "sympercent".

model	The type of model for the differences. "gls" (default) is a marginal generalized least squares model (<code>nlme::gls()</code>) with the correlation structure set by <code>cor_type</code> . "lme" is a linear mixed model (<code>nlme::lme()</code>) with a random intercept for each id, which requires id; with two id columns it has nested random intercepts (outer level, and inner level within outer). With compound symmetry and no variance function both give the same fit; "lme" differs when serial correlation (<code>cor_type = "ar1"</code> or <code>"car1"</code>) or a variance function (condition or weights) is added, because it keeps a persistent subject effect and applies the variance function to the residuals only. See "Model assumptions" in details.
cor_type	The type of correlation structure. "sym" is for Compound Symmetry, "car1" is for continuous autocorrelation structure of order 1, or "ar1" for autocorrelation structure of order 1. The autoregressive options ("ar1" and "car1") have no persistent subject effect; see "Model assumptions" in details. With <code>model = "lme"</code> , "sym" fits the random intercept alone, "ar1" and "car1" add serial correlation of the residuals within id on top of the random intercept, and "none" is not allowed.
correlation	an optional <code>corStruct</code> object describing the within-group correlation structure that overrides the default setting. See the documentation of <code>corClasses</code> for a description of the available <code>corStruct</code> classes. If a grouping variable is to be used, it must be specified in the <code>form</code> argument to the <code>corStruct</code> constructor. Defaults to <code>NULL</code> . With <code>model = "lme"</code> , this is the correlation of the residuals within the innermost level of id (on top of the random intercepts) and must be grouped by it using the internal names: id for one level (e.g., <code>nlme::corAR1(form = ~ time id)</code>) or id/id_2 for two (e.g., <code>nlme::corAR1(form = ~ time id/id_2)</code>).
weights	an optional <code>varFunc</code> object or one-sided formula describing the within-group heteroskedasticity structure that overrides the default setting. If given as a formula, it is used as the argument to <code>varFixed</code> , corresponding to fixed variance weights. See the documentation on <code>varClasses</code> for a description of the available <code>varFunc</code> classes. Variance covariates must use the internal column names (<code>avg</code> , <code>condition</code> , <code>x</code> , <code>y</code> , <code>time</code>) or <code>fitted(.)</code> . Currently <code>varIdent</code> , <code>varFixed</code> , <code>varExp</code> , and <code>varPower</code> are supported. If the variance depends on <code>avg</code> , the limits are reported at the minimum, median, and maximum of <code>avg</code> even when <code>prop_bias = FALSE</code> . With <code>model = "lme"</code> , the variance function applies to the residuals only; the random-intercept variance is common to all observations.
keep_model	Logical indicator to retain the fitted model (<code>gls</code> or <code>lme</code>). Useful when working with large data and the model is very large.
replicates	The number of bootstrap replicates. Passed on to the <code>boot</code> function. Default is 999.

Details

The tolerance limits calculated in this function are based on the papers by Francq & Govaerts (2016), Francq, et al. (2019), and Francq, et al. (2020). The formulas, including how the degrees of freedom are determined for clustered data, are given in `vignette("agreement_analysis", package = "SimplyAgree")`.

The output contains three kinds of interval, which support different claims:

- **Prediction interval** (`lower.PL/upper.PL`): a beta-expectation tolerance interval. It is expected to contain a single future difference, from a new subject, with probability `pred_level`,

averaged over repeated studies. It is not a confidence statement: any single prediction interval may cover less than `pred_level` of the differences.

- **Tolerance limits**, `bound_type = "joint"` (`lower.TL/upper.TL`): a beta-content, gamma-confidence tolerance interval. With confidence `tol_level`, at least `pred_level` of all differences lie within the limits. If the limits lie within a maximal allowable difference of plus or minus `delta`, one can conclude (at level $1 - \text{tol_level}$) that at least `pred_level` of the differences lie within plus or minus `delta` (Lin, 2000). The tails do not need to be split equally.
- **Tolerance limits**, `bound_type = "iu"`: a one-sided `tol_level` bound on the lower and on the upper $(1 - \text{pred_level})/2$ percentile of the differences, i.e., on the limits of agreement. If both bounds lie within plus or minus `delta`, one can reject, at level $1 - \text{tol_level}$, that either limit of agreement lies outside plus or minus `delta` (intersection-union test). Both bounds hold together only about $1 - 2 * (1 - \text{tol_level})$ of the time, so they should not be reported as a joint `tol_level` interval. These target the same quantities as the default confidence bounds from `agreement_limit()` (numerically close but not identical, because the models differ).

The "joint" and "iu" limits test different hypotheses (coverage versus each tail), so they do not nest: the "iu" bounds can be wider than the "joint" limits, and a skewed distribution can pass one test and fail the other.

With `tol_method = "analytic"`:

- "joint" uses the approximation of Howe (1969) as described by Francq et al. (2020), with the standard error of prediction (SEP) in place of the sample standard deviation. For independent data this matches Howe's approximation exactly.
- "iu" uses the exact noncentral t bound for independent data, and otherwise the MOVER bound of Zou (2013).
- Both use a one-sided upper confidence bound for the residual standard deviation (SD). With independent data this is the usual chi-square bound. With compound symmetry (`cor_type = "sym"`), the between- and within-subject variance components are combined with the MOVER, as in `agreement_limit()` with `data_type = "nest"`. With other correlation structures, the bound uses an effective degrees of freedom (`SD.df`) from the approximate covariance of the variance parameters.

The analytic limits are exact for independent data (without a variance function). In simulations (95% target), they were close to nominal with compound symmetry (about 0.95, with 10 to 20 subjects) and with AR(1) correlation (about 0.95). With condition (a separate residual variance per condition) combined with compound symmetry, the variance components for each condition are bounded using that condition's cluster sizes and degrees of freedom; coverage was then about 0.96. Other variance functions (weights) use the cluster sizes of the whole data set and have not been checked by simulation.

With `tol_method = "boot_cal"`, the analytic limits are calibrated by a parametric bootstrap (Loh, 1987; Beran, 1987). New data are simulated from the fitted model (Francq et al., 2019), the model is refit to each replicate, and the analytic limits are computed for each replicate over a range of nominal confidence levels. The level is chosen at which the replicates achieve the target confidence for the fitted model:

- "joint": a proportion `tol_level` of the replicate intervals contain at least `pred_level` of the fitted distribution of the differences.

- "iu": separately for each bound, a proportion `tol_level` of the replicate bounds lie beyond the fitted limit of agreement.

The limits for the observed data are the analytic limits at the calibrated levels, which are returned as `lower.TL.level` and `upper.TL.level`. `tol_method = "boot_cal"` is **experimental**. The calibration is judged against the fitted model, so it treats the estimated correlation as known. With few subjects this adds its own error: in simulations with 10 to 15 subjects (95% target), coverage ranged from about 0.93 (compound symmetry, with or without condition) to about 0.96 (AR(1) correlation), while the analytic limits were 0.95 to 0.965 in the same settings. The analytic limits are therefore recommended; the bootstrap may be useful as a check for models whose analytic limits have not been evaluated (e.g., other variance functions or `prop_bias = TRUE`).

For clustered data (`id` supplied), all limits refer to a new difference from a new subject: the SD includes both the between- and within-subject variance.

Model assumptions:

With `model = "gls"` (default), the model is a marginal (generalized least squares) model fit with `nlme::gls()`. For the default setup (`cor_type = "sym"` without a variance function), compound symmetry with a non-negative correlation gives the same likelihood as a random-intercept model, and targets the marginal distribution of a single difference from a randomly chosen subject. With `model = "lme"`, the model is a linear mixed model fit with `nlme::lme()` with a random intercept for each `id`; with compound symmetry and no variance function it gives the same fit and limits. With `model = "lme"`, the degrees of freedom for the bias are the containment degrees of freedom (subjects - 1 for the models fit here), because the Satterthwaite approximation for `lme` models in `emmeans` can fail. The other options carry assumptions that should be checked:

- **Autoregressive correlation** (`cor_type = "ar1"` or `"car1"`): with `model = "gls"`, the correlation between two measurements from the same subject decays towards zero as they get further apart in time. A persistent subject-specific bias (e.g., a subject by method interaction) instead makes all measurements from that subject equally correlated. When such an effect exists, the autoregressive structures miss most of the long-range correlation, so the standard error of the bias is too small and the limits are too narrow; the bootstrap does not correct this, because it simulates from the same model. In one simulation (20 subjects with 19 measurements each, and a random subject effect), the 95% confidence interval for the bias covered the true value 74% of the time with AR(1), versus 96% with compound symmetry. With `model = "lme"`, the serial correlation is added to the residuals on top of a random intercept, which keeps the persistent subject effect. In simulations with a random subject effect and AR(1) residuals (15 subjects, 8 measurements each; 95% target), coverage of the joint tolerance limits was about 0.89 with `model = "gls"`, `cor_type = "ar1"` and about 0.95 with `model = "lme"`, `cor_type = "ar1"`. Use the autoregressive options with `model = "gls"` only when no persistent subject effect is expected. The fits can be compared with `AIC()` on the returned models (a REML comparison is valid because the fixed effects are the same).
- **Variance functions with compound symmetry** (`condition`, or `weights` together with `cor_type = "sym"`): in `gls`, the correlation applies to the standardized residuals, so the covariance between two measurements from the same subject is $\rho * \sigma_i * \sigma_j$. The between-subject variance therefore scales with the variance function (e.g., it is larger in a condition with a larger residual SD), rather than being common to all conditions. The marginal variance of each condition, which drives the limits, is not directly affected, but the standard error of each condition's bias and the reported variance components are. With `model = "lme"`, the variance function applies to the residuals only, and the between-subject variance is common

to all conditions. In simulations where that was true (12 subjects; 95% target), the joint tolerance limits had coverage of about 0.96 with `model = "gls"` and about 0.97 with `model = "lme"`.

- **Clustering levels:** `model = "gls"` has a single grouping factor (`id`). With `model = "lme"`, `id` can name two nested levels (e.g., `id = c("golfer", "club")`) for shots with several clubs per golfer), giving random intercepts for the outer level and for the inner level within the outer level. Setting `id` to the inner level alone (e.g., a golfer-by-club identifier) should be avoided: it drops the correlation across inner levels within the same outer level, which understates the uncertainty. In simulations (15 subjects with 2 to 4 settings each; 95% target), the joint tolerance limits had coverage of about 0.88 to 0.91 with the inner level as `id`, versus about 0.96 with nested random intercepts. Grouping by the outer level only (`id = "golfer"`) also gave close to nominal coverage (about 0.96), because the inner-level variance is then absorbed into the residual; the nested model additionally reports the inner-level variance component (`SD.nested`). More than two levels are not supported.
- **Variance components:** the SD of a single difference splits into between-subject (`SD.between`), inner-level (`SD.nested`, with two `id` columns), and within-subject (`SD.within`) components. With `model = "gls"` and compound symmetry these are $\sqrt{\rho} * \text{SD}$ and $\sqrt{1 - \rho} * \text{SD}$; with `model = "lme"`, `SD.between` is the random-intercept SD and `SD.within` the residual SD. They can be used, for example, for the point estimate of limits for the mean of m measurements per subject, $\text{bias} \pm z * \sqrt{\text{SD.between}^2 + \text{SD.within}^2 / m}$ (without residual correlation), or for the between-subject floor of agreement. They are point estimates only.
- **Proportional bias** (`prop_bias = TRUE`): a non-zero slope of the differences on the average can appear without any true proportional bias. Whenever the two methods have unequal measurement error variances, $\text{cov}(\text{difference}, \text{average}) = (\text{var}(x) - \text{var}(y)) / 2$ is not zero (Bland & Altman, 1999), so the slope should not be read as proportional bias on its own. Errors-in-variables methods such as `dem_reg()` or `pb_reg()` are better suited to assessing proportional bias.

Value

Returns single `tolerance_delta` class object with the results of the agreement analysis with a prediction interval and tolerance limits.

- **limits:** A data frame containing the prediction/tolerance limits. Columns include `bias` (estimated mean difference), `SEM` (standard error of the bias), `SD` (residual standard deviation of a single difference at that row, from the variance function if one is in the model), `SEP` (standard error of prediction, $\sqrt{\text{SD}^2 + \text{SEM}^2}$), `SD.df` (degrees of freedom of the residual variance), `SD.upper` (one-sided `tol_level` upper confidence bound for SD), `SD.between/SD.nested/SD.within` (between-subject, nested (inner level), and within-subject components of SD, under compound symmetry or with `model = "lme"`; NA otherwise, and `SD.nested` is NA without nesting), `lower.CL/upper.CL` (confidence limits for the bias), `lower.PL/upper.PL` (prediction limits), `lower.TL/upper.TL` (tolerance limits), and, for `tol_method = "boot_cal"`, `lower.TL.level/upper.TL.level` (the calibrated nominal levels at which the analytic limits were computed).
- **model:** The fitted `gls` or `lme` model; NULL if `keep_model` set to FALSE. The data (with the internal column names `x`, `y`, `delta`, `avg`, `id` (the row number if not supplied), and, if supplied, `condition` and `time`), correlation structure, and variance function are stored with the model, so `update()` and `nlme::getData()` can be used on it directly.
- **call:** The matched call.

References

- Francq, B. G., & Govaerts, B. (2016). How to regress and predict in a Bland–Altman plot? Review and contribution based on tolerance intervals and correlated-errors-in-variables models. *Statistics in mMdicine*, 35(14), 2328-2358.
- Francq, B. G., Lin, D., & Hoyer, W. (2019). Confidence, prediction, and tolerance in linear mixed models. *Statistics in Medicine*, 38(30), 5603-5622.
- Francq, B. G., Berger, M., & Boachie, C. (2020). To tolerate or to agree: A tutorial on tolerance intervals in method comparison studies with BivRegBLS R Package. *Statistics in Medicine*, 39(28), 4334-4349.
- Howe, W. G. (1969). Two-sided tolerance limits for normal populations—some improvements. *Journal of the American Statistical Association*, 64(326), 610-620.
- Lin, L. I. (2000). Total deviation index for measuring individual agreement with applications in laboratory performance and bioequivalence. *Statistics in Medicine*, 19(2), 255-270.
- Loh, W. Y. (1987). Calibrating confidence coefficients. *Journal of the American Statistical Association*, 82(397), 155-162.
- Beran, R. (1987). Prepivoting to reduce level error of confidence sets. *Biometrika*, 74(3), 457-468.
- Zou, G. Y. (2013). Confidence interval estimation for the Bland–Altman limits of agreement with multiple observations per individual. *Statistical Methods in Medical Research*, 22(6), 630-642.

Examples

```
data('reps')

# Simple
tolerance_limit(x = "x", y = "y", data = reps)

# Nested
tolerance_limit(x = "x", y = "y", data = reps, id = "id")

# Nested, random intercept (mixed) model
tolerance_limit(x = "x", y = "y", data = reps, id = "id", model = "lme")

# Nested random intercepts (e.g., measurements within settings within subjects)

reps2 = reps
reps2$setting = rep(1:2, length.out = nrow(reps2))
tolerance_limit(x = "x", y = "y", data = reps2, id = c("id", "setting"),
               model = "lme")
```

Index

* datasets

- reps, [51](#)
- temps, [56](#)

agree_assurance, [3](#)
agree_assurance(), [8](#), [9](#)
agree_coef, [6](#)
agree_expected_half, [7](#)
agree_expected_half(), [5](#)
agree_nest, [10](#)
agree_np, [12](#)
agree_reps, [13](#)
agree_test, [15](#), [52](#)
agreement_limit, [16](#), [35](#)
agreement_limit(), [61](#)
as.data.frame, [30–33](#)

ba1986 (reps), [51](#)
blandPowerCurve, [19](#), [48](#)

ccc_test, [20](#)
check (simple_agree-methods), [52](#)
check.loa (loa-methods), [35](#)
check.loa_mermod (loa_mermod-methods),
[37](#)
check.simple_eiv (simple_eiv-methods),
[53](#)
check.simple_reli
(simple_reli-methods), [55](#)
check.tolerance_delta
(tolerance_delta-methods), [57](#)
coef.simple_eiv (simple_eiv-methods), [53](#)
confint.simple_eiv
(simple_eiv-methods), [53](#)

dem_reg, [22](#)
dem_reg(), [17](#), [63](#)
deming_power, [25](#)
deming_power_sim (deming_power), [25](#)
deming_sample_size, [27](#)

find_n (powerCurve-methods), [47](#)
fitted.simple_eiv (simple_eiv-methods),
[53](#)
formula.simple_eiv
(simple_eiv-methods), [53](#)

jmvagree, [29](#)
jmvagreemulti, [30](#)
jmvdeming, [31](#)
jmvreli, [33](#)
joint_test, [33](#)
joint_test(), [43](#)

loa-methods, [35](#)
loa_lme, [36](#)
loa_mermod-methods, [37](#)
loa_mixed, [38](#), [38](#), [40](#)
loa_mixed_bs-methods, [40](#)

model.frame.simple_eiv
(simple_eiv-methods), [53](#)

nlme::gls(), [60](#), [62](#)
nlme::lme(), [60](#), [62](#)

pb_reg, [41](#)
pb_reg(), [17](#), [63](#)
plot.loa (loa-methods), [35](#)
plot.loa_mermod (loa_mermod-methods), [37](#)
plot.loa_mixed_bs
(loa_mixed_bs-methods), [40](#)
plot.powerCurve (powerCurve-methods), [47](#)
plot.simple_agree
(simple_agree-methods), [52](#)
plot.simple_eiv (simple_eiv-methods), [53](#)
plot.simple_reli (simple_reli-methods),
[55](#)
plot.tolerance_delta
(tolerance_delta-methods), [57](#)
plot_joint (simple_eiv-methods), [53](#)
plot_joint(), [43](#)

power_agreement_exact, 45
power_agreement_exact(), 5, 9
powerCurve-methods, 47
predict.simple_eiv
 (simple_eiv-methods), 53
print.loa (loa-methods), 35
print.loa_mermod (loa_mermod-methods),
 37
print.loa_mixed_bs
 (loa_mixed_bs-methods), 40
print.simple_agree
 (simple_agree-methods), 52
print.simple_eiv (simple_eiv-methods),
 53
print.simple_reli
 (simple_reli-methods), 55
print.tolerance_delta
 (tolerance_delta-methods), 57

recpre_long (temps), 56
reli_aov (reli_stats), 48
reli_stats, 48, 56
reps, 51
residuals.simple_eiv
 (simple_eiv-methods), 53

simple_agree-methods, 52
simple_eiv-methods, 53
simple_reli-methods, 55
summary.simple_eiv
 (simple_eiv-methods), 53

temps, 56
tolerance_delta-methods, 57
tolerance_limit, 57, 58

vcov.simple_eiv (simple_eiv-methods), 53