Package: MSBMisc (via r-universe)

September 4, 2024

Type Package

Title Some functions I wrote that I find useful

Version 0.0.1.14

Author person(``Mattan S.", ``Ben-Shachar")

Maintainer Mattan S. Ben-Shachar <matanshm@post.bgu.ac.il>

Description misc. functions.

License GPL-3

URL https://mattansb.github.io/MSBMisc/

BugReports https://github.com/mattansb/MSBMisc/issues/

Encoding UTF-8

Depends R (>= 4.1)

Imports stats, utils

Suggests afex, clipr, dplyr, correlation, effectsize, emmeans, ggeffects, ggplot2, ggtrace, GPArotation, insight, lavaan, lme4, lubridate, marginaleffects, MASS, patchwork, psych, qqplotr, rstudioapi, semTools, shiny, stringr, tidySEM, testthat (>= 3.0.0), knitr

Remotes yjunechoe/ggtrace

LazyData true

RoxygenNote 7.2.3

Config/testthat/edition 3

Roxygen list(markdown = TRUE)

VignetteBuilder knitr

Repository https://mattansb.r-universe.dev

RemoteUrl https://github.com/mattansb/MSBMisc

RemoteRef HEAD

RemoteSha bd9fd05bc6f749f42c47648f633c3c9cf22b6487

Contents

`age_in_unit` *Age in some units*

Description

Age in some units

Usage

```
age_in_unit(
  DOB,
  REFDATE = Sys.Date(),
  years = TRUE,
  months = TRUE,
  weeks = TRUE,
  days = TRUE
)
```

Arguments

DOB, REFDATE	Two dates
years, months, weeks, days	the units.

Examples

```
DOB <- as.Date("1989-08-05")
TODAY <- Sys.Date()
age_in_unit(DOB, TODAY)
```

---

bracketifyFile *bracketify*

Description

Adpated from stla/bracketify

Usage

```
bracketifyFile()

bracketifySelection()
```

---

chisq_pairwise *Follow-up for contingency table test*

Description

Follow-up for contingency table test

Usage

```
chisq_pairwise(
  Xsq,
  population_in_row = TRUE,
  adjust = stats::p.adjust.methods,
  effect_size = c("V", "phi"),
  ci = 0.95,
  ...
)

chisq_residual(
  Xsq,
```

```
  adjust = stats::p.adjust.methods,
  res_type = c("pearson", "standardized"),
  ci = 0.95
)
```

Arguments

Xsq	Result from chisq.test()
population_in_row	Comparisons by row? (If not, by column.)
adjust	Method for correcting p-values. See stats::p.adjust.
effect_size	Type of effect size to use.
ci	Confidence Interval (CI) level
...	Passed to chisq.test().
res_type	Type of residuals to use.

Examples

```
M <- as.table(rbind(
  c(762, 327, 468),
  c(484, 239, 477)
))
dimnames(M) <- list(
  gender = c("F", "M"),
  party = c("Democrat", "Independent", "Republican")
)
M

res <- chisq.test(M)
chisq_pairwise(res)
chisq_pairwise(res, population_in_row = FALSE)

chisq_residual(res)
```

---

compare_cor *Compare correlations*

Description

This function is a wrapper around psych::r.test.

Usage

```
compare_cor(data, r1, r2, data2 = NULL, by = NULL, ci = 0.95)
```

Arguments

data	A data frame
r1, r2	Names of variable for the first and second correlations
data2	Where to look for the r2 columns (if not provided, looked for on data).
by	Name of column to split data by. The column must have only 2 unique values. If provided, the correlation between r1 is compared between the two groups.
ci	Confidence level for correlations.

Details

- If data2 is provided, the correlation between r1 (in data) and r2 (in data2) is compared.
  - If r2 not provided, the correlation between r1 (in data) and r1 (in data2) is compared
- If by is provided r1 (in data) is compared between the 2 groups.
- Else, a test for the difference of two dependent correlations is conducted.

Value

A list of two data frames:

1. The two correlations + their CIs
2. The test results

Examples

```
# Test dependent correlations ------------------
## different variables
compare_cor(mtcars, r1 = c("mpg", "hp"), r2 = c("drat", "am"))

## 1 shared variable
compare_cor(mtcars, r1 = c("mpg", "hp"), r2 = c("mpg", "am"))



# Test independent correlations -----------------
## Different data sets
compare_cor(
  data = mtcars, r1 = c("mpg", "hp"),
  data2 = iris, r2 = c("Sepal.Length", "Sepal.Width")
)

## Groups
compare_cor(mtcars, r1 = c("mpg", "hp"), by = "am")
```

compare_freqs *Compare the frequencies of levels of a factor*

Description

Using `stats::mcnemar.test` for comparing dependent proportions.

This is function is dubious. Best not to use it.

Usage

```
compare_freqs(f, adjust = stats::p.adjust.methods, correct = TRUE)
```

Arguments

f	A factor vector.
adjust	Method for correcting p-values. See `stats::p.adjust`.
correct	a logical indicating whether to apply continuity correction when computing the test statistic.

Examples

```
f <- c(
  rep("A", 12),
  rep("B", 45),
  rep("C", 42),
  rep("D", 20)
)

compare_freqs(f)
```

contrast_weights *Build Contrast Weights*

Description

To be used ideally with emmeans::contrast. Each contrasts is tested (sum to 0?) and scaled so that all positive weights sum to 1 (and all negative weights to -1).

Usage

```
contrast_weights(..., .name = "custom", .adjust = NULL)

cw(..., .name = "custom", .adjust = NULL)
```

Arguments

...	Can be: • Unnamed scalars. • (Possibly named) vectors if equal length
.name	The label as it will appear in the results in emmeans.
.adjust	Gives the default adjustment method for multiplicity (used in emmeans).

Value

Depending on input, either a vector or a data frame of scaled weights.

Examples

```
data(mtcars)

mod <- lm(mpg ~ factor(cyl) * am, mtcars)


my_contrasts <- data.frame(
  "squares" = c(-1, 2, -1),
  "4 vs 6" = c(-30, 30, 0),
  check.names = FALSE
)

(my_contrasts2 <- cw(my_contrasts))
my_contrasts3 <- cw(my_contrasts, .adjust = "fdr")


library(emmeans)
(emms <- emmeans(mod, ~ cyl + am))

contrast(emms, method = my_contrasts, by = "am")
contrast(emms, method = my_contrasts2, by = "am") # estimate is affected!
contrast(emms, method = my_contrasts3, by = "am") # p value is affected

# Also in interaction contrasts
contrast(emms, interaction = list(cyl = my_contrasts2, am = "pairwise"))
```

---

crop_coord_polar *Crop* coord_polar

Description

Crop coord_polar

Usage

```
crop_coord_polar(
  plot,
  start = 0,
  end = 2 * pi,
  padding = 0.02,
  fix_aspect.ratio = TRUE
)
```

Arguments

plot	A ggplot
start, end	The angular values (in radians) marking where the plot starts and ends.
padding	How much padding to add around the crop?
fix_aspect.ratio	Should the aspect ratio be fixed?

Details

This function uses {ggtrace} by June Choe, adapted from this twitter thread.

Examples

```
library(ggplot2)

polar_plot <- ggplot(mtcars, aes(hp, mpg)) +
  geom_point() +
  geom_smooth(method = "lm") +
  expand_limits(y = c(0, 60)) +
  coord_polar(start = 0, theta = "y")

crop_coord_polar(polar_plot, end = pi)
crop_coord_polar(polar_plot, end = pi / 2)
crop_coord_polar(polar_plot, start = 3 * pi / 2, end = pi / 2)



# Also works with facets!
d <- data.frame(
  x = seq(1, 7, length = 6 * 5),
  y = rnorm(6 * 5),
  g = rep(letters[1:6], each = 5)
)

polar_plot_facet <- ggplot(d, aes(x, y)) +
  geom_point(aes(color = x), size = 2) +
  facet_wrap(~g) +
  scale_x_continuous(breaks = seq(0, 6), minor_breaks = NULL) +
  coord_polar()
```

```
crop_coord_polar(polar_plot_facet, start = pi)

# Use multiple values - one for each facet:
start <- seq(0, 5) * 2 * pi / 6
end <- start + start[2]

crop_coord_polar(polar_plot_facet,
  start = start, end = end
)
```

---

dbind *Bind matrices diagonally*

Description

Bind matrices diagonally

Usage

```
dbind(..., .fill = NULL)
```

Arguments

...	Matrices.
.fill	Value to fill the off-"diagonal" values. If NULL, the value is the default value of the inputs' mode.

Examples

```
M1 <- matrix(1:8, 2, 4)
M2 <- matrix(9:14, 2, 3)
dbind(M1, M2)
dbind(M1, M2, .fill = NA)


M1 <- matrix(letters[1:4], 2, 2)
M2 <- matrix(LETTERS[5:10], 3, 2)
dbind(M1, M2)
dbind(M1, M2, .fill = "Banana")


M1 <- matrix(TRUE, 2, 3)
M2 <- matrix(NA, 3, 4)
dbind(M1, M2)
```

delta_method *Transform means a (co)variances using the* delta *method*

Description

Transform means a (co)variances using the *delta* method

Usage

```
delta_method(..., .means, .V, return = c("means", "cov", "stddev", "cor"))
```

Arguments

`...`	Unquoted transformations. See example.
`.means`	A named vector of means.
`.V`	A covariance matrix.
`return`	What should be returned?

Value

A list with one of (optionally named):

- `means` of the transformed variables.
- `cov` (co) variance matrix of the the transformed variables.
- `stddev` standard deviations of the transformed variables (`sqrt(diag(cov))`).
- `cor` correlation matrix of the transformed variables. (`cov2cor(cov)`)

Note

Most of this function is mostly copied from `msm::deltamethod()`.

Examples

```
M <- sapply(mtcars, mean)
V <- cov(mtcars)

delta_method(
  (mpg^2) / hp,
  log_am = log1p(am),
  .means = M, .V = V,
  return = "cor"
)

# Sobel Test ----

data("mtcars")
mod.y <- lm(mpg ~ hp + cyl, mtcars[1:5, ])
```

```
mod.m <- lm(hp ~ cyl, mtcars[1:5, ])

bhat <- c(coef(mod.y), coef(mod.m))[c(2, 5)]
Vhat <- dbind(vcov(mod.y), vcov(mod.m))[c(2, 5), c(2, 5)]

res <- delta_method(
  hp * cyl,
  .means = bhat, .V = Vhat,
  return = c("means", "stddev")
)

res$means / res$stddev

# Compare:
(bhat[1] * bhat[2]) /
  sqrt(bhat[1]^2 * Vhat[2, 2] + bhat[2]^2 * Vhat[1, 1])

# Special character will give you a bad time...
m <- lm(mpg ~ factor(cyl), mtcars[1:5, ])

bhat <- coef(m)
names(bhat) <- c("cyl4", "cyl6", "cyl8")
V <- vcov(m)

delta_method(cyl4, cyl4 + cyl6, cyl4 + cyl8,
  .means = bhat,
  .V = V
)
```

`fa_reliability` *Calculate reliability from E/CFA*

Description

Calculate reliability from E/CFA

Usage

```
fa_reliability(x, ...)

## S3 method for class 'fa'
fa_reliability(x, keys = NULL, threshold = 0, labels = NULL, ...)

## S3 method for class 'lavaan'
fa_reliability(x, ...)
```

Arguments

x	E/CFA model (e.g., the result from psych::fa() or lavaan::cfa()).
...	For lavaan objects, arguments passed to semTools::compRelSEM()
keys	optional, see ?psych::make.keys
threshold	which values from the loadings should be used? Only used if keys = NULL.
labels	optional factor labels

Author(s)

Brenton M. Wiernik

Examples

```
data("Harman74.cor")
EFA <- psych::fa(Harman74.cor$cov, 4)

fa_reliability(EFA)


HS.model <- " visual  =~ x1 + x2 + x3
              textual =~ x4 + x5 + x6
              speed   =~ x7 + x8 + x9 "

CFA <- lavaan::cfa(HS.model, data = lavaan::HolzingerSwineford1939)

fa_reliability(CFA)
```

---

get_data_for_grid *Get raw data for plotting with model predictions*

Description

Get raw data for plotting with model predictions

Usage

```
get_data_for_grid(grid, model, residualize = FALSE, collapse_by = FALSE, ...)

## S3 method for class 'data.frame'
get_data_for_grid(
  grid,
  model,
  residualize = FALSE,
  collapse_by = FALSE,
  pred_name,
  ...
```

```
)

## S3 method for class 'ggeffects'
get_data_for_grid(
  grid,
  model,
  residualize = FALSE,
  collapse_by = FALSE,
  protect_names = TRUE,
  ...
)

## S3 method for class 'emmGrid'
get_data_for_grid(
  grid,
  model,
  residualize = FALSE,
  collapse_by = FALSE,
  protect_names = TRUE,
  ...
)

## S3 method for class 'predictions'
get_data_for_grid(grid, model, residualize = FALSE, collapse_by = FALSE, ...)
```

Arguments

grid	A data grid with predictions
model	The statistical model
residualize	Should data be residualized?
collapse_by	Name of grouping variable to collaple across. If TRUE name of grouping variable is automatically detected from the model.
...	Args passed from / to other functions.
pred_name	Name of column that has the predictions in the data grid
protect_names	Logical, if TRUE, preserves column names from the ggeffects object.

Examples

```
data("mtcars")
mtcars <- mtcars |> transform(cyl = factor(cyl))
mod <- lm(mpg ~ hp + cyl, data = mtcars[1:10, ])

nd <- expand.grid(
  hp = seq(50, 350, by = 50),
  cyl = "4"
)
```

```
nd$predicted_mpg <- predict(mod, newdata = nd)

get_data_for_grid(nd, mod)

get_data_for_grid(nd, mod, residualize = TRUE, pred_name = "predicted_mpg")



library(ggplot2)
ggplot(nd, aes(hp, predicted_mpg)) +
  geom_line() +
  geom_point(aes(y = mpg, color = "Raw"),
    data = get_data_for_grid(nd, mod)
  ) +
  geom_point(aes(color = "Residualized"),
    data = get_data_for_grid(nd, mod, residualize = TRUE, pred_name = "predicted_mpg")
  ) +
  labs(
    title = "Partial residual plot",
    color = "Data"
  )

## Support of data-grid packages ------
# - ggeffects
# - emmeans
# - marginaleffects


pred_ggeffects <- ggeffects::ggpredict(mod, c("hp [50:350, by = 50]", "cyl [4]"))
get_data_for_grid(pred_ggeffects, residualize = TRUE)


at <- list(hp = seq(50, 350, by = 50), cyl = "4")
pred_emmeans <- emmeans::emmeans(mod, ~ hp + cyl, at = at)
get_data_for_grid(pred_emmeans, mod, residualize = TRUE)


# pred_marginaleffects <- marginaleffects::predictions(mod, newdata = nd)
# get_data_for_grid(pred_marginaleffects, residualize = TRUE)


## Collapes across group ------
fm1 <- lme4::lmer(angle ~ temperature + (1 | recipe),
  data = cake
)

pred_ggeffects <- ggeffects::ggpredict(fm1, c("temperature", "recipe"))
nd <- marginaleffects::datagrid(
  temperature = unique(cake$temperature),
  model = fm1
)
pred_marginaleffects <- marginaleffects::predictions(fm1, newdata = nd)
```

```
get_data_for_grid(pred_marginaleffects, collapse_by = TRUE)
# get_data_for_grid(pred_marginaleffects, collapse_by = TRUE, residualize = TRUE)
```

---

gt *Compare multiple vectors*

Description

Compare multiple vectors

Usage

```
gt(...)

lt(...)

eq(...)

neq(...)

leq(...)

geq(...)

x %>>% y

x %<<% y

x %==% y

x %!=% y

x %<=% y

x %>=% y
```

Arguments

x, y, ...	Vectors, typically numerical, to be compared.

Value

A logical vector. For the operator, a `last_y` attribute stores the last RHS values from the comparisons (strip away with `as.vector()`). See examples.

Examples

```
x <- c(1, 3, 1, 1, 2)
y <- c(2, 2, 1, 1, 1)
z <- c(3, 1, 1, 2, 1)


lt(x, y, z) # >
eq(x, y, z) # ==
neq(x, y, z) # !=
leq(x, y, z) # <=
geq(x, y, z) # >=

gt(x, y, z) # <

# same as
x %>>% y %>>% z

# same as
x > y & y > z

# Operators can be mixed!

x %>>% y %==% z

# Or broken
(l1 <- x %>>% y)

(l2 <- l1 %==% z)

# same as
x > y & y == z

as.vector(l2)
```

---

has_any_data *Test rowwise if each row is missing / has all data in select columns*

Description

Test rowwise if each row is missing / has all data in select columns

Usage

```
has_any_data(.data, ..., .name)

has_all_data(.data, ..., .name)

missing_any_data(.data, ..., .name)
```

```
missing_all_data(.data, ..., .name)
```

Arguments

.data	A data frame, data frame extension (e.g. a tibble), or a lazy data frame (e.g. from dbplyr or dtplyr). See *Methods*, below, for more details.
...	<tidy-select> One or more unquoted expressions separated by commas. Variable names can be used as if they were positions in the data frame, so expressions like x:y can be used to select a range of variables.
.name	Name of the new column with the logical index.

Examples

```
data(mtcars)

mtcars[1, 1] <- NA
mtcars[2, ] <- NA

mtcars[1:3, 1:3] |>
  has_any_data(mpg:disp, .name = "has_any") |>
  has_all_data(mpg:disp, .name = "has_all") |>
  missing_any_data(mpg:disp, .name = "missing_any") |>
  missing_all_data(mpg:disp, .name = "missing_all")
```

---

is.TRUE *It's just logical*

Description

It's just logical

Usage

```
is.TRUE(x)

is.FALSE(x)

allTRUE(...)

anyTRUE(...)
```

Arguments

x, ...	Values to be tested.

Examples

```
x <- list(1, TRUE, list(TRUE), FALSE, "Hello world!")

is.TRUE(x)

is.FALSE(x)

allTRUE(TRUE, FALSE, stop("NOOOO"))

anyTRUE(TRUE, FALSE, stop("NOOOO"))
```

`lnorm_mean` *Functions to convert parameters of a log-normal distribution to meaningfull values on the response scale.*

Description

Functions to convert parameters of a log-normal distribution to meaningfull values on the response scale.

Usage

```
lnorm_mean(meanlog, sdlog, ...)

lnorm_median(meanlog, ...)

lnorm_var(meanlog, sdlog, ...)

lnorm_sd(meanlog, sdlog, ...)
```

Arguments

`meanlog, sdlog`	mean and standard deviation of the distribution on the log scale with default values of `0` and `1` respectively.
`...`	Not used

Examples

```
x <- rlnorm(1e4, meanlog = 1.5, sdlog = 1.2)

m <- lm(log(x) ~ 1)

meanlog <- coef(m)
sdlog <- sigma(m)

lnorm_mean(meanlog, sdlog)
mean(x)
```

```
lnorm_median(meanlog, sdlog)
median(x)

lnorm_sd(meanlog, sdlog)
sd(x)
```

`logLik_lnorm` *Information criteria for log-normal models*

Description

Log-likelihood (and by extension *AIC* and *BIC*) for log-normal models fit with `stats::lm(log(y) ~ ...)` are computed with `stats::dnorm(log(y), ...)` instead of with `stats::dlnorm(y, ...)`, which makes comparing different families difficult. This function is aimed at rectifying this. See examples.

Usage

```
logLik_lnorm(object, REML = FALSE)

AIC_lnorm(object, k = 2, REML = FALSE)

BIC_lnorm(object, REML = FALSE)
```

Arguments

`object`	A fitted model object. The model must meet all of the following (will throw an error if not met): 1. A Gaussian likelihood with an identity link function. 2. The LHS of the model's formula must use the `log()` function. 3. No weights (not yet supported).
`REML`	Only `FALSE` supported.
`k`	numeric, the *penalty* per parameter to be used; the default `k = 2` is the classical AIC.

Note

`REML` is not (yet) supported. Make sure you are comparing correct LL/AIC/BIC values.

Examples

```
data("mtcars")
mtcars$mpg <- floor(mtcars$mpg)

model_lnorm <- lm(log(mpg) ~ factor(cyl), mtcars)
model_norm <- lm(mpg ~ factor(cyl), mtcars)
model_pois <- glm(mpg ~ factor(cyl), mtcars, family = poisson())

# No, that first one is wrong...
(aics <- AIC(model_lnorm, model_norm, model_pois))

aics[1, "AIC"] <- AIC_lnorm(model_lnorm)
aics # better!


# Should support any model really... ====================
model_lnorm <- lme4::lmer(log(mpg) ~ factor(cyl) + (1 | gear), mtcars, REML = FALSE)
model_norm <- lme4::lmer(mpg ~ factor(cyl) + (1 | gear), mtcars, REML = FALSE)
model_pois <- lme4::glmer(mpg ~ factor(cyl) + (1 | gear), mtcars, family = poisson())

# No, that first one is wrong...
(aics <- AIC(model_lnorm, model_norm, model_pois))

aics[1, "AIC"] <- AIC_lnorm(model_lnorm)
aics # better!
```

`Paste` *Copy / Paste as Comment / Roxygen*

Description

Adpated from stla/pasteAsComment

Usage

```
Paste(prefix = "#> ")

PasteRoxygen()

Copy(prefix = "#> ")

CopyRoxygen()
```

Arguments

`prefix`	The prefix to use for the copied text.

`php.t` *Function for Post-Hoc Power analysis*

Description

Based on Lenth (2007) *Post Hoc Power: Tables and Commentary*. `php.guf()` give's Lenth's *grand unified formula for post hoc power*.

Usage

```
php.t(tval, p, df = Inf, alpha = 0.05, one.tailed = FALSE)

php.F(Fval, p, df1, df2 = Inf, alpha = 0.05)

php.z(zval, p, alpha = 0.05, one.tailed = FALSE)

php.chisq(chisqval, p, df, alpha = 0.05)

php.guf(p, alpha = 0.05)
```

Arguments

`tval`, `zval`, `Fval`, `chisqval`	Observed test statistic
`p`	Observed p-value (used if test statistic not supplied)
`df`, `df1`, `df2`	Test statistics' degrees of freedom
`alpha`	Confidence level of the test.
`one.tailed`	Is the p-value from a one-tailed test?

Examples

```
lm(hp ~ am, mtcars) |> summary()

php.t(p = 0.18, df = 30)

php.guf(p = 0.18)



# Table 1 - one tailed
expand.grid(
  p = c(.001, .01, .05, .1, .25, .5, .75),
  df = c(1, 2, 5, 10, 20, 50, 200, 1000, Inf)
) |>
  transform(PHP = php.t(p = p, df = df, one.tailed = TRUE)) |>
  stats::reshape(
    direction = "wide",
```

```
    idvar = "df", timevar = "p"
  )


# Table 1 - two tailed
expand.grid(
  p = c(.001, .01, .05, .1, .25, .5, .75),
  df = c(1, 2, 5, 10, 20, 50, 200, 1000, Inf)
) |>
  transform(PHP = php.t(p = p, df = df)) |>
  stats::reshape(
    direction = "wide",
    idvar = "df", timevar = "p"
  )


# Table 2
expand.grid(
  p = c(.001, .01, .05, .1, .25, .5, .75),
  df2 = c(1, 2, 5, 10, 20, 50, 200, 1000, Inf),
  df1 = c(2, 3, 4, 10)
) |>
  transform(PHP = php.F(p = p, df1 = df1, df2 = df2)) |>
  stats::reshape(
    direction = "wide",
    idvar = c("df1", "df2"), timevar = "p"
  )
```

`print_library` *Print Loading/Attaching of Packages*

Description

Useful in RMarkdown.

Usage

```
print_library(..., .character.only = FALSE, .version = TRUE, .load = TRUE)

print_require(..., .character.only = FALSE, .version = TRUE, .load = TRUE)

print_library_md(..., .character.only = FALSE, .version = TRUE, .load = TRUE)
```

Arguments

`...`	Names of packages.
`.character.only`	Is `...` from characters?

.version	Print library version?
.load	Load package, or just print?

Examples

```
print_library(afex, tidyverse, emmeans, MASS,
  .load = FALSE
)
```

---

qq_worm_plot *Create a worm plot*

Description

This function is a wrapper around qqplotr::stat_pp_*(detrend = TRUE).

Usage

```
qq_worm_plot(x, distribution = "norm", ...)
```

Arguments

x	A numerical vector
distribution	Name of a distribution, matching the d*, p* and q* function names.
...	Args passed to d*, p* and q* functions.

Details

Some related tweets.

Examples

```
x <- rnorm(100)
qq_worm_plot(x)

x <- rbeta(100, shape1 = 2, shape2 = 3)
qq_worm_plot(x, distribution = "beta", shape1 = 2, shape2 = 3)

x <- rt(100, df = 3)
qq_worm_plot(x, distribution = "t", df = 3)

# x <- rexp(100)
# qq_worm_plot(x, distribution = "exp")

# x <- rpois(100, lambda = 15)
# qq_worm_plot(x, distribution = "pois", lambda = 15)

# x <- runif(100)
```

```
# qq_worm_plot(x, distribution = "unif")
```

R2 R^2 *Overkill*

Description

Way too many ways to calculate R^2.

Usage

```
R2(pred, obs, type = 1, na.rm = TRUE)
```

Arguments

pred	The predicted values by some model; typically the result of a call to predict().
obs	The true observed values.
type	Which of the 8 versions of *R*-square to use. See details.
na.rm	a logical value indicating whether NA values should be stripped before the computation proceeds.

Details

The types of R^2:

- $R_1^2 = 1 - \sum(y - \hat{y})^2 / \sum(y - \bar{y})^2$
- $R_2^2 = \sum(\hat{y} - \bar{y})^2 / \sum(y - \bar{y})^2$
- $R_3^2 = \sum(\hat{y} - \bar{\hat{y}})^2 / \sum(y - \bar{y})^2$
- $R_4^2 = 1 - \sum(e - \bar{e})^2 / \sum(y - \bar{y})^2$
- $R_5^2 =$ squared multiple correlation coefficient between the regressand and the regressors
- $R_6^2 = r_{y,\hat{y}}^2$
- $R_7^2 = 1 - \sum(y - \hat{y})^2 / \sum y^2$
- $R_8^2 = \sum \hat{y}^2 / \sum y^2$

References

Kvålseth, T. O. (1985). Cautionary note about R 2. The American Statistician, 39(4), 279-285.

Examples

```
X <- c(1, 2, 3, 4, 5, 6)
Y <- c(15, 37, 52, 59, 83, 92)

m1 <- lm(Y ~ X)
m2 <- lm(Y ~ 0 + X)
m3 <- nls(Y ~ a * X^b, start = c(a = 1, b = 1))

# Table 2 from Kvålset0 (1985)
data.frame(
  mod1 = sapply(1:8, R2, pred = predict(m1), obs = Y),
  mod2 = sapply(1:8, R2, pred = predict(m2), obs = Y),
  mod3 = sapply(1:8, R2, pred = 16.3757 * X^0.99, obs = Y)
)
```

---

r_SB *Spearman-Brown Split half reliability*

Description

Spearman-Brown Split half reliability

Usage

```
r_SB(x, y = NULL, var.equal = TRUE)
```

Arguments

x	A correlation or numeric vector
y	A numeric vector
var.equal	Assume equal var of x and y? (ignored if y is not NULL)

Examples

```
r_SB(1:30, -exp(1 / 1:30), var.equal = TRUE)

r_SB(1:30, -exp(1 / 1:30), var.equal = FALSE)

r_SB(0.57)
```

seq_range *Sequence Generation Based on the Values of a Vector*

Description

Sequence Generation Based on the Values of a Vector

Usage

```
seq_range(
  x,
  length.out = NULL,
  by = NULL,
  along.with = NULL,
  na.rm = TRUE,
  padding = 0.05
)

seq_quantile(
  x,
  probs,
  length.out = NULL,
  by = NULL,
  along.with = NULL,
  na.rm = TRUE
)

seq_IQR(x, length.out = NULL, by = NULL, along.with = NULL, na.rm = TRUE)

mean_sd(x, na.rm = TRUE, out = c("vector", "data.frame"))

median_mad(x, na.rm = TRUE, out = c("vector", "data.frame"))
```

Arguments

x	A numeric vector
length.out	desired length of the sequence. If no other arguments are valued, defaults to 20.
by	number: increment of the sequence.
along.with	take the length from the length of this argument.
na.rm	a logical evaluating to TRUE or FALSE indicating whether NA values should be stripped before the computation proceeds.
padding	Padding factor for the range.
probs	numeric vector of probabilities with values in $[0, 1]$. (Values up to '2e-14' outside that range are accepted and moved to the nearby endpoint.)
out	If "data.frame" can be used as a summary function in ggplot2.

Examples

```
set.seed(1)
x <- rt(100, df = 3)
seq_range(x, length.out = 5)
seq_IQR(x, length.out = 5)
seq_quantile(x, c(.05, .95), length.out = 5)

mean_sd(x)


library(ggplot2)
ggplot(mtcars, aes(cyl, mpg)) +
  stat_summary(aes(color = "Mean (SD)"),
    fun.data = mean_sd,
    fun.args = list(out = "data.frame")
  ) +
  stat_summary(aes(color = "Median (MAD)"),
    fun.data = median_mad,
    fun.args = list(out = "data.frame"),
    position = position_nudge(x = 0.5)
  )
```

---

simple_effects *Compute Simple Effects Omnibus Tests*

Description

This is a wrapper for `emmeans::joint_tests()` that provides an easy way to specify which simple effects we wish to test, and within what variable(s).

Usage

```
simple_effects(model, effect, inside, ...)
```

Arguments

`model`	The model.
`effect`	The name of the required simple effect. e.g., `"A"` for a simple effect of *A*, or `"A:B"` for a simple *A-by-B* interaction.
`inside`	A vector of the name(s) of the variable(s) within whose levels the `effect` will be tested. Can also be the name of an interaction (e.g., `"B:C"`). If not specified, will use all the terms not in `effect`.
`...`	Passed to `emmeans::joint_tests()`, e.g., `cov.reduce`, `at`, etc.

Examples

```
library(afex)

data(obk.long, package = "afex")
A <- aov_car(value ~ treatment * gender + Error(id / (phase * hour)),
  data = obk.long
)

simple_effects(A, effect = "treatment")

simple_effects(A, effect = "treatment:phase")

simple_effects(A, effect = "phase", inside = "treatment")

simple_effects(A, effect = "phase", inside = c("treatment", "gender"))
# simple_effects(A, effect = "phase", inside = "treatment:gender") # same

simple_effects(A,
  effect = "phase", inside = c("treatment", "gender"),
  at = list(gender = "F")
)

simple_effects(A, effect = "phase:treatment", inside = "gender")
```

---

simple_mediation_plot *Plot simple mediations with* tidySEM

Description

Plot simple mediations with tidySEM

Usage

```
simple_mediation_plot(
  a = NA,
  b = NA,
  direct = NA,
  indirect = NA,
  total = NA,
  X_name = "X",
  M_name = "M",
  Y_name = "Y",
  ...
)
```

Arguments

a, b, direct, indirect, total	Values or labels to put on the paths (edges).
X_name, M_name, Y_name	Values or labels to put on the variables (nodes).
...	Passed to tidySEM::prepare_graph.

Examples

```
mod_a <- lm(hp ~ gear, data = mtcars)
mod_bc <- lm(mpg ~ hp + gear, data = mtcars)

a <- coef(mod_a)[2]
b <- coef(mod_bc)[2]
direct <- coef(mod_bc)[3]
indirect <- a * b
total <- direct + indirect

med_plot <- simple_mediation_plot(
  a = round(a, 3),
  b = round(b, 3),
  direct = round(direct, 3),
  indirect = round(indirect, 3),
  total = round(total, 3),
  X_name = "Gears",
  M_name = "Horse\nPower",
  Y_name = "Miles\nPer Gallon"
)

plot(med_plot)
```

---

stat_demo_apps *Some stats demos*

Description

Some stats demos

Usage

```
stat_demo_apps(
  demo = c("paired ttest", "truncated correlation", "berksons paradox")
)
```

Arguments

demo	Which demo to show? (Partial matching supported)

Details

paired ttest: Shows a paired vs un-paired ttest, and how these differences are affected by the correlation.

truncated correlation: Demo of how truncation affects correlations and doesn't affect MSE.

berksons paradox: Demo of how How sampling bias can affect estimated correlations and effects. See related Numberphile video.

`vlookup` *vlookup*

Description

vlookup

Usage

```
vlookup(this, data, key, value, add = FALSE)
```

Arguments

`this`	A vector of values
`data`	A data frame to search in
`key`	Where should `this` be looked up?
`value`	Name of the column from which values should be returned.
`add`	Should `this` and the resulting values be returned as a data frame? (Else a vector)

Examples

```
df <- data.frame(
  a = letters[c(1, 1:9)],
  b = 51:60
)

vlookup(c("a", "e", "c"), df, key = "a", value = "b")
vlookup(c("a", "e", "c"), df, key = "a", value = "b", add = TRUE)
```

Index