Package ‘pwrss’

September 16, 2025

Version 1.0.0

Type Package

Title Statistical Power and Sample Size Calculation Tools

Date 2025-09-13

Description The 'pwrss' R package provides flexible and comprehensive functions for statistical power and minimum required sample size calculations across a wide range of commonly used hypothesis tests in psychological, biomedical, and social sciences.

Suggests knitr, rmarkdown

VignetteBuilder knitr

ByteCompile yes

LazyLoad yes

License GPL (>= 3)

Maintainer Metin Bulus <bulusmetin@gmail.com>

RoxygenNote 7.3.3

Encoding UTF-8

Language en-US

NeedsCompilation no

Author Metin Bulus [aut, cre],
Sebastian Jentschke [ctb]

Repository CRAN

Date/Publication 2025-09-16 05:40:02 UTC

Contents

`cors.to.q` *Conversion from Correlation Difference to Cohen's q*

Description

Helper function to convert correlation difference to Cohen's q (and vice versa). `cor.to.z()` function applies Fisher's z transformation.

Usage

```
cor.to.z(rho, verbose = TRUE)

cors.to.q(rho1, rho2, verbose = TRUE)

q.to.cors(q, rho1 = NULL, rho2 = NULL, verbose = TRUE)
```

Arguments

rho	correlation.
rho1	first correlation.
rho2	second correlation.
q	Cohen's q effect size.
verbose	logical; whether the output should be printed on the console. TRUE by default.

Value

rho1	first correlation.
rho2	second correlation.
delta	correlation difference: rho1 - rho2.
q	Cohen's q effect size.

Examples

```
q.to.cors(q = 0.10, rho1 = 0.50)
q.to.cors(q = 0.30, rho1 = 0.50)
q.to.cors(q = 0.50, rho1 = 0.50)

cors.to.q(rho2 = 0.5712027, rho1 = 0.50)
cors.to.q(rho2 = 0.6907068, rho1 = 0.50)
cors.to.q(rho2 = 0.7815365, rho1 = 0.50)
```

---

d.to.cles *Conversion from Cohen's d to Common Language Effect Size*

Description

Helper function to convert Cohen's d to common language effect size (or vice versa). The result is the probability of superiority for independent samples. It can be interpreted as the probability that a randomly selected observation from Group 1 exceeds a randomly selected observation from Group 2. The rationale is the same for paired-samples and one-sample designs, but the interpretation differs: For paired samples, it can be interpreted as the probability that the difference score (i.e., the score under Condition 1 minus the score under Condition 2) is greater than zero for a randomly selected individual. For a one-sample design, it can be interpreted as the probability that a randomly selected observation is greater than the reference value (e.g., 0).

Usage

```
d.to.cles(d, design = c("independent", "paired", "one.sample"), verbose = TRUE)

cles.to.d(cles, design = c("independent", "paired", "one.sample"), verbose = TRUE)
```

Arguments

d	Cohen's d
design	character; one of the "independent", "paired", or "one.sample". The default is "independent".
cles	common language effect size.
verbose	logical; whether the output should be printed on the console. TRUE by default.

Value

d	Cohen's d
cles	common language effect size.

Examples

```
d.to.cles(0.20) # small
d.to.cles(0.50) # medium
d.to.cles(0.80) # large

cles.to.d(0.5562315)
cles.to.d(0.6381632)
cles.to.d(0.7141962)
```

---

f.to.etasq *Conversion from Cohen's f to Eta-squared*

Description

Helper function to convert between Cohen's f and Eta-squared.

Usage

```
f.to.etasq(f, verbose = TRUE)

etasq.to.f(eta.squared, verbose = TRUE)
```

Arguments

f	Cohen's f.
eta.squared	(Partial) Eta-squared.
verbose	logical; whether the output should be printed on the console. TRUE by default.

Value

f	Cohen's f.
eta.squared	(Partial) Eta-squared.

References

Cohen, J. (1988). Statistical power analysis for the behavioral sciences (2nd ed.). Lawrence Erlbaum Associates.

Examples

```
f.to.etasq(f = 0.10) # small
f.to.etasq(f = 0.25) # medium
f.to.etasq(f = 0.40) # large
```

`f.to.rsq` *Conversion from Cohen's f to R-squared*

Description

Helper function to convert between Cohen's f and R-squared.

Usage

```
rsq.to.f(r.squared.full, r.squared.reduced = 0, verbose = TRUE)

f.to.rsq(f, r.squared.full = NULL, verbose = TRUE)
```

Arguments

`f`	Cohen's f.
`r.squared.full`	R-squared for the full model.
`r.squared.reduced`	R-squared for the reduced model.
`verbose`	logical; whether the output should be printed on the console. `TRUE` by default.

Value

`f`	Cohen's f.
`f.squared`	Cohen's f-squared.
`r.squared.full`	R-squared for the full model.
`r.squared.reduced`	R-squared for the reduced model.

References

Cohen, J. (1988). Statistical power analysis for the behavioral sciences (2nd ed.). Lawrence Erlbaum Associates.

Selya, A. S., Rose, J. S., Dierker, L. C., Hedeker, D., & Mermelstein, R. J. (2012). A practical guide to calculating Cohen's f2, a measure of local effect size, from PROC MIXED. Frontiers in Psychology, 3, 111. doi: 10.3389/fpsyg.2012.00111

Examples

```
f.to.rsq(f = 0.10) # small
f.to.rsq(f = 0.25) # medium
f.to.rsq(f = 0.40) # large
```

`factorial.contrasts` *Factorial Contrasts*

Description

Helper function to construct the default contrast coefficients for various coding schemes.

Note that R has a partial matching feature which allows you to specify shortened versions of arguments, such as `coding` instead of `coding.scheme`.

Validated using `lm()` and `aov()` functions.

Usage

```
factorial.contrasts(factor.levels = c(3, 2),
                    coding.scheme = rep("deviation", length(factor.levels)),
                    base = factor.levels,
                    intercept = FALSE,
                    verbose = TRUE)
```

Arguments

`factor.levels`	Integer. Number of levels or groups in each factor. For example, for two factors each having two levels or groups use e.g. c(2, 2), for three factors each having two levels or groups use e.g. c(2, 2, 2)
`coding.scheme`	Character vector. Coding scheme for each factor. "sum" for deviation or effect coding, "treatment" for dummy coding, "helmert" for Helmert type of coding, and "poly" for polynomial coding. Each factor can have their own coding scheme. If a single character value is provided, it will be copied to other factors
`base`	Integer vector. Specifies which group is considered the baseline group. Ignored for coding schemes other than "treatment"
`intercept`	Logical. `FALSE` by default. If `TRUE` contrast matrix includes the intercept
`verbose`	Logical. `TRUE` by default. If `FALSE` no output is printed on the console

Value

`factor.levels`	Number of levels (or groups) in each factor
`factor.data`	Unique combination of factor levels
`model.matrix`	Model (design) matrix based on unique combination of factor levels
`contrast.matrix`	Contrast matrix

Examples

```
#################################################
############### dummy coding scheme ###############
#################################################

# one factor w/ 3 levels
contrast.object <- factorial.contrasts(factor.levels = 3,
                                       coding = "treatment")
# model matrix
contrast.object$model.matrix

# contrast matrix
contrast.object$contrast.matrix

#################################################
###############  deviation coding scheme ##############
#################################################

# especially useful for factorial designs
# two factors w/ 2 and 3 levels, respectively
contrast.object <- factorial.contrasts(factor.levels = c(2, 3),
                                       coding = "sum")

# model matrix
contrast.object$model.matrix

# contrast matrix
contrast.object$contrast.matrix


#################################################
###############  Helmert coding scheme ###############
#################################################

# one factor w/ 3 levels
contrast.object <- factorial.contrasts(factor.levels = 3,
                                       coding = "helmert")

# model matrix
contrast.object$model.matrix

# contrast matrix
contrast.object$contrast.matrix

#################################################
###############  polynomial coding scheme ###############
#################################################

# one factor w/ 3 levels
contrast.object <- factorial.contrasts(factor.levels = 3,
                                       coding = "poly")
```

```
# model matrix
contrast.object$model.matrix

# contrast matrix
contrast.object$contrast.matrix
```

`inflate.sample` *Inflate Sample Size for Attrition*

Description

Helper function to inflate sample size for attrition.

Usage

```
inflate.sample(n, rate = 0.05,
               ceiling = TRUE,
               verbose = TRUE)
```

Arguments

`n`	sample size.
`rate`	attrition rate.
`ceiling`	rounds-up the inflated sample size.
`verbose`	logical; whether the output should be printed on the console. `TRUE` by default.

Value

inflated sample size.

Examples

```
inflate.sample(n = 100, rate = 0.05)
```

joint.probs.2x2 *Conversion Between Joint and Marginal Probabilities for the McNemar Test*

Description

Helper function to converts joint probabilities to marginal probabilities (and vice versa) for the McNemar test applied to paired binary data.

Usage

```
joint.probs.2x2(prob1, prob2, rho = 0.50, verbose = TRUE)

marginal.probs.2x2(prob11, prob10, prob01, prob00, verbose = TRUE)
```

Arguments

prob1	(marginal) probability of success in case group (or after).
prob2	(marginal) probability of success in matched-control group (or before).
rho	the correlation between case and matched-control, or after and before (phi coefficient).
prob11	(joint) probability of success in both groups. 'prob11' and 'prob00' are known as concordant probs.
prob10	(joint) probability of success in case (or after) but failure in matched control (or before). 'prob10' and 'prob01' are known as discordant probs.
prob01	(joint) probability of failure in case (or after) but success in matched control (or before). prob10' and 'prob01' are known as discordant probs.
prob00	(joint) probability of failure in both groups. 'prob11' and 'prob00' are known as concordant probs.
verbose	if FALSE no output is printed on the console.

Value

parms	list of parameters used in calculation.
prob1	(marginal) probability of success in case group (or after).
prob2	(marginal) probability of success in matched-control group (or before).
rho	the correlation between case and matched-control, or after and before (phi coefficient).
prob11	(joint) probability of success in both groups. 'prob11' and 'prob00' are known as concordant probs.
prob10	(joint) probability of success in case (or after) but failure in matched control (or before). 'prob10' and 'prob01' are known as discordant probs.

prob01 (joint) probability of failure in case (or after) but success in matched control (or before). prob10' and 'prob01' are known as discordant probs.

prob00 (joint) probability of failure in both groups. 'prob11' and 'prob00' are known as concordant probs.

References

Zhang, S., Cao, J., and Ahn, C. (2017). Inference and sample size calculation for clinical trials with incomplete observations of paired binary outcomes. Statistics in Medicine, 36(4), 581-591. doi: 10.1002/sim.7168

Examples

```
# example data for a matched case-control design
# subject  case     control
# <int>    <dbl>    <dbl>
#   1        1        1
#   2        0        1
#   3        1        0
#   4        0        1
#   5        1        1
#   ...     ...      ...
#   100      0        0

# example summary stats
# prob1 = mean(case) which is 0.55
# prob2 = mean(control) which is 0.45
# rho = cor(case, control) which is 0.4141414


# example data for a before-after design
# subject  before   after
# <int>    <dbl>    <dbl>
#   1        1        1
#   2        0        1
#   3        1        0
#   4        0        1
#   5        1        1
#   ...     ...      ...
#   100      0        0

# example summary stats
# prob1 = mean(after) which is 0.55
# prob2 = mean(before) which is 0.45
# rho = cor(after, before) which is 0.4141414

# convert to a 2 x 2 frequency table
freqs <- matrix(c(30, 10, 20, 40), nrow = 2, ncol = 2)
colnames(freqs) <- c("control_1", "control_0")
rownames(freqs) <- c("case_1", "case_0")
freqs
```

```
# convert to a 2 x 2 proportion table
props <- freqs / sum(freqs)
props

# discordant pairs (0 and 1, or 1 and 0) in 'props' matrix
# are the sample estimates of prob01 and prob10


# we may not have 2 x 2 joint probs
# convert marginal probs to joint probs using summary stats
jp <- joint.probs.2x2(prob1 = 0.55, # mean of case (or after)
                          prob2 = 0.45, # mean of matched control (or before)
                          # correlation b/w matched case-control / before-after
                          rho = 0.4141414)

# required sample size for exact test
# assuming prob01 and prob10 are population parameters
power.exact.mcnemar(prob01 = jp$prob01,
                    prob10 = jp$prob10,
                    power = 0.80, alpha = 0.05,
                    method = "exact")

# convert joint probs to marginal probs and calc phi coefficient (rho)
# these values can be used in other procedures
marginal.probs.2x2(prob11 = 0.35, # mean of case (or after)
                    prob10 = 0.20, # mean of matched control (or before)
                    prob01 = 0.10,
                    prob00 = 0.35)
```

`means.to.d` *Conversion from Means and Standard Deviations to Cohen's d*

Description

Helper function to convert means and standard deviations to Cohen's d.

Usage

```
means.to.d(mu1, mu2 = 0,
           sd1 = 1, sd2 = 1,
           n2, n.ratio = 1,
           paired = FALSE,
           rho.paired = 0.50,
           verbose = TRUE)
```

Arguments

`mu1`	mean of the first group.
`mu2`	mean of the second group.

sd1	standard deviation of the first group.
sd2	standard deviation of the second group.
n.ratio	n1/n2 ratio (applies to independent samples only).
paired	if TRUE paired samples
rho.paired	correlation between repeated measures for paired samples (e.g., pretest and posttest).
n2	integer; sample size in the second group (or for the single group in paired samples).
verbose	logical; whether the output should be printed on the console. TRUE by default.

Value

d	Cohen's d

Examples

```
# means and standard deviations from independent samples
means.to.d(mu1 = 20, mu2 = 17.5,
           sd1 = 5, sd2 = 15,
           n2 = 30, n.ratio = 1)

# means and standard deviations from paired samples
means.to.d(mu1 = 20, mu2 = 17.5,
           sd1 = 5, sd2 = 15,
           n2 = 30, n.ratio = 1,
           paired = TRUE,
           rho.paired = 0.50)
```

---

power.binom *Power Analysis for the Generic Binomial Test*

Description

Calculates power for the generic binomial test with (optional) Type 1 and Type 2 error plots.

Usage

```
power.binom.test(size, prob, null.prob = 0.5, alpha = 0.05,
                 alternative = c("two.sided", "one.sided", "two.one.sided"),
                 plot = TRUE, verbose = TRUE, pretty = FALSE)
```

Arguments

size	number of trials (zero or more).
prob	probability of success on each trial under alternative.
null.prob	probability of success on each trial under null.
alpha	type 1 error rate, defined as the probability of incorrectly rejecting a true null hypothesis, denoted as α.
alternative	direction or type of the hypothesis test: "two.sided", "one.sided", or "two.one.sided". For non-inferiority or superiority tests, add or subtract the margin from the null hypothesis value and use alternative = "one.sided".
plot	logical; FALSE switches off Type 1 and Type 2 error plot. TRUE by default.
verbose	logical; whether the output should be printed on the console. TRUE by default.
pretty	logical; whether the output should show Unicode characters (if encoding allows for it). FALSE by default.

Value

size	number of trials (zero or more).
prob	probability of success on each trial under alternative.
null.prob	probability of success on each trial under null.
binom.alpha	critical value(s).
power	statistical power $(1 - \beta)$.

Examples

```
# one-sided
power.binom.test(size = 200, prob = 0.6, null.prob = 0.5,
                 alpha = 0.05, alternative = "one.sided")

# two-sided
power.binom.test(size = 200, prob = 0.4, null.prob = 0.5,
                 alpha = 0.05, alternative = "two.sided")

# equivalence
power.binom.test(size = 200, prob = 0.5, null.prob = c(0.4, 0.6),
                 alpha = 0.05, alternative = "two.one.sided")
```

---

power.chisq *Statistical Power for the Generic Chi-square Test*

Description

Calculates power for the generic chi-square test with (optional) Type 1 and Type 2 error plots.

Usage

```
power.chisq.test(ncp, null.ncp = 0, df, alpha = 0.05,
                 plot = TRUE, verbose = TRUE, pretty = FALSE)
```

Arguments

ncp	non-centrality parameter for the alternative.
null.ncp	non-centrality parameter for the null.
df	integer; degrees of freedom. For example, for the test of independence df = (nrow - 1)*(ncol - 1).
alpha	type 1 error rate, defined as the probability of incorrectly rejecting a true null hypothesis, denoted as α.
plot	logical; FALSE switches off Type 1 and Type 2 error plot. TRUE by default.
verbose	logical; whether the output should be printed on the console. TRUE by default.
pretty	logical; whether the output should show Unicode characters (if encoding allows for it). FALSE by default.

Value

power	statistical power $(1 - \beta)$.

Examples

```
# power is defined as the probability of observing Chi-square-statistics
# greater than the critical  value
power.chisq.test(ncp = 20, df = 100, alpha = 0.05)
```

---

power.chisq.gof *Power and Sample Size for Chi-square Goodness-of-Fit or Independence Tests*

Description

Calculates power or sample size (only one can be NULL at a time) for Chi-square goodness-of-fit or independence tests.

NOTE: The pwrss.chisq.gofit() function is deprecated. However, it will remain available as a wrapper for the power.chisq.gof() function.

Usage

```
power.chisq.gof(w, null.w = 0, df,
                n = NULL, power = NULL, alpha = 0.05,
                ceiling = TRUE, verbose = TRUE, pretty = FALSE)
```

Arguments

w	Cohen's w effect size under alternative. It can be any of Cohen's W, Phi coefficient, or Cramer's V but degrees of freedom should be specified accordingly. Phi coefficient is defined as `sqrt(X2/n)` and Cramer's V is defined as `sqrt(X2/(n*v))` where v is `min(nrow - 1, ncol - 1)` and X2 is the chi-square statistic.
null.w	Cohen's w effect size under null.
df	integer; degrees of freedom. Defined as (n.cells - 1) if p1 is a vector, and as (n.rows - 1) * (n.cols - 1) if p1 is a matrix.
n	integer; total sample size.
power	statistical power, defined as the probability of correctly rejecting a false null hypothesis, denoted as $1 - \beta$.
alpha	type 1 error rate, defined as the probability of incorrectly rejecting a true null hypothesis, denoted as α.
ceiling	logical; whether sample size should be rounded up. TRUE by default.
verbose	logical; whether the output should be printed on the console. TRUE by default.
pretty	logical; whether the output should show Unicode characters (if encoding allows for it). FALSE by default.

Value

parms	list of parameters used in calculation.
test	type of the statistical test (Chi-square Test).
df	degrees of freedom.
ncp	non-centrality parameter under alternative.
null.ncp	non-centrality parameter under null.
chisq.alpha	critical value.
power	statistical power $(1 - \beta)$.
n	total sample size.

References

Cohen, J. (1988). Statistical power analysis for the behavioral sciences (2nd ed.). Lawrence Erlbaum Associates.

Examples

```
# ------------------------------------------------------------#
# Example 1: Cohen's W                                        #
# goodness-of-fit test for 1 x k or k x 1 table               #
# How many subjects are needed to claim that                  #
# girls choose STEM related majors less than males?          #
# ------------------------------------------------------------#

## Option 1: Use cell probabilities
```

```
## from https://www.aauw.org/resources/research/the-stem-gap/
## 28 percent of the  workforce in STEM field is women
prob.vector <- c(0.28, 0.72)
null.prob.vector <- c(0.50, 0.50)
probs.to.w(prob.vector, null.prob.vector)

power.chisq.gof(w = 0.44, df = 1,
                alpha = 0.05, power = 0.80)


# ----------------------------------------------------------#
# Example 2: Phi Coefficient (or Cramer's V or Cohen's W)  #
# test of independence for 2 x 2 contingency tables        #
# How many subjects are needed to claim that               #
# girls are underdiagnosed with ADHD?                      #
# ----------------------------------------------------------#

## from https://time.com/growing-up-with-adhd/
## 5.6 percent of girls and 13.2 percent of boys are diagnosed with ADHD
prob.matrix <- rbind(c(0.056, 0.132),
                     c(0.944, 0.868))
colnames(prob.matrix) <- c("Girl", "Boy")
rownames(prob.matrix) <- c("ADHD", "No ADHD")
prob.matrix

probs.to.w(prob.matrix)

power.chisq.gof(w = 0.1302134, df = 1,
                alpha = 0.05, power = 0.80)


# ----------------------------------------------------------#
# Example 3: Cramer's V (or Cohen's W)                     #
# test of independence for j x k contingency tables        #
# How many subjects are needed to detect the relationship #
# between depression severity and gender?                  #
# ----------------------------------------------------------#

## from https://doi.org/10.1016/j.jad.2019.11.121
prob.matrix <- cbind(c(0.6759, 0.1559, 0.1281, 0.0323, 0.0078),
                     c(0.6771, 0.1519, 0.1368, 0.0241, 0.0101))
rownames(prob.matrix) <- c("Normal", "Mild", "Moderate",
                           "Severe", "Extremely Severe")
colnames(prob.matrix) <- c("Female", "Male")
prob.matrix

probs.to.w(prob.matrix)

power.chisq.gof(w = 0.03022008, df = 4,
                alpha = 0.05, power = 0.80)
```

power.exact.fisher *Power Analysis for Fisher's Exact Test (Independent Proportions)*

Description

Calculates power or sample size for Fisher's exact test on independent binary outcomes. Approximate and exact methods are available.

Validated via PASS and G*Power.

Usage

```
power.exact.fisher(prob1, prob2, n2 = NULL, n.ratio = 1,
                   alpha = 0.05, power = NULL,
                   alternative = c("two.sided", "one.sided"),
                   method = c("exact", "approximate"),
                   ceiling = TRUE, verbose = TRUE, pretty = FALSE)
```

Arguments

prob1	probability of success in the first group.
prob2	probability of success in the second group.
n2	integer; sample size for the second group.
n.ratio	n1 / n2 ratio.
power	statistical power, defined as the probability of correctly rejecting a false null hypothesis, denoted as $1 - \beta$.
alpha	type 1 error rate, defined as the probability of incorrectly rejecting a true null hypothesis, denoted as α.
alternative	character; direction or type of the hypothesis test: "two.sided" or "one.sided".
method	character; method used for power calculation. "exact" specifies Fisher's exact test, while "approximate" refers to the Z-Test based on the normal approximation.
ceiling	logical; if TRUE rounds up sample size in each group.
verbose	logical; if FALSE no output is printed on the console.
pretty	logical; whether the output should show Unicode characters (if encoding allows for it). FALSE by default.

Value

parms	list of parameters used in calculation.
test	type of the test, which is "exact" or "z".
odds.ratio	odds ratio.
mean	mean of the alternative distribution.

sd	standard deviation of the alternative distribution.
null.mean	mean of the null distribution.
null.sd	standard deviation of the null distribution.
z.alpha	critical value(s).
power	statistical power $(1 - \beta)$.
n	sample sizes for the first and second groups, specified as c(n1, n2).
n.total	total sample size, which is sum of cell frequencies in 2 x 2 table (f11 + f10 + f01 + f00), or number of rows in a data frame with group variable stacked.

References

Bennett, B. M., & Hsu, P. (1960). On the power function of the exact test for the 2 x 2 contingency table. Biometrika, 47(3/4), 393-398. doi: 10.2307/2333309

Fisher, R. A. (1935). The logic of inductive inference. Journal of the Royal Statistical Society, 98(1), 39-82. doi: 10.2307/2342435

Examples

```
# example data for a randomized controlled trial
# subject  group    success
# <int>    <dbl>      <dbl>
#   1        1          1
#   2        0          1
#   3        1          0
#   4        0          1
#   5        1          1
#   ...     ...        ...
#   100      0          0

# prob1 = mean(success | group = 1)
# prob2 = mean(success | group = 0)

# post-hoc exact power
power.exact.fisher(prob1 = 0.60, prob2 = 0.40, n2 = 50)

# we may have 2 x 2 joint probs such as
# -------------------------------------
#             | group (1) | group (0) |
# -------------------------------------
# success (1) |  0.24     |   0.36    |
# -------------------------------------
# success (0) |  0.16     |   0.24    |
# -------------------------------------

# convert joint probs to marginal probs
marginal.probs.2x2(prob11 = 0.24, prob10 = 0.36,
                   prob01 = 0.16, prob00 = 0.24)
```

power.exact.mcnemar *Power Analysis for McNemar's Exact Test (Paired Proportions)*

Description

Calculates power or sample size for McNemar's test on paired binary outcomes. Approximate and exact methods are available (check references for details).

Validated via PASS and G*Power.

Usage

```
power.exact.mcnemar(prob10, prob01, n.paired = NULL,
                    power = NULL,  alpha = 0.05,
                    alternative = c("two.sided", "one.sided"),
                    method = c("exact", "approximate"),
                    ceiling = TRUE, verbose = TRUE, pretty = FALSE)
```

Arguments

prob10	(joint) probability of success in case (or after) but failure in matched control (or before). 'prob10' and 'prob01' are known as discordant probs.
prob01	(joint) probability of failure in case (or after) but success in matched control (or before). prob10' and 'prob01' are known as discordant probs.
n.paired	number of pairs, which is sum of cell frequencies in 2 x 2 table (f11 + f10 + f01 + f00), or number of rows in a data frame with matched variables 'case' and 'control' or 'after' and 'before'.
power	statistical power, defined as the probability of correctly rejecting a false null hypothesis, denoted as $1 - \beta$.
alpha	type 1 error rate, defined as the probability of incorrectly rejecting a true null hypothesis, denoted as α.
method	character; the method used for power calculation. "exact" specifies Fisher's exact test, while "approximate" refers to the z-test based on the normal approximation.
alternative	character; the direction or type of the hypothesis test: "not equal", "greater", "less". Optionally, users can specify 'two.sided' as an alternative to 'not equal' for consistency with other R statistical functions.
ceiling	logical; if TRUE rounds up sample size in each cell. This procedure assumes symmetry for concordant probs, which are 'p11' and 'p00'). Thus results may differ from other software by a few units. To match results set 'ceiling = FALSE'.
verbose	logical; if FALSE no output is printed on the console.
pretty	logical; whether the output should show Unicode characters (if encoding allows for it). FALSE by default.

Value

`parms`	list of parameters used in calculation.
`test`	type of the test, which is "exact" or "z".
`odds.ratio`	odds ratio.
`mean`	mean of the alternative distribution.
`sd`	standard deviation of the alternative distribution.
`null.mean`	mean of the null distribution.
`null.sd`	standard deviation of the null distribution.
`z.alpha`	critical value(s).
`power`	statistical power $(1 - \beta)$.
`n.paired`	paired sample size, which is sum of cell frequencies in 2 x 2 table (f11 + f10 + f01 + f00), or number of rows in a data frame with variables 'case' and 'control' or 'after' and 'before'.

References

Bennett, B. M., & Underwood, R. E. (1970). 283. Note: On McNemar's Test for the 2 * 2 Table and Its Power Function. Biometrics, 26(2), 339-343. doi: 10.2307/2529083

Connor, R. J. (1987). Sample size for testing differences in proportions for the paired-sample design. Biometrics, 43(1), 207-211. doi: 10.2307/2531961

Duffy, S. W. (1984). Asymptotic and exact power for the McNemar test and its analogue with R controls per case. Biometrics, 40(4) 1005-1015. doi: 10.2307/2531151

McNemar, Q. (1947). Note on the sampling error of the difference between correlated proportions or percentages. Psychometrika, 12(2), 153-157. doi: 10.1007/BF02295996

Miettinen, O. S. (1968). The matched pairs design in the case of all-or-none responses. Biometrics, 24(2), 339-352. doi: 10.2307/2528039

Examples

```
# example data for a matched case-control design
# subject  case     control
# <int>    <dbl>    <dbl>
#   1        1        1
#   2        0        1
#   3        1        0
#   4        0        1
#   5        1        1
#   ...     ...      ...
#   100      0        0

# example data for a before-after design
# subject  before   after
# <int>    <dbl>    <dbl>
#   1        1        1
#   2        0        1
#   3        1        0
```

```
#   4        0        1
#   5        1        1
#   ...     ...      ...
#   100      0        0

# convert to a 2 x 2 frequency table
freqs <- matrix(c(30, 10, 20, 40), nrow = 2, ncol = 2)
colnames(freqs) <- c("control_1", "control_0")
rownames(freqs) <- c("case_1", "case_0")
freqs

# convert to a 2 x 2 proportion table
props <- freqs / sum(freqs)
props

# discordant pairs (0 and 1, or 1 and 0) in 'props' matrix
# are the sample estimates of prob01 and prob10

# post-hoc exact power
power.exact.mcnemar(prob10 = 0.20, prob01 = 0.10,
                    n.paired = 100, alpha = 0.05,
                    method = "exact", alt = "two.sided")

# required sample size for exact test
# assuming prob01 and prob10 are population parameters
power.exact.mcnemar(prob10 = 0.20, prob01 = 0.10,
                    power = 0.80, alpha = 0.05,
                    method = "exact", alt = "two.sided")

# we may not have 2 x 2 joint probs
# convert marginal probs to joint probs
joint.probs.2x2(prob1 = 0.55, # mean of case group (or after)
                    prob2 = 0.45, # mean of matched control group (or before)
                    # correlation between matched case-control or before-after
                    rho = 0.4141414
)
```

`power.f` *Statistical Power for the Generic F-Test*

Description

Calculates power for the generic F-Test with (optional) Type 1 and Type 2 error plots.

Usage

```
power.f.test(ncp, null.ncp = 0, df1, df2, alpha = 0.05,
             plot = TRUE, verbose = TRUE, pretty = FALSE)
```

Arguments

ncp	non-centrality parameter for the alternative.
null.ncp	non-centrality parameter for the null.
alpha	type 1 error rate, defined as the probability of incorrectly rejecting a true null hypothesis, denoted as α.
df1	integer; numerator degrees of freedom.
df2	integer; denominator degrees of freedom.
plot	logical; FALSE switches off Type 1 and Type 2 error plot. TRUE by default.
verbose	logical; whether the output should be printed on the console. TRUE by default.
pretty	logical; whether the output should show Unicode characters (if encoding allows for it). FALSE by default.

Value

df1	numerator degrees of freedom.
df2	denominator degrees of freedom.
ncp	non-centrality parameter under alternative.
ncp.null	non-centrality parameter under null.
f.alpha	critical value(s).
power	statistical power $(1 - \beta)$.

Examples

```
# power is defined as the probability of observing F-statistics
# greater than the critical value
power.f.test(ncp = 1, df1 = 4, df2 = 100, alpha = 0.05)
```

---

power.f.ancova *Power Analysis for One-, Two-, Three-Way ANOVA/ANCOVA Using Effect Size (F-Test)*

Description

Calculates power or sample size for one-way, two-way, or three-way ANOVA/ANCOVA. Set k.cov = 0 for ANOVA, and k.cov > 0 for ANCOVA. Note that in the latter, the effect size (eta.squared should be obtained from the relevant ANCOVA model, which is already adjusted for the explanatory power of covariates (thus, an additional R-squared argument is not required as an input).

Note that R has a partial matching feature which allows you to specify shortened versions of arguments, such as k or k.cov instead of k.covariates.

Formulas are validated using G*Power and tables in PASS documentation.

Usage

```
power.f.ancova(eta.squared,
               null.eta.squared = 0,
               factor.levels = 2,
               k.covariates = 0,
               n.total = NULL,
               power = NULL,
               alpha = 0.05,
               ceiling = TRUE,
               verbose = TRUE,
               pretty = FALSE)
```

Arguments

`eta.squared`	(partial) eta-squared for the alternative.
`null.eta.squared`	(partial) eta-squared for the null.
`factor.levels`	integer; number of levels or groups in each factor. For example, for two factors each having two levels or groups use e.g. c(2, 2), for three factors each having two levels (groups) use e.g. c(2, 2, 2).
`k.covariates`	integer; number of covariates in the ANCOVA model.
`n.total`	integer; total sample size
`power`	statistical power, defined as the probability of correctly rejecting a false null hypothesis, denoted as $1 - \beta$.
`alpha`	type 1 error rate, defined as the probability of incorrectly rejecting a true null hypothesis, denoted as α.
`ceiling`	logical; if `FALSE` sample size in each cell is not rounded up.
`verbose`	logical; if `FALSE` no output is printed on the console.
`pretty`	logical; whether the output should show Unicode characters (if encoding allows for it). `FALSE` by default.

Value

`parms`	list of parameters used in calculation.
`test`	type of the statistical test (F-Test).
`df1`	numerator degrees of freedom.
`df2`	denominator degrees of freedom.
`ncp`	non-centrality parameter for the alternative.
`null.ncp`	non-centrality parameter for the null.
`f.alpha`	critical value.
`power`	statistical power $(1 - \beta)$.
`n.total`	total sample size.

References

Bulus, M., & Polat, C. (2023). pwrss R paketi ile istatistiksel guc analizi [Statistical power analysis with pwrss R package]. Ahi Evran Universitesi Kirsehir Egitim Fakultesi Dergisi, 24(3), 2207-2328. doi: 10.29299/kefad.1209913

Cohen, J. (1988). Statistical power analysis for the behavioral sciences (2nd ed.). Lawrence Erlbaum Associates.

Examples

```
############################################
#              one-way ANOVA               #
############################################

# Cohen's d = 0.50 between treatment and control
# translating into Eta-squared = 0.059

# estimate sample size using ANOVA approach
power.f.ancova(eta.squared = 0.059,
               factor.levels = 2,
               alpha = 0.05, power = .80)

# estimate sample size using regression approach(F-Test)
power.f.regression(r.squared = 0.059,
                   k.total = 1,
                   alpha = 0.05, power = 0.80)

# estimate sample size using regression approach (T-Test)
p <- 0.50 # proportion of sample in treatment (allocation rate)
power.t.regression(beta = 0.50, r.squared = 0,
                   k.total = 1,
                   sd.predictor = sqrt(p*(1-p)),
                   alpha = 0.05, power = 0.80)

# estimate sample size using t test approach
power.t.student(d = 0.50, alpha = 0.05, power = 0.80)

############################################
#              two-way ANOVA               #
############################################

# a researcher is expecting a partial Eta-squared = 0.03
# for interaction of treatment (Factor A) with
# gender consisting of two levels (Factor B)

power.f.ancova(eta.squared = 0.03,
               factor.levels = c(2,2),
               alpha = 0.05, power = 0.80)

# estimate sample size using regression approach (F test)
# one dummy for treatment, one dummy for gender, and their interaction (k = 3)
# partial Eta-squared is equivalent to the increase in R-squared by adding
```

```
# only the interaction term (m = 1)
power.f.regression(r.squared = 0.03,
                   k.total = 3, k.test = 1,
                   alpha = 0.05, power = 0.80)

##########################################
#              one-way ANCOVA            #
##########################################

# a researcher is expecting an adjusted difference of
# Cohen's d = 0.45 between treatment and control after
# controllling for the pretest (k.cov = 1)
# translating into Eta-squared = 0.048

power.f.ancova(eta.squared = 0.048,
               factor.levels = 2,
               k.covariates = 1,
               alpha = 0.05, power = .80)

##########################################
#              two-way ANCOVA            #
##########################################

# a researcher is expecting an adjusted partial Eta-squared = 0.02
# for interaction of treatment (Factor A) with
# gender consisting of two levels (Factor B)

power.f.ancova(eta.squared = 0.02,
               factor.levels = c(2,2),
               k.covariates = 1,
               alpha = 0.05, power = .80)
```

`power.f.ancova.keppel` *Power Analysis for One-Way ANOVA/ANCOVA Using Means and Standard Deviations (F test)*

Description

Calculates power or sample size for one-way ANOVA/ANCOVA. Set `k.cov = 0` for one-way ANOVA (without any pretest or covariate adjustment). Set `k.cov > 0` in combination with `r2 > 0` for one-way ANCOVA (with pretest or covariate adjustment).

Note that R has a partial matching feature which allows you to specify shortened versions of arguments, such as `mu` or `mu.vec` instead of `mu.vector`, or such as `k` or `k.cov` instead of `k.covariates`.

Formulas are validated using PASS documentation.

Usage

```
power.f.ancova.keppel(mu.vector, sd.vector,
                      n.vector = NULL, p.vector = NULL,
```

```
factor.levels = length(mu.vector),
r.squared = 0, k.covariates = 0,
power = NULL, alpha = 0.05,
ceiling = TRUE, verbose = TRUE,
pretty = FALSE)
```

Arguments

mu.vector	vector of adjusted means (or estimated marginal means) for each level of a factor.
sd.vector	vector of unadjusted standard deviations for each level of a factor.
n.vector	vector of sample sizes for each level of a factor.
p.vector	vector of proportion of total sample size in each level of a factor. These proportions should sum to one.
factor.levels	integer; number of levels or groups in each factor. For example, for two factors each having two levels or groups use e.g. c(2, 2), for three factors each having two levels or groups use e.g. c(2, 2, 2)
r.squared	explanatory power of covariates (R-squared) in the ANCOVA model. The default is r.squared = 0, which means an ANOVA model would be of interest.
k.covariates	integer; number of covariates in the ANCOVA model. The default is k.cov = 0, which means an ANOVA model would be of interest.
power	statistical power, defined as the probability of correctly rejecting a false null hypothesis, denoted as $1 - \beta$.
alpha	type 1 error rate, defined as the probability of incorrectly rejecting a true null hypothesis, denoted as α.
ceiling	logical; whether sample size should be rounded up. TRUE by default.
verbose	logical; whether the output should be printed on the console. TRUE by default.
pretty	logical; whether the output should show Unicode characters (if encoding allows for it). FALSE by default.

Value

parms	list of parameters used in calculation.
test	type of the statistical test (F-Test).
df1	numerator degrees of freedom.
df2	denominator degrees of freedom.
ncp	non-centrality parameter under alternative.
null.ncp	non-centrality parameter under null.
power	statistical power $(1 - \beta)$.
n.total	total sample size.

References

Keppel, G., & Wickens, T. D. (2004). Design and analysis: A researcher's handbook (4th ed.). Pearson.

Examples

```
# required sample size to detect a mean difference of
# Cohen's d = 0.50 for a one-way two-group design
power.f.ancova.keppel(mu.vector = c(0.50, 0), # marginal means
                      sd.vector = c(1, 1), # unadjusted standard deviations
                      n.vector = NULL, # sample size (will be calculated)
                      p.vector = c(0.50, 0.50), # balanced allocation
                      k.cov = 1, # number of covariates
                      r.squared = 0.50, # explanatory power of covariates
                      alpha = 0.05, # Type 1 error rate
                      power = .80)

# effect size approach
power.f.ancova(eta.squared = 0.111, # effect size that is already adjusted for covariates
               factor.levels = 2, # one-way ANCOVA with two levels (groups)
               k.covariates = 1, # number of covariates
               alpha = 0.05, # Type 1 error rate
               power = .80)

# regression approach
p <- 0.50
power.t.regression(beta = 0.50,
                   sd.predictor = sqrt(p * (1 - p)),
                   sd.outcome = 1,
                   k.total = 1,
                   r.squared = 0.50,
                   n = NULL, power = 0.80)
```

`power.f.ancova.shieh` *Power Analysis for One-, Two-, Three-Way ANCOVA Using Means, Standard Deviations, and (Optionally) Contrasts (F test)*

Description

Calculates power or sample size for one-, two-, three-way ANCOVA. For factorial designs, use the argument `factor.levels` but note that unique combination of levels (cells in this case) should follow a specific order for the test of interaction. The order of marginal means and standard deviations is printed as a warning message.

Note that R has a partial matching feature which allows you to specify shortened versions of arguments, such as `mu` or `mu.vec` instead of `mu.vector`, or such as `k` or `k.cov` instead of `k.covariates`.

Formulas are validated using examples and tables in Shieh (2020).

Usage

```
power.f.ancova.shieh(mu.vector, sd.vector,
                     n.vector = NULL, p.vector = NULL,
                     factor.levels = length(mu.vector),
                     r.squared = 0, k.covariates = 1,
```

```
                    contrast.matrix = NULL,
                    power = NULL, alpha = 0.05,
                    ceiling = TRUE, verbose = TRUE,
                    pretty = FALSE)
```

Arguments

mu.vector	vector; adjusted means (or estimated marginal means) for each level of a factor.
sd.vector	vector; unadjusted standard deviations for each level of a factor. If a pooled standard deviation is provided, repeat its value to match the number of group means. A warning will be issued if group standard deviations differ substantially beyond what is expected due to sampling error.
n.vector	vector; sample sizes for each level of a factor.
p.vector	vector; proportion of total sample size in each level of a factor. These proportions should sum to one.
factor.levels	integer; number of levels or groups in each factor. For example, for two factors each having two levels or groups use e.g. c(2, 2), for three factors each having two levels or groups use e.g. c(2, 2, 2).
r.squared	explanatory power of covariates (R-squared) in the ANCOVA model.
k.covariates	integer; number of covariates in the ANCOVA model.
contrast.matrix	vector or matrix; contrasts should not be confused with the model (design) matrix. Rows of contrast matrix indicate independent vector of contrasts summing to zero. The default contrast matrix is constructed using deviation coding scheme (a.k.a. effect coding). Columns in the contrast matrix indicate number of levels or groups (or cells in factorial designs).
power	statistical power, defined as the probability of correctly rejecting a false null hypothesis, denoted as $1 - \beta$.
alpha	type 1 error rate, defined as the probability of incorrectly rejecting a true null hypothesis, denoted as α.
ceiling	logical; TRUE by default. If FALSE sample size in each cell is NOT rounded up.
verbose	logical; TRUE by default. If FALSE no output is printed on the console.
pretty	logical; whether the output should show Unicode characters (if encoding allows for it). FALSE by default.

Value

parms	list of parameters used in calculation.
test	type of the statistical test (F-Test)
eta.squared	(partial) eta-squared.
f	Cohen's f statistic.
df1	numerator degrees of freedom.
df2	denominator degrees of freedom.
ncp	non-centrality parameter for the alternative.

null.ncp	non-centrality parameter for the null.
power	statistical power $(1 - \beta)$.
n.total	total sample size.

References

Shieh, G. (2020). Power analysis and sample size planning in ANCOVA designs. Psychometrika, 85(1), 101-120. doi: 10.1007/s11336019096923

Examples

```
######################################################################
#########################  main effect  ##############################
######################################################################

# power for one-way ANCOVA (two levels or groups)
power.f.ancova.shieh(mu.vector = c(0.20, 0), # marginal means
                     sd.vector = c(1, 1), # unadjusted standard deviations
                     n.vector = c(150, 150), # sample sizes
                     r.squared = 0.50, # proportion of variance explained by covariates
                     k.covariates = 1, # number of covariates
                     alpha = 0.05)


# sample size for one-way ANCOVA (two levels or groups)
power.f.ancova.shieh(mu.vector = c(0.20, 0), # marginal means
                     sd.vector = c(1, 1), # unadjusted standard deviations
                     p.vector = c(0.50, 0.50), # allocation, should sum to 1
                     r.squared = 0.50,
                     k.covariates = 1,
                     alpha = 0.05,
                     power = 0.80)

######################################################################
#######################  interaction effect  #########################
######################################################################

# sample size for two-way ANCOVA (2 x 2)
power.f.ancova.shieh(mu.vector = c(0.20, 0.25, 0.15, 0.05), # marginal means
                     sd.vector = c(1, 1, 1, 1), # unadjusted standard deviations
                     p.vector = c(0.25, 0.25, 0.25, 0.25), # allocation, should sum to 1
                     factor.levels = c(2, 2), # 2 by 2 factorial design
                     r.squared = 0.50,
                     k.covariates = 1,
                     alpha = 0.05,
                     power = 0.80)
# Elements of `mu.vector`, `sd.vector`, `n.vector` or `p.vector` should follow this specific order:
#  A1:B1  A1:B2  A2:B1  A2:B2

######################################################################
#######################  planned contrasts  ##########################
######################################################################
```

```
#########################
## dummy coding scheme ##
#########################

contrast.object <- factorial.contrasts(factor.levels = 3, # one factor w/ 3 levels
                                       coding = "treatment") # use dummy coding scheme

# get contrast matrix from the contrast object
contrast.matrix <- contrast.object$contrast.matrix

# calculate sample size given design characteristics
ancova.design <- power.f.ancova.shieh(mu.vector = c(0.15, 0.30, 0.20), # marginal means
                                sd.vector = c(1, 1, 1), # unadjusted standard deviations
                               p.vector = c(1/3, 1/3, 1/3), # allocation, should sum to 1
                                      contrast.matrix = contrast.matrix,
                                      r.squared = 0.50,
                                      k.covariates = 1,
                                      alpha = 0.05,
                                      power = 0.80)

# power of planned contrasts, adjusted for alpha level
power.t.contrasts(ancova.design, adjust.alpha = "fdr")

##########################
## Helmert coding scheme ##
##########################

contrast.object <- factorial.contrasts(factor.levels = 3, # one factor w/ 4 levels
                                       coding = "helmert") # use helmert coding scheme

# get contrast matrix from the contrast object
contrast.matrix <- contrast.object$contrast.matrix

# calculate sample size given design characteristics
ancova.design <- power.f.ancova.shieh(mu.vector = c(0.15, 0.30, 0.20), # marginal means
                                sd.vector = c(1, 1, 1), # unadjusted standard deviations
                               p.vector = c(1/3, 1/3, 1/3), # allocation, should sum to 1
                                      contrast.matrix = contrast.matrix,
                                      r.squared = 0.50,
                                      k.covariates = 1,
                                      alpha = 0.05,
                                      power = 0.80)

# power of planned contrasts
power.t.contrasts(ancova.design)

############################
## polynomial coding scheme ##
############################

contrast.object <- factorial.contrasts(factor.levels = 3, # one factor w/ 4 levels
                                       coding = "poly") # use polynomial coding scheme
```

```
# get contrast matrix from the contrast object
contrast.matrix <- contrast.object$contrast.matrix

# calculate sample size given design characteristics
ancova.design <- power.f.ancova.shieh(mu.vector = c(0.15, 0.30, 0.20), # marginal means
                                sd.vector = c(1, 1, 1), # unadjusted standard deviations
                               p.vector = c(1/3, 1/3, 1/3), # allocation, should sum to 1
                                     contrast.matrix = contrast.matrix,
                                     r.squared = 0.50,
                                     k.covariates = 1,
                                     alpha = 0.05,
                                     power = 0.80)

# power of the planned contrasts
power.t.contrasts(ancova.design)

######################
## custom contrasts ##
######################

# custom contrasts
contrast.matrix <- rbind(
  cbind(A1 = 1, A2 = -0.50, A3 = -0.50),
  cbind(A1 = 0.50, A2 = 0.50, A3 = -1)
)
# labels are not required for custom contrasts,
# but they make it easier to understand power.t.contrasts() output

# calculate sample size given design characteristics
ancova.design <- power.f.ancova.shieh(mu.vector = c(0.15, 0.30, 0.20), # marginal means
                                sd.vector = c(1, 1, 1), # unadjusted standard deviations
                               p.vector = c(1/3, 1/3, 1/3), # allocation, should sum to 1
                                     contrast.matrix = contrast.matrix,
                                     r.squared = 0.50,
                                     k.covariates = 1,
                                     alpha = 0.05,
                                     power = 0.80)

# power of the planned contrasts
power.t.contrasts(ancova.design)
```

`power.f.mixed.anova` *Power Analysis for Mixed-Effects Analysis of Variance (F-Test)*

Description

Calculates power or sample size for mixed-effects ANOVA design with two factors (between and within). When there is only one group observed over time, this design is often referred to as repeated-measures ANOVA.

Formulas are validated using G*Power and tables in PASS documentation.

NOTE: The `pwrss.f.rmanova()` function is deprecated and will no longer be supported, but it will remain available as a wrapper for `power.f.mixed.anova()` during the transition period.

Usage

```
power.f.mixed.anova(eta.squared, null.eta.squared = 0,
                    factor.levels = c(2, 2),
                    factor.type = c("between", "within"),
                    rho.within = 0.50, epsilon = 1,
                    n.total = NULL, power = NULL, alpha = 0.05,
                    effect = c("between", "within", "interaction"),
                    ceiling = TRUE, verbose = TRUE, pretty = FALSE)
```

Arguments

`eta.squared`	(partial) eta-squared for the alternative.
`null.eta.squared`	(partial) eta-squared for the null.
`rho.within`	Correlation between repeated measures. For example, for pretest/post-test designs, this is the correlation between pretest and post-test scores regardless of group membership. The default is 0.50. If `eta.squared` is already adjusted for this correlation specify 'rho.within = NA'.
`factor.levels`	vector; integer; length of two representing the number of levels for groups and measures. For example, in randomized controlled trials with two arms (treatment and control) where pre-test, post-test, and follow-up test are administered, this would be represented as c(2, 3).
`factor.type`	vector; character; length of two indicating the order of between-subject and within-subject factors. By default, the first value represents the between-subject factor and the second value represents the within-subject factor. This argument is rarely needed, except when unsure which element in 'factor.levels' represent between-subject or within-subject factors. Therefore, specify the 'factor.levels' accordingly.
`epsilon`	non-sphericity correction factor, default is 1 (means no violation of sphericity). Lower bound for this argument is `epsilon` = 1 / (`factor.levels[2]` - 1).
`n.total`	integer; total sample size.
`power`	statistical power, defined as the probability of correctly rejecting a false null hypothesis, denoted as $1 - \beta$.
`alpha`	type 1 error rate, defined as the probability of incorrectly rejecting a true null hypothesis, denoted as α.
`effect`	character; the effect of interest: "between", "within", or "interaction".
`ceiling`	logical; `TRUE` by default. If `FALSE` sample size in each group is NOT rounded up.
`verbose`	logical; `TRUE` by default. If `FALSE` no output is printed on the console.
`pretty`	logical; whether the output should show Unicode characters (if encoding allows for it). `FALSE` by default.

Value

parms	list of parameters used in calculation.
test	type of the statistical test (F-Test).
df1	numerator degrees of freedom.
df2	denominator degrees of freedom.
ncp	non-centrality parameter under alternative.
null.ncp	non-centrality parameter under null.
power	statistical power $(1 - \beta)$.
n.total	total sample size.

References

Bulus, M., & Polat, C. (2023). pwrss R paketi ile istatistiksel guc analizi [Statistical power analysis with pwrss R package]. Ahi Evran Universitesi Kirsehir Egitim Fakultesi Dergisi, 24(3), 2207-2328. doi: 10.29299/kefad.1209913

Examples

```
#######################################################
# pretest-post-test design with treatment group only  #
#######################################################

# a researcher is expecting a difference of Cohen's d = 0.30
# between post-test and pretest score translating into
# Eta-squared = 0.022

# adjust effect size for correlation with 'rho.within'
power.f.mixed.anova(eta.squared = 0.022,
                    factor.levels = c(1, 2), # 1 between 2 within
                    rho.within = 0.50,
                    effect = "within",
                    alpha = 0.05, power = 0.80)

# if effect size is already adjusted for correlation
# use 'rho.within = NA'
power.f.mixed.anova(eta.squared = 0.08255,
                    factor.levels = c(1, 2), # 1 between 2 within
                    rho.within = NA,
                    effect = "within",
                    alpha = 0.05, power = 0.80)

#########################################################
# post-test only design with treatment and control groups #
#########################################################

# a researcher is expecting a difference of Cohen's d = 0.50
# on the post-test score between treatment and control groups
# translating into Eta-squared = 0.059
power.f.mixed.anova(eta.squared = 0.059,
```

```
                    factor.levels = c(2, 1),  # 2 between 1 within
                    effect = "between",
                    alpha = 0.05, power = 0.80)


############################################################
# pretest-post-test design with treatment and control groups #
############################################################

# a researcher is expecting a difference of Cohen's d = 0.40
# on the post-test score between treatment and control groups
# after controlling for the pretest translating into
# partial Eta-squared = 0.038
power.f.mixed.anova(eta.squared = 0.038,
                    factor.levels = c(2, 2),  # 2 between 2 within
                    rho.within = 0.50,
                    effect = "between",
                    alpha = 0.05, power = 0.80)

# a researcher is expecting an interaction effect
# (between groups and time) of Eta-squared = 0.01
power.f.mixed.anova(eta.squared = 0.01,
                    factor.levels = c(2, 2),  # 2 between 2 within
                    rho.within = 0.50,
                    effect = "interaction",
                    alpha = 0.05, power = 0.80)

# a researcher is expecting an interaction effect
# (between groups and time) of Eta-squared = 0.01
power.f.mixed.anova(eta.squared = 0.01,
                    factor.levels = c(2, 2),  # 2 between 2 within
                    rho.within = 0.50,
                    effect = "within",
                    alpha = 0.05, power = 0.80)
```

`power.f.regression` *Power Analysis for Linear Regression: R-squared or R-squared Change (F-Test)*

Description

Calculates power or sample size (only one can be NULL at a time) to test R-squared deviation from 0 (zero) in linear regression or to test R-squared change between two linear regression models. The test of R-squared change is often used to evaluate incremental contribution of a set of predictors in hierarchical linear regression.

Formulas are validated using Monte Carlo simulation, G*Power, and tables in PASS documentation.

NOTE: The `pwrss.f.reg()` function and its alias `pwrss.f.regression` are deprecated, but they will remain available as a wrapper for `power.f.regression()` during the transition period.

Usage

```
power.f.regression(r.squared.change = NULL, margin = 0,
                   k.total, k.tested = k.total,
                   n = NULL, power = NULL, alpha = 0.05,
                   ceiling = TRUE, verbose = TRUE, pretty = FALSE)
```

Arguments

r.squared.change	R-squared (or R-squared change).
margin	margin - ignorable R-squared (or R-squared change).
k.total	integer; total number of predictors.
k.tested	integer; number of predictors in the subset of interest. By default m.tested = k.total, which implies that one is interested in the contribution of all predictors, and tests whether R-squared value is different from 0 (zero).
n	integer; sample size.
power	statistical power, defined as the probability of correctly rejecting a false null hypothesis, denoted as $1 - \beta$.
alpha	type 1 error rate, defined as the probability of incorrectly rejecting a true null hypothesis, denoted as α.
ceiling	logical; whether sample size should be rounded up. TRUE by default.
verbose	logical; whether the output should be printed on the console. TRUE by default.
pretty	logical; whether the output should show Unicode characters (if encoding allows for it). FALSE by default.

Value

parms	list of parameters used in calculation.
test	type of the statistical test (F-Test).
df1	numerator degrees of freedom.
df2	denominator degrees of freedom.
ncp	non-centrality parameter for the alternative.
null.ncp	non-centrality parameter for the null.
f.alpha	critical value.
power	statistical power $(1 - \beta)$.
n	sample size.

References

Bulus, M., & Polat, C. (2023). pwrss R paketi ile istatistiksel guc analizi [Statistical power analysis with pwrss R package]. Ahi Evran Universitesi Kirsehir Egitim Fakultesi Dergisi, 24(3), 2207-2328. doi: 10.29299/kefad.1209913

Cohen, J. (1988). Statistical power analysis for the behavioral sciences (2nd ed.). Lawrence Erlbaum Associates.

Examples

```
# in the outcome (R-squared = 0.15).
power.f.regression(r.squared = 0.15,
                   k.total = 3, # total number of predictors
                   power = 0.80)

# adding two more variables will increase R-squared
# from 0.15 (with 3 predictors) to 0.25 (with 3 + 2 predictors)
power.f.regression(r.squared.change = 0.10, # R-squared change
                   k.total = 5, # total number of predictors
                   k.tested = 2, # predictors to be tested
                   power = 0.80)
```

`power.np.wilcoxon` *Power Analysis for Non-parametric Rank-Based Tests (One-Sample, Independent, and Paired Designs)*

Description

Calculates power or sample size (only one can be NULL at a time) for non-parametric rank-based tests. The following tests and designs are available:

- Wilcoxon Signed-Rank Test (One Sample)
- Wilcoxon Rank-Sum or Mann-Whitney U Test (Independent Samples)
- Wilcoxon Matched-Pairs Signed-Rank Test (Paired Samples)

Use `means.to.d()` to convert raw means and standard deviations to Cohen's d, and `d.to.cles()` to convert Cohen's d to the probability of superiority. Note that this interpretation is appropriate only when the underlying distribution is approximately normal and the two groups have similar population variances.

Formulas are validated using G*Power and tables in PASS documentation. However, we adopt rounding convention used by G*Power.

Note that R has a partial matching feature which allows you to specify shortened versions of arguments, such as `alt` instead of `alternative`, or `dist` instead of `distribution`.

NOTE: `pwrss.np.2means()` function is no longer supported. `pwrss.np.2groups()` will remain available for some time.

Usage

```
power.np.wilcoxon(d, null.d = 0, margin = 0,
                  n2 = NULL, n.ratio = 1, power = NULL, alpha = 0.05,
                  alternative = c("two.sided", "one.sided", "two.one.sided"),
                  design = c("independent", "paired", "one.sample"),
                  distribution = c("normal", "uniform", "double.exponential",
                                   "laplace", "logistic"),
                  method = c("guenther", "noether"),
                  ceiling = TRUE, verbose = TRUE, pretty = FALSE)
```

Arguments

d	Cohen's d or Hedges' g.
null.d	Cohen's d or Hedges' g under null, typically 0 (zero).
margin	margin - ignorable d - null.d difference.
n2	integer; sample size in the second group (or for the single group in paired samples or one-sample)
n.ratio	n1/n2 ratio (applies to independent samples only)
power	statistical power, defined as the probability of correctly rejecting a false null hypothesis, denoted as $1 - \beta$.
alpha	type 1 error rate, defined as the probability of incorrectly rejecting a true null hypothesis, denoted as α.
design	character; "independent" (default), "one.sample", or "paired".
alternative	character; direction or type of the hypothesis test: "two.sided", "one.sided", or "two.one.sided".
distribution	character; parent distribution: "normal", "uniform", "double.exponential", "laplace", or "logistic".
method	character; non-parametric approach: "guenther" (default) or "noether"
ceiling	logical; whether sample size should be rounded up. TRUE by default.
verbose	logical; whether the output should be printed on the console. TRUE by default.
pretty	logical; whether the output should show Unicode characters (if encoding allows for it). FALSE by default.

Value

parms	list of parameters used in calculation.
test	type of the statistical test (Z- or T-Test).
df	degrees of freedom (applies when method = 'guenther').
ncp	non-centrality parameter for the alternative (applies when method = 'guenther').
null.ncp	non-centrality parameter for the null (applies when method = 'guenther').
t.alpha	critical value(s) (applies when method = 'guenther').
mean	mean of the alternative (applies when method = 'noether').
null.mean	mean of the null (applies when method = 'noether').
sd	standard deviation of the alternative (applies when method = 'noether').
null.sd	standard deviation of the null (applies when method = 'noether').
z.alpha	critical value(s) (applies when method = 'noether').
power	statistical power $(1 - \beta)$.
n	sample size ('n' or 'c(n1, n2)' depending on the design.

References

Al-Sunduqchi, M. S. (1990). Determining the appropriate sample size for inferences based on the Wilcoxon statistics [Unpublished doctoral dissertation]. University of Wyoming - Laramie

Chow, S. C., Shao, J., Wang, H., and Lokhnygina, Y. (2018). Sample size calculations in clinical research (3rd ed.). Taylor & Francis/CRC.

Lehmann, E. (1975). Nonparameterics: Statistical methods based on ranks. McGraw-Hill.

Noether, G. E. (1987). Sample size determination for some common nonparametric tests. Journal of the American Statistical Association, 82(1), 645-647.

Ruscio, J. (2008). A probability-based measure of effect size: Robustness to base rates and other factors. Psychological Methods, 13(1), 19-30.

Ruscio, J., & Mullen, T. (2012). Confidence intervals for the probability of superiority effect size measure and the area under a receiver operating characteristic curve. Multivariate Behavioral Research, 47(2), 201-223.

Zhao, Y.D., Rahardja, D., & Qu, Y. (2008). Sample size calculation for the Wilcoxon-Mann-Whitney test adjusting for ties. Statistics in Medicine, 27(3), 462-468.

Examples

```
# Mann-Whitney U or Wilcoxon rank-sum test
# (a.k.a Wilcoxon-Mann-Whitney test) for independent samples

## difference between group 1 and group 2 is not equal to zero
## estimated difference is Cohen'd = 0.25
power.np.wilcoxon(d = 0.25,
                  power = 0.80)

## difference between group 1 and group 2 is greater than zero
## estimated difference is Cohen'd = 0.25
power.np.wilcoxon(d = 0.25,
                  power = 0.80,
                  alternative = "one.sided")

## mean of group 1 is practically not smaller than mean of group 2
## estimated difference is Cohen'd = 0.10 and can be as small as -0.05
power.np.wilcoxon(d = 0.10,
                  margin = -0.05,
                  power = 0.80,
                  alternative = "one.sided")

## mean of group 1 is practically greater than mean of group 2
## estimated difference is Cohen'd = 0.10 and can be as small as 0.05
power.np.wilcoxon(d = 0.10,
                  margin = 0.05,
                  power = 0.80,
                  alternative = "one.sided")

## mean of group 1 is practically same as mean of group 2
## estimated difference is Cohen'd = 0
## and can be as small as -0.05 and as high as 0.05
```

```
power.np.wilcoxon(d = 0,
                margin = c(-0.05, 0.05),
                power = 0.80,
                alternative = "two.one.sided")


# Wilcoxon signed-rank test for matched pairs (dependent samples)

## difference between time 1 and time 2 is not equal to zero
## estimated difference between time 1 and time 2 is Cohen'd = -0.25
power.np.wilcoxon(d = -0.25,
                power = 0.80,
                design = "paired")

## difference between time 1 and time 2 is greater than zero
## estimated difference between time 1 and time 2 is Cohen'd = -0.25
power.np.wilcoxon(d = -0.25,
                power = 0.80,
                design = "paired",
                alternative = "one.sided")

## mean of time 1 is practically not smaller than mean of time 2
## estimated difference is Cohen'd = -0.10 and can be as small as 0.05
power.np.wilcoxon(d = -0.10,
                margin = 0.05,
                power = 0.80,
                design = "paired",
                alternative = "one.sided")

## mean of time 1 is practically greater than mean of time 2
## estimated difference is Cohen'd = -0.10 and can be as small as -0.05
power.np.wilcoxon(d = -0.10,
                margin = -0.05,
                power = 0.80,
                design = "paired",
                alternative = "one.sided")

## mean of time 1 is practically same as mean of time 2
## estimated difference is Cohen'd = 0
## and can be as small as -0.05 and as high as 0.05
power.np.wilcoxon(d = 0,
                margin = c(-0.05, 0.05),
                power = 0.80,
                design = "paired",
                alternative = "two.one.sided")
```

`power.t` *Statistical Power for the Generic T-Test*

Description

Calculates power for the generic T-Test with (optional) Type 1 and Type 2 error plots.

Usage

```
power.t.test(ncp, null.ncp = 0, df, alpha = 0.05,
             alternative = c("two.sided", "one.sided", "two.one.sided"),
             plot = TRUE, verbose = TRUE, pretty = FALSE)
```

Arguments

ncp	non-centrality parameter for the alternative.
null.ncp	non-centrality parameter for the null. When alternative = "two.one.sided", the function expects two values in the form c(lower, upper). If a single value is provided, it is interpreted as the absolute bound and automatically expanded to c(-value, +value).
df	degrees of freedom.
alpha	type 1 error rate, defined as the probability of incorrectly rejecting a true null hypothesis, denoted as α.
alternative	character; direction or type of the hypothesis test: "one.sided", "two.sided", or "two.one.sided". "two.one.sided" is used for equivalence and minimal effect testing.
plot	logical; FALSE switches off Type 1 and Type 2 error plot. TRUE by default.
verbose	logical; whether the output should be printed on the console. TRUE by default.
pretty	logical; whether the output should show Unicode characters (if encoding allows for it). FALSE by default.

Value

df	degrees of freedom.
ncp	non-centrality parameter under alternative.
ncp.null	non-centrality parameter under null.
t.alpha	critical value(s).
power	statistical power $(1 - \beta)$.

Examples

```
# two-sided
# power defined as the probability of observing a test statistic
# greater than the positive critical value OR
# less than the negative critical value
power.t.test(ncp = 1.96, df = 100, alpha = 0.05,
             alternative = "two.sided")

# one-sided
# power is defined as the probability of observing a test statistic
# greater than the critical value
power.t.test(ncp = 1.96, df = 100, alpha = 0.05,
             alternative = "one.sided")
```

```
# equivalence
# power is defined as the probability of observing a test statistic
# greater than the upper critical value (for the lower bound) AND
# less than the lower critical value (for the upper bound)
power.t.test(ncp = 0, df = 100,
             null.ncp = c(-2, 2), alpha = 0.05,
             alternative = "two.one.sided")

# minimal effect testing
# power is defined as the probability of observing a test statistic
# greater than the upper critical value (for the upper bound) OR
# less than the lower critical value (for the lower bound).
power.t.test(ncp = 2, df = 100,
             null.ncp = c(-1, 1), alpha = 0.05,
             alternative = "two.one.sided")
```

`power.t.regression` *Power Analysis for Linear Regression: Single Coefficient (T-Test)*

Description

Calculates power or sample size (only one can be NULL at a time) to test a single coefficient in multiple linear regression. The predictor is assumed to be continuous by default. However, one can calculate power or sample size for a binary predictor (such as treatment and control groups in an experimental design) by specifying `sd.predictor = sqrt(p*(1-p))` where `p` is the proportion of subjects in one of the groups. The sample size in each group would be `n*p` and `n*(1-p)`. `power.t.regression()pwrss.t.regression()` are the same functions, as well as `power.t.reg()` and `pwrss.t.reg()`.

Minimal effect and equivalence tests are implemented in line with Hodges and Lehmann (1954), Kim and Robinson (2019), Phillips (1990), and Dupont and Plummer (1998).

Formulas are validated using Monte Carlo simulation, G*Power, tables in PASS documentation, and tables in Bulus (2021).

NOTE: The `pwrss.t.regression()` function and its alias `pwrss.z.reg()` are deprecated, but they will remain available as a wrapper for `power.t.regression()` during the transition period.

Usage

```
power.t.regression(beta, null.beta = 0, margin = 0,
                   sd.predictor = 1, sd.outcome = 1,
                   r.squared = (beta * sd.predictor / sd.outcome)^2,
                   k.total = 1, n = NULL, power = NULL, alpha = 0.05,
                   alternative = c("two.sided", "one.sided", "two.one.sided"),
                   ceiling = TRUE, verbose = TRUE, pretty = FALSE)
```

Arguments

beta	regression coefficient. One can use standardized regression coefficient, but should keep sd.predictor = 1 and sd.outcome = 1 or leave them out as they are default specifications.
null.beta	regression coefficient under null hypothesis (typically zero). One can use standardized regression coefficient, but should keep sd.predictor = 1 and sd.outcome = 1 or leave them out as they are default specifications.
margin	margin - ignorable beta - null.beta difference.
sd.predictor	standard deviation of the predictor. For a binary predictor, sd.predictor = sqrt(p * (1 - p)) where p is the proportion of subjects in one of the groups.
sd.outcome	standard deviation of the outcome.
k.total	integer; total number of predictors, including the predictor of interest.
r.squared	model R-squared. The default is r.squared = (beta * sd.predictor / sd.outcome)^2 assuming a linear regression with one predictor. Thus, an r.squared below this value will throw a warning. To consider other covariates in the model provide a value greater than the default r.squared along with the argument k.total > 1.
n	integer; sample size.
power	statistical power, defined as the probability of correctly rejecting a false null hypothesis, denoted as $1 - \beta$.
alpha	type 1 error rate, defined as the probability of incorrectly rejecting a true null hypothesis, denoted as α.
alternative	character; the direction or type of the hypothesis test: "two.sided", "one.sided", or "two.one.sided".
ceiling	logical; whether sample size should be rounded up. TRUE by default.
verbose	logical; whether the output should be printed on the console. TRUE by default.
pretty	logical; whether the output should show Unicode characters (if encoding allows for it). FALSE by default.

Value

parms	list of parameters used in calculation.
test	type of the statistical test (T-Test).
df	degrees of freedom.
ncp	non-centrality parameter for the alternative.
null.ncp	non-centrality parameter for the null.
t.alpha	critical value(s).
power	statistical power $(1 - \beta)$.
n	sample size.

References

Bulus, M. (2021). Sample size determination and optimal design of randomized/non-equivalent pretest-post-test control-group designs. Adiyaman University Journal of Educational Sciences, 11(1), 48-69. doi: 10.17984/adyuebd.941434

Hodges Jr, J. L., & Lehmann, E. L. (1954). Testing the approximate validity of statistical hypotheses. Journal of the Royal Statistical Society Series B: Statistical Methodology, 16(2), 261-268. doi: 10.1111/j.25176161.1954.tb00169.x

Kim, J. H., & Robinson, A. P. (2019). Interval-based hypothesis testing and its applications to economics and finance. Econometrics, 7(2), 21. doi: 10.3390/econometrics7020021

Phillips, K. F. (1990). Power of the two one-sided tests procedure in bioequivalence. Journal of Pharmacokinetics and Biopharmaceutics, 18(2), 137-144. doi: 10.1007/bf01063556

Dupont, W. D., and Plummer, W. D. (1998). Power and sample size calculations for studies involving linear regression. Controlled Clinical Trials, 19(6), 589-601. doi: 10.1016/s01972456(98)00037-3

Examples

```
# continuous predictor x (and 4 covariates)
power.t.regression(beta = 0.20,
            k.total = 5,
            r.squared = 0.30,
            power = 0.80)

# binary predictor x (and 4 covariates)
p <- 0.50 # proportion of subjects in one group
power.t.regression(beta = 0.20,
            sd.predictor = sqrt(p*(1-p)),
            k.total = 5,
            r.squared = 0.30,
            power = 0.80)

# non-inferiority test with binary predictor x (and 4 covariates)
p <- 0.50 # proportion of subjects in one group
power.t.regression(beta = 0.20, # Cohen's d
            margin = -0.05, # non-inferiority margin in Cohen's d unit
            alternative = "one.sided",
            sd.predictor = sqrt(p*(1-p)),
            k.total = 5,
            r.squared = 0.30,
            power = 0.80)

# superiority test with binary predictor x (and 4 covariates)
p <- 0.50 # proportion of subjects in one group
power.t.regression(beta = 0.20, # Cohen's d
            margin = 0.05, # superiority margin in Cohen's d unit
            alternative = "one.sided",
            sd.predictor = sqrt(p*(1-p)),
            k.total = 5,
            r.squared = 0.30,
            power = 0.80)
```

```
# equivalence test with binary predictor x (and 4 covariates)
p <- 0.50 # proportion of subjects in one group
power.t.regression(beta = 0, # Cohen's d
            margin = c(-0.05, 0.05), # equivalence bounds in Cohen's d unit
            alternative = "two.one.sided",
            sd.predictor = sqrt(p*(1 - p)),
            k.total = 5,
            r.squared = 0.30,
            power = 0.80)
```

`power.t.student` *Power Analysis for Student's and Welch's T-Tests*

Description

Calculates power or sample size (only one can be NULL at a time) for Student's and Welch's T-Tests. Welch's T-Test implementation relies on formulas proposed by Bulus (2024).

Use `means.to.d()` to convert raw means and standard deviations to Cohen's d, and `d.to.cles()` to convert Cohen's d to the probability of superiority. Note that this interpretation is appropriate only when the underlying distribution is approximately normal and the two groups have similar population variances.

In contrast to previous versions, users can now specify whether their claims will be based on raw score mean difference with P-values or standardized mean difference with confidence intervals. While results typically differ by only a few units, these distinctions can be particularly consequential in studies with small sample sizes or high-risk interventions.

Formulas are validated using Monte Carlo simulations (see Bulus, 2024), G*Power, http://powerandsamplesize.com/, and tables in PASS documentation. One key difference between PASS and `pwrss` lies in how they handle non-inferiority and superiority tests-that is, one-sided tests defined by a negligible effect margin (implemented as of this version). PASS shifts the test statistic so that the null hypothesis assumes a zero effect, treating the negligible margin as part of the alternative hypothesis. As a result, the test statistic is evaluated against a central distribution. In contrast, `pwrss` treats the negligible effect as the true null value, and the test statistic is evaluated under a non-central distribution. This leads to slight differences up to third decimal place. To get the same results, reflect the margin in `null.d` and specify `margin = 0`.

Equivalence tests are implemented in line with Bulus and Polat (2023), Chow et al. (2018) and Lakens (2017).

NOTE: The functions `pwrss.z.mean()` and `pwrss.z.2means()` are no longer supported. The `pwrss.t.mean()` and `pwrss.t.2means()` functions are deprecated, but they will remain available as wrappers for `power.t.student()` or `power.t.welch()` during the transition period.

Usage

```
power.t.student(d, null.d = 0, margin = 0,
                n2 = NULL, n.ratio = 1, power = NULL, alpha = 0.05,
                alternative = c("two.sided", "one.sided", "two.one.sided"),
```

```
                  design = c("independent", "paired", "one.sample"),
                  claim.basis = c("md.pval", "smd.ci"),
                  ceiling = TRUE, verbose = TRUE, pretty = FALSE)

power.t.welch(d, null.d = 0, margin = 0,
              var.ratio = 1, n.ratio = 1, n2 = NULL,
              power = NULL, alpha = 0.05,
              alternative = c("two.sided", "one.sided", "two.one.sided"),
              claim.basis = c("md.pval", "smd.ci"),
              ceiling = TRUE, verbose = TRUE, pretty = FALSE)
```

Arguments

d	Cohen's d or Hedges' g.
null.d	Cohen's d or Hedges' g under null, typically 0(zero).
margin	margin - ignorable d - null.d difference.
var.ratio	variance ratio in the form of sd1^2 / sd2^2.
n2	integer; sample size in the second group (or for the single group in paired samples or one-sample).
n.ratio	n1/n2 ratio (applies to independent samples only)
power	statistical power, defined as the probability of correctly rejecting a false null hypothesis, denoted as $1 - \beta$.
alpha	type 1 error rate, defined as the probability of incorrectly rejecting a true null hypothesis, denoted as α.
alternative	character; the direction or type of the hypothesis test: "one.sided", "two.sided", "two.one.sided". For non-inferiority or superiority tests, add or subtract the margin from the null hypothesis value and use alternative = "one.sided".
design	character; "independent", "paired" or "one.sample".
claim.basis	character; "md.pval" when claims are based on raw mean differences and p-values, "smd.ci" when claims are based on standardized mean differences and confidence intervals.
ceiling	logical; whether sample size should be rounded up. TRUE by default.
verbose	logical; whether the output should be printed on the console. TRUE by default.
pretty	logical; whether the output should show Unicode characters (if encoding allows for it). FALSE by default.

Value

parms	list of parameters used in calculation.
test	type of the statistical test (T-Test).
df	degrees of freedom.
ncp	non-centrality parameter for the alternative.
null.ncp	non-centrality parameter for the null.
t.alpha	critical value(s).
power	statistical power $(1 - \beta)$.
n	sample size ('n' or 'c(n1, n2)' depending on the design.

References

Bulus, M. (2024). Robust standard errors and confidence intervals for standardized mean difference [Preprint].doi: 10.31219/osf.io/k6mbs

Bulus, M., & Polat, C. (2023). pwrss R paketi ile istatistiksel guc analizi [Statistical power analysis with pwrss R package]. Ahi Evran Universitesi Kirsehir Egitim Fakultesi Dergisi, 24(3), 2207-2328. doi: 10.29299/kefad.1209913

Chow, S. C., Shao, J., Wang, H., & Lokhnygina, Y. (2018). Sample size calculations in clinical research (3rd ed.). Taylor & Francis/CRC.

Cohen, J. (1988). Statistical power analysis for the behavioral sciences (2nd ed.). Lawrence Erlbaum Associates.

Lakens, D. (2017). Equivalence tests: A practical primer for t tests, correlations, and meta-analyses. Social psychological and personality science, 8(4), 355-362. doi: 10.1177/1948550617697177

Examples

```
#######################
# Independent Samples #
#######################

## difference between group 1 and group 2 is not equal to zero
## targeting minimal difference of Cohen'd = 0.20
## non-parametric
power.np.wilcoxon(d = 0.20,
                  power = 0.80,
                  alternative = "two.sided",
                  design = "independent")

## parametric
power.t.student(d = 0.20,
                power = 0.80,
                alternative = "two.sided",
                design = "independent")

## when sample size ratio and group variances differ
power.t.welch(d = 0.20,
              n.ratio = 2,
              var.ratio = 2,
              power = 0.80,
              alternative = "two.sided")


## difference between group 1 and group 2 is greater than zero
## targeting minimal difference of Cohen'd = 0.20
## non-parametric
power.np.wilcoxon(d = 0.20,
                  power = 0.80,
                  alternative = "one.sided",
                  design = "independent")

## parametric
```

```
power.t.student(d = 0.20,
                power = 0.80,
                alternative = "one.sided",
                design = "independent")

## when sample size ratio and group variances differ
power.t.welch(d = 0.20,
              n.ratio = 2,
              var.ratio = 2,
              power = 0.80,
              alternative = "one.sided")


## mean of group 1 is practically not smaller than mean of group 2
## targeting minimal difference of Cohen'd = 0.20 and can be as small as -0.05
## non-parametric
power.np.wilcoxon(d = 0.20,
                  margin = -0.05,
                  power = 0.80,
                  alternative = "one.sided",
                  design = "independent")

## parametric
power.t.student(d = 0.20,
                margin = -0.05,
                power = 0.80,
                alternative = "one.sided",
                design = "independent")

## when sample size ratio and group variances differ
power.t.welch(d = 0.20,
              margin = -0.05,
              n.ratio = 2,
              var.ratio = 2,
              power = 0.80,
              alternative = "one.sided")


## mean of group 1 is practically greater than mean of group 2
## targeting minimal difference of Cohen'd = 0.20 and can be as small as 0.05
## non-parametric
power.np.wilcoxon(d = 0.20,
                  margin = 0.05,
                  power = 0.80,
                  alternative = "one.sided",
                  design = "independent")

## parametric
power.t.student(d = 0.20,
                margin = 0.05,
                power = 0.80,
                alternative = "one.sided",
                design = "independent")
```

```
## when sample size ratio and group variances differ
power.t.welch(d = 0.20,
              margin = 0.05,
              n.ratio = 2,
              var.ratio = 2,
              power = 0.80,
              alternative = "one.sided")


## mean of group 1 is practically same as mean of group 2
## targeting minimal difference of Cohen'd = 0
## and can be as small as -0.05 or as high as 0.05
## non-parametric
power.np.wilcoxon(d = 0,
                  margin = c(-0.05, 0.05),
                  power = 0.80,
                  alternative = "two.one.sided",
                  design = "independent")

## parametric
power.t.student(d = 0,
                margin = c(-0.05, 0.05),
                power = 0.80,
                alternative = "two.one.sided",
                design = "independent")

## when sample size ratio and group variances differ
power.t.welch(d = 0,
              margin = c(-0.05, 0.05),
              n.ratio = 2,
              var.ratio = 2,
              power = 0.80,
              alternative = "two.one.sided")


##################
# Paired Samples #
##################

## difference between time 1 and time 2 is not equal to zero
## targeting minimal difference of Cohen'd = -0.20
## non-parametric
power.np.wilcoxon(d = -0.20,
                  power = 0.80,
                  alternative = "two.sided",
                  design = "paired")

## parametric
power.t.student(d = -0.20,
                power = 0.80,
                alternative = "two.sided",
                design = "paired")
```

```
## difference between time 1 and time 2 is less than zero
## targeting minimal difference of Cohen'd = -0.20
## non-parametric
power.np.wilcoxon(d = -0.20,
                  power = 0.80,
                  alternative = "one.sided",
                  design = "paired")

## parametric
power.t.student(d = -0.20,
                power = 0.80,
                alternative = "one.sided",
                design = "paired")

## mean of time 1 is practically not greater than mean of time 2
## targeting minimal difference of Cohen'd = -0.20 and can be as small as 0.05
## non-parametric
## non-parametric
power.np.wilcoxon(d = 0.20,
                  margin = 0.05,
                  power = 0.80,
                  alternative = "one.sided",
                  design = "paired")

## parametric
power.t.student(d = 0.20,
                margin = 0.05,
                power = 0.80,
                alternative = "one.sided",
                design = "paired")

## mean of time 1 is practically greater than mean of time 2
## targeting minimal difference of Cohen'd = -0.20 and can be as small as -0.05
## non-parametric
power.np.wilcoxon(d = 0.20,
                  margin = -0.05,
                  power = 0.80,
                  alternative = "one.sided",
                  design = "paired")

## parametric
power.t.student(d = 0.20,
                margin = -0.05,
                power = 0.80,
                alternative = "one.sided",
                design = "paired")


## mean of time 1 is practically same as mean of time 2
## targeting minimal difference of Cohen'd = 0
## and can be as small as -0.05 or as high as 0.05
## non-parametric
```

```
## non-parametric
power.np.wilcoxon(d = 0,
                  margin = c(-0.05, 0.05),
                  power = 0.80,
                  alternative = "two.one.sided",
                  design = "paired")

## parametric
power.t.student(d = 0,
                margin = c(-0.05, 0.05),
                power = 0.80,
                alternative = "two.one.sided",
                design = "paired")
```

---

power.z *Statistical Power for the Generic Z-Test*

Description

Calculates power for the generic Z-Test with (optional) Type 1 and Type 2 error plots.

Usage

```
power.z.test(mean = NULL, sd = 1, null.mean = 0, null.sd = 1,
             alpha = 0.05, alternative = c("two.sided",
                                           "one.sided", "two.one.sided"),
             plot = TRUE, verbose = TRUE, pretty = FALSE, ...)
```

Arguments

mean	mean of the alternative.
sd	standard deviation of the alternative. Do not change this value except when some sort of variance correction is applied (e.g. as in logistic and Poisson regressions).
null.mean	mean of the null. When alternative = "two.one.sided", the function expects two values in the form c(lower, upper). If a single value is provided, it is interpreted as the absolute bound and automatically expanded to c(-value, +value).
null.sd	standard deviation of the null. Do not change this value except when some sort of correction is applied.
alpha	type 1 error rate, defined as the probability of incorrectly rejecting a true null hypothesis, denoted as α.
alternative	character; direction or type of the hypothesis test: "one.sided", "two.sided", or "two.one.sided". "two.one.sided" is used for equivalence and minimal effect testing.
plot	logical; FALSE switches off Type 1 and Type 2 error plot. TRUE by default.
verbose	logical; whether the output should be printed on the console. TRUE by default.
...	legacy inputs will be mapped to their corresponding arguments (silent). e.g. ncp
pretty	logical; whether the output should show Unicode characters (if encoding allows for it). FALSE by default.

Value

mean	mean of the alternative distribution.
sd	standard deviation of the alternative distribution.
null.mean	mean of the null distribution.
null.sd	standard deviation of the null distribution.
z.alpha	critical value(s).
power	statistical power $(1 - \beta)$.

Examples

```
# two-sided
# power defined as the probability of observing z-statistics
# greater than the positive critical t value OR
# less than the negative critical t value
power.z.test(mean = 1.96, alpha = 0.05,
             alternative = "two.sided")

# one-sided
# power is defined as the probability of observing z-statistics
# greater than the critical t value
power.z.test(mean = 1.96, alpha = 0.05,
             alternative = "one.sided")

# equivalence
# power is defined as the probability of observing a test statistic
# greater than the upper critical value (for the lower bound) AND
# less than the lower critical value (for the upper bound)
power.z.test(mean = 0, null.mean = c(-2, 2), alpha = 0.05,
             alternative = "two.one.sided")

# minimal effect testing
# power is defined as the probability of observing a test statistic
# greater than the upper critical value (for the upper bound) OR
# less than the lower critical value (for the lower bound).
power.z.test(mean = 2, null.mean = c(-1, 1), alpha = 0.05,
             alternative = "two.one.sided")
```

---

power.z.logistic *Power Analysis for Logistic Regression Coefficient (Wald's Z-Test)*

Description

Calculates power or sample size (only one can be NULL at a time) to test a single coefficient in logistic regression. `power.z.logistic()` and `power.z.logreg()` are the same functions, as well as `pwrss.z.logistic()` and `pwrss.z.logreg()`.

The distribution of the predictor variable can be one of the following: `c("normal", "poisson", "uniform", "exponential", "binomial", "bernouilli", "lognormal")` for Demidenko (2007)

procedure but only c("normal", "binomial", "bernouilli") for Hsieh et al. (1998) procedure. The default parameters for these distributions are

```
distribution = list(dist = "normal", mean = 0, sd = 1)
distribution = list(dist = "poisson", lambda = 1)
distribution = list(dist = "uniform", min = 0, max = 1)
distribution = list(dist = "exponential", rate = 1)
distribution = list(dist = "binomial", size = 1, prob = 0.50)
distribution = list(dist = "bernoulli", prob = 0.50)
distribution = list(dist = "lognormal", meanlog = 0, sdlog = 1)
```

Parameters defined in list() form can be modified, but element names should be kept the same. It is sufficient to use distribution's name for default parameters (e.g. dist = "normal").

NOTE: The pwrss.z.logistic() and its alias pwrss.z.logreg() are deprecated. However, they will remain available as wrappers for the power.z.logistic() function.

Formulas are validated using G*Power and tables in PASS documentation.

Usage

```
power.z.logistic(prob = NULL, base.prob = NULL,
                 odds.ratio  = (prob/(1-prob))/(base.prob/(1-base.prob)),
                 beta0 = log(base.prob/(1-base.prob)), beta1 = log(odds.ratio),
                 n = NULL, power = NULL, r.squared.predictor = 0,
                 alpha = 0.05, alternative = c("two.sided", "one.sided"),
                 method = c("demidenko(vc)", "demidenko", "hsieh"),
                 distribution = "normal", ceiling = TRUE,
                 verbose = TRUE, pretty = FALSE)
```

Arguments

base.prob	base probability under null hypothesis (probability that an event occurs without the influence of the predictor - or when the value of the predictor is zero).
prob	probability under alternative hypothesis (probability that an event occurs when the value of the predictor is increased from 0 to 1). Warning: This is base probability + incremental increase.
beta0	regression coefficient defined as beta0 = log(base.prob/(1-base.prob))
beta1	regression coefficient for the predictor X defined as beta1 = log((prob/(1-prob))/(base.prob/(1-base.prob)))
odds.ratio	odds ratio defined as odds.ratio = exp(beta1) = (prob/(1-prob))/(base.prob/(1-base.prob))
n	integer; sample size
power	statistical power, defined as the probability of correctly rejecting a false null hypothesis, denoted as $1 - \beta$.
r.squared.predictor	proportion of variance in the predictor accounted for by other covariates. This is not a pseudo R-squared. To compute it, regress the predictor on the covariates and extract the adjusted R-squared from that model.

alpha	type 1 error rate, defined as the probability of incorrectly rejecting a true null hypothesis, denoted as α.
alternative	character; direction or type of the hypothesis test: "not equal", "greater", "less"
method	character; analytic method. "demidenko(vc)" stands for Demidenko (2007) procedure with variance correction; "demidenko" stands for Demidenko (2007) procedure without variance correction; "hsieh" stands for Hsieh et al. (1998) procedure. "demidenko" and "hsieh" methods produce similar results but "demidenko(vc)" is more precise
distribution	character; distribution family. Can be one of the c("noramal", "poisson", "uniform", "exponential", "binomial", "bernouilli", "lognormal") for Demidenko (2007) procedure but only c("normal", "binomial", "bernouilli") for Hsieh et al. (1998) procedure.
ceiling	logical; whether sample size should be rounded up. TRUE by default.
verbose	logical; whether the output should be printed on the console. TRUE by default.
pretty	logical; whether the output should show Unicode characters (if encoding allows for it). FALSE by default.

Value

parms	list of parameters used in calculation.
test	type of the statistical test (Z-Test).
mean	mean of the alternative distribution.
sd	standard deviation of the alternative distribution.
null.mean	mean of the null distribution.
null.sd	standard deviation of the null distribution.
z.alpha	critical value(s).
power	statistical power $(1 - \beta)$.
n	sample size.

References

Demidenko, E. (2007). Sample size determination for logistic regression revisited. Statistics in Medicine, 26(18), 3385-3397. doi: 10.1002/sim.2771

Hsieh, F. Y., Bloch, D. A., & Larsen, M. D. (1998). A simple method of sample size calculation for linear and logistic regression. Statistics in Medicine, 17(4), 1623-1634.

Examples

```
########################################
# predictor X follows normal distribution #
########################################

## probability specification
power.z.logistic(base.prob = 0.15, prob = 0.20,
                 alpha = 0.05, power = 0.80,
```

```
                dist = "normal")

## odds ratio specification
power.z.logistic(base.prob = 0.15, odds.ratio = 1.416667,
                alpha = 0.05, power = 0.80,
                dist = "normal")

## regression coefficient specification
power.z.logistic(beta0 = -1.734601, beta1 = 0.3483067,
                alpha = 0.05, power = 0.80,
                dist = "normal")

## change parameters associated with predictor X
pred.dist <- list(dist = "normal", mean = 10, sd = 2)
power.z.logistic(base.prob = 0.15, beta1 = 0.3483067,
                alpha = 0.05, power = 0.80,
                dist = pred.dist)


##############################################
# predictor X follows Bernoulli distribution #
# (such as treatment/control groups)         #
##############################################

## odds ratio specification
power.z.logistic(base.prob = 0.15, odds.ratio = 1.416667,
                alpha = 0.05, power = 0.80,
                dist = "bernoulli")

## change parameters associated with predictor X
pred.dist <- list(dist = "bernoulli", prob = 0.30)
power.z.logistic(base.prob = 0.15, odds.ratio = 1.416667,
                alpha = 0.05, power = 0.80,
                dist = pred.dist)

####################################
# predictor X is an ordinal factor #
####################################

## generating an ordinal predictor
x.ord <- sample(
  x = c(1, 2, 3, 4), # levels
  size = 1e5, # sample size large enough to get stable estimates
  prob = c(0.25, 0.25, 0.25, 0.25), # category probabilities
  replace = TRUE
)

## dummy coding the ordinal predictor
x.ord <- factor(x.ord, ordered = TRUE)
contrasts(x.ord) <- contr.treatment(4, base = 4)
x.dummy <- model.matrix( ~ x.ord)[,-1]
x.data <- as.data.frame(x.dummy)
```

```
## fit linear regression to get multiple r-squared
x.fit <- lm(x.ord1 ~ x.ord2 + x.ord3, data = x.data)

## extract parameters
bern.prob <- mean(x.data$x.ord1)
r.squared.pred <- summary(x.fit)$adj.r.squared

## change parameters associated with predictor X
pred.dist <- list(dist = "bernoulli", prob = bern.prob)
power.z.logistic(base.prob = 0.15, odds.ratio = 1.416667,
               alpha = 0.05, power = 0.80,
               r.squared.pred = r.squared.pred,
               dist = pred.dist)
```

`power.z.mediation` *Power Analysis for Indirect Effects in a Mediation Model (Z, Joint, and Monte Carlo Tests)*

Description

Calculates power or sample size (only one can be NULL at a time) to test indirect effects in a mediation model (Z-Test, Joint Test, and Monte Carlo Interval Test). One can consider explanatory power of the covariates in the mediator and outcome model via specifying R-squared values accordingly. `power.z.mediation()` and `power.z.med()` are the same functions.

NOTE: The function `pwrss.z.mediation()` (or its alias `pwrss.z.med()`) are no longer supported. However, they will remain available as wrappers for the `power.z.mediation` function.

Formulas are validated using Monte Carlo simulation.

Usage

```
power.z.mediation(beta.a, beta.b, beta.cp = 0,
                  sd.predictor = 1, sd.mediator = 1, sd.outcome = 1,
                 r.squared.mediator = beta.a^2 * sd.predictor^2 / sd.mediator^2,
                  r.squared.outcome = (beta.b^2 * sd.mediator^2 +
                                       beta.cp^2 * sd.predictor^2) / sd.outcome^2,
                  n = NULL, power = NULL, alpha = 0.05,
                  alternative = c("two.sided", "one.sided"),
                  method = c("sobel", "aroian", "goodman",
                             "joint", "monte.carlo"),
                  n.simulation = 1000,
                  n.draws = 1000,
                  ceiling = TRUE,
                  verbose = TRUE,
                  pretty = FALSE)
```

Arguments

beta.a	regression coefficient for X -> M path. One can use standardized regression coefficient, but should keep sd.predictor = 1 and sd.mediator = 1 or leave them out as they are default specifications.
beta.b	regression coefficient for M -> Y path. One can use standardized regression coefficient, but should keep sd.mediator = 1 and sd.outcome = 1 or leave them out as they are default specifications.
beta.cp	regression coefficient for X -> Y path (the direct path). One can use standardized regression coefficient, but should keep sd.predictor = 1 and sd.outcome = 1 or leave them out as they are default specifications.
sd.predictor	standard deviation of the predictor (X). For a binary predictor, sd.predictor = sqrt(p*(1-p)) wherep is the proportion of subjects in one of the groups.
sd.mediator	standard deviation of the mediator (M).
sd.outcome	standard deviation of the outcome (Y).
r.squared.mediator	R-squared value for the mediator model (M ~ X). The default is r.squared.mediator = beta.a^2 * sd.predictor^2 / sd.mediator^2 assuming that X is the only predictor. Thus, an r.squared.mediator below this value will throw a warning. To consider other covariates in the mediator model provide a value greater than the default.
r.squared.outcome	R-squared value for the outcome model (Y ~ M + X). The default is r.squared.outcome = (beta.b^2 * sd.mediator^2 + beta.cp^2 * sd.predictor^2) / sd.outcome^2 assuming that M and X are the only predictors. Thus, an r.squared.outcome below this value will throw a warning. To consider other covariates in the outcome model provide a value greater than the default.
n	integer; sample size.
power	statistical power, defined as the probability of correctly rejecting a false null hypothesis, denoted as $1 - \beta$.
alpha	type 1 error rate, defined as the probability of incorrectly rejecting a true null hypothesis, denoted as α.
alternative	character; direction or type of the hypothesis test: "two.sided" or "one.sided".
method	character; "sobel", "aroian", "goodman", "joint" or "monte.carlo". "joint" and "monte.carlo" methods cannot be used for sample size calculation.
n.simulation	integer; number of replications (applies when method = "monte.carlo").
n.draws	integer; number of draws from the distribution of the path coefficients for each replication (applies when method = "monte.carlo").
ceiling	logical; whether sample size should be rounded up. TRUE by default.
verbose	logical; whether the output should be printed on the console. TRUE by default.
pretty	logical; whether the output should show Unicode characters (if encoding allows for it). FALSE by default.

Value

`parms`	list of parameters used in calculation.
`test`	type of the statistical test ("Z-Test", "Joint Test", or "Monte Carlo Interval Test").
`mean`	mean of the alternative distribution.
`sd`	standard deviation of the alternative distribution.
`null.mean`	mean of the null distribution.
`null.sd`	standard deviation of the null distribution.
`z.alpha`	critical value(s).
`power`	statistical power $(1 - \beta)$.
`n`	sample size.

References

Aroian, L. A. (1947). The probability function of the product of two normally distributed variables. Annals of Mathematical Statistics, 18(2), 265-271.

Goodman, L. A. (1960). On the exact variance of products. Journal of the American Statistical Association, 55(292), 708-713.

MacKinnon, D. P., & Dwyer, J. H. (1993). Estimating mediated effects in prevention studies. Evaluation Review, 17(2), 144-158.

MacKinnon, D. P., Warsi, G., & Dwyer, J. H. (1995). A simulation study of mediated effect measures. Multivariate Behavioral Research, 30(1), 41-62.

Preacher, K. J., & Hayes, A. F. (2004). SPSS and SAS procedures for estimating indirect effects in simple mediation models. Behavior Research Methods, Instruments, & Computers, 36, 717-731.

Preacher, K. J., & Hayes, A. F. (2008). Asymptotic and resampling strategies for assessing and comparing indirect effects in multiple mediator models. Behavior Research Methods, 40, 879-891.

Sobel, M. E. (1982). Asymptotic intervals for indirect effects in structural equations models. In S. Leinhart (Ed.), Sociological methodology 1982 (pp. 290-312). Jossey-Bass.

Examples

```
# with standardized coefficients

## statistical power
power.z.mediation(beta.a = 0.25,
            beta.b = 0.25,
            beta.cp = 0.10,
            n = 200)

## minimum required sample size
power.z.mediation(beta.a = 0.25,
            beta.b = 0.25,
            beta.cp = 0.10,
            power = 0.80)

## adjust for covariates in the outcome model
```

```
power.z.mediation(beta.a = 0.25,
            beta.b = 0.25,
            beta.cp = 0.10,
            r.squared.outcome = 0.50,
            power = 0.80)

# with binary predictor X such as treatment/control variable
# in this case standardized coefficients for path a and cp would be Cohen's d values

## statistical power
p <- 0.50 # proportion of subjects in one group
power.z.mediation(beta.a = 0.40,
            beta.b = 0.25,
            beta.cp = 0.10,
            sd.predictor = sqrt(p*(1-p)),
            n = 200)

## minimum required sample size
power.z.mediation(beta.a = 0.40,
            beta.b = 0.25,
            beta.cp = 0.10,
            sd.predictor = sqrt(p*(1-p)),
            power = 0.80)

## adjust for covariates in the outcome model
power.z.mediation(beta.a = 0.40,
            beta.b = 0.25, beta.cp = 0.10,
            r.squared.outcome = 0.50,
            sd.predictor = sqrt(p*(1-p)),
            power = 0.80)
```

`power.z.onecor` *Power Analysis for One-Sample Correlation*

Description

Calculates power or sample size (only one can be NULL at a time) to test a (Pearson) correlation against a constant using Fisher's z transformation.

Formulas are validated using PASS and G*Power.

Usage

```
power.z.onecor(rho, null.rho = 0,
               n = NULL, power = NULL, alpha = 0.05,
               alternative = c("two.sided", "one.sided"),
               ceiling = TRUE, verbose = TRUE, pretty = FALSE)
```

Arguments

rho	correlation.
null.rho	correlation when null is true.
n	sample size.
power	statistical power, defined as the probability of correctly rejecting a false null hypothesis, denoted as $1 - \beta$.
alpha	type 1 error rate, defined as the probability of incorrectly rejecting a true null hypothesis, denoted as α.
alternative	character; direction or type of the hypothesis test: "two.sided" or "one.sided".
ceiling	logical; whether sample size should be rounded up. TRUE by default.
verbose	logical; whether the output should be printed on the console. TRUE by default.
pretty	logical; whether the output should show Unicode characters (if encoding allows for it). FALSE by default.

Value

parms	list of parameters used in calculation.
test	type of the statistical test (Z-Test)
mean	mean of the alternative distribution.
sd	standard deviation of the alternative distribution.
null.mean	mean of the null distribution.
null.sd	standard deviation of the null distribution.
z.alpha	critical value(s).
power	statistical power $(1 - \beta)$.
n	sample size.

References

Bulus, M., & Polat, C. (2023). pwrss R paketi ile istatistiksel guc analizi [Statistical power analysis with pwrss R package]. Ahi Evran Universitesi Kirsehir Egitim Fakultesi Dergisi, 24(3), 2207-2328. doi: 10.29299/kefad.1209913

Chow, S. C., Shao, J., Wang, H., & Lokhnygina, Y. (2018). Sample size calculations in clinical research (3rd ed.). Taylor & Francis/CRC.

Cohen, J. (1988). Statistical power analysis for the behavioral sciences (2nd ed.). Lawrence Erlbaum Associates.

Examples

```
# expected correlation is 0.20 and it is different from 0
# it could be 0.20 as well as -0.20
power.z.onecor(rho = 0.20,
               power = 0.80,
               alpha = 0.05,
```

```
               alternative = "two.sided")

# expected correlation is 0.20 and it is greater than 0.10
power.z.onecor(rho = 0.20, null = 0.10,
               power = 0.80,
               alpha = 0.05,
               alternative = "one.sided")
```

`power.z.oneprop` *Power Analysis for the Test of One Proportion (Normal Approximation and Exact Methods)*

Description

Calculates power or sample size (only one can be NULL at a time) for test of a proportion against a constant using normal approximation or exact method.

Formulas are validated using PASS documentation.

NOTE: The `pwrss.z.prop()` function is deprecated, but it will remain available as a wrapper for the `power.z.oneprop()` function during the transition period.

Usage

```
power.z.oneprop(prob, null.prob = 0.50,
                n = NULL, power = NULL, alpha = 0.05,
                alternative = c("two.sided", "one.sided", "two.one.sided"),
                std.error = c("null", "alternative"),
                arcsine = FALSE, correct = FALSE,
                ceiling = TRUE, verbose = TRUE, pretty = FALSE)

power.exact.oneprop(prob, null.prob = 0.50,
                    n = NULL, power = NULL, alpha = 0.05,
                    alternative = c("two.sided", "one.sided", "two.one.sided"),
                    verbose = TRUE, pretty = FALSE)
```

Arguments

`prob`	probability of success under alternative.
`null.prob`	probability of success under null.
`n`	integer; sample size.
`power`	statistical power, defined as the probability of correctly rejecting a false null hypothesis, denoted as $1 - \beta$.
`alpha`	type 1 error rate, defined as the probability of incorrectly rejecting a true null hypothesis, denoted as α.
`std.error`	character; whether to calculate standard error using "null" or "alternative" value. "null" by default.

arcsine	logical; whether arcsine transformation should be applied. FALSE by default. Note that when arcsine = TRUE, any specification to correct and std.error will be ignored.
correct	logical; whether Yate's continuity correction should be applied.
alternative	character; the direction or type of the hypothesis test: "two.sided", "one.sided", or "two.one.sided". For non-inferiority or superiority tests, add margin to the null hypothesis value and use alternative = "one.sided".
ceiling	logical; whether sample size should be rounded up. TRUE by default.
verbose	logical; whether the output should be printed on the console. TRUE by default.
pretty	logical; whether the output should show Unicode characters (if encoding allows for it). FALSE by default.

Value

parms	list of parameters used in calculation.
test	type of the statistical test ("exact").
mean	mean of the alternative distribution.
sd	standard deviation of the alternative distribution.
null.mean	mean of the null distribution.
null.sd	standard deviation of the null distribution.
z.alpha	critical value(s).
power	statistical power $(1 - \beta)$.
n	sample size.

References

Bulus, M., & Polat, C. (2023). pwrss R paketi ile istatistiksel guc analizi [Statistical power analysis with pwrss R package]. Ahi Evran Universitesi Kirsehir Egitim Fakultesi Dergisi, 24(3), 2207-2328. doi: 10.29299/kefad.1209913

Examples

```
# power
power.z.oneprop(prob = 0.45, null.prob = 0.50,
                alpha = 0.05, n = 500,
                alternative = "one.sided")

power.exact.oneprop(prob = 0.45, null.prob = 0.50,
                    alpha = 0.05, n = 500,
                    alternative = "one.sided")



# sample size
power.z.oneprop(prob = 0.45, null.prob = 0.50,
                alpha = 0.05, power = 0.80,
```

```
                  alternative = "one.sided")

power.exact.oneprop(prob = 0.45, null.prob = 0.50,
                    alpha = 0.05, power = 0.80,
                    alternative = "one.sided")
```

---

power.z.poisson *Power Analysis for Poisson Regression Coefficient (Wald's z Test)*

Description

Calculates power or sample size (only one can be NULL at a time) to test a single coefficient in poisson regression. `power.z.poisson()` and `power.z.poisreg()` are the same functions, as well as `pwrss.z.poisson()` and `pwrss.z.poisreg()`. The distribution of the predictor variable can be one of the following: `c("normal", "poisson", "uniform", "exponential", "binomial", "bernouilli", "lognormal")`. The default parameters for these distributions are

```
distribution = list(dist = "normal", mean = 0, sd = 1)
distribution = list(dist = "poisson", lambda = 1)
distribution = list(dist = "uniform", min = 0, max = 1)
distribution = list(dist = "exponential", rate = 1)
distribution = list(dist = "binomial", size = 1, prob = 0.50)
distribution = list(dist = "bernoulli", prob = 0.50)
distribution = list(dist = "lognormal", meanlog = 0, sdlog = 1)
```

Parameters defined in `list()` form can be modified, but the names should be kept the same. It is sufficient to use distribution's name for default parameters (e.g. `dist = "normal"`).

Formulas are validated using Monte Carlo simulation, G*Power, and tables in PASS documentation.

NOTE: The `pwrss.z.poisson()` and its alias `pwrss.z.poisreg()` are deprecated. However, they will remain available as wrappers for the `power.z.logistic()` function.

Usage

```
power.z.poisson(base.rate = NULL, rate.ratio = NULL,
                beta0 = log(base.rate), beta1 = log(rate.ratio),
                n = NULL, power = NULL,
                r.squared.predictor = 0, mean.exposure = 1,
                alpha = 0.05, alternative = c("two.sided", "one.sided"),
                method = c("demidenko(vc)", "demidenko", "signorini"),
                distribution = "normal", ceiling = TRUE,
                verbose = TRUE, pretty = FALSE)
```

Arguments

`base.rate`	the base mean event rate.
`rate.ratio`	event rate ratio. The relative increase in the mean event rate for one unit increase in the predictor (similar to odds ratio in logistic regression).

beta0	log(base.rate) or natural logarithm of the base mean event rate.
beta1	log(rate.ratio) or natural logarithm of the relative increase in the mean event rate for one unit increase in the predictor.
mean.exposure	the mean exposure time (should be > 0). Usually 1
n	integer; sample size.
power	statistical power, defined as the probability of correctly rejecting a false null hypothesis, denoted as $1 - \beta$.
r.squared.predictor	proportion of variance in the predictor accounted for by other covariates. This is not a pseudo R-squared. To compute it, regress the predictor on the covariates and extract the adjusted R-squared from that model.
alpha	type 1 error rate, defined as the probability of incorrectly rejecting a true null hypothesis, denoted as α.
alternative	character; direction or type of the hypothesis test: "not equal", "greater", "less".
method	character; calculation method. "demidenko(vc)" stands for Demidenko (2007) procedure with variance correction; "demidenko" stands for Demidenko (2007) procedure without variance correction; "signorini" stands for Signorini (1991) procedure. "demidenko" and "signorini" methods produce similar results but "demidenko(vc)" is more precise.
distribution	character; distribution family. Can be one of the c("normal", "poisson", "uniform", "exponential", "binomial", "bernouilli", "lognormal").
ceiling	logical; whether sample size should be rounded up. TRUE by default.
verbose	logical; whether the output should be printed on the console. TRUE by default.
pretty	logical; whether the output should show Unicode characters (if encoding allows for it). FALSE by default.

Value

parms	list of parameters used in calculation.
test	type of the statistical test (Z-Test).
mean	mean of the alternative distribution.
sd	standard deviation of the alternative distribution.
null.mean	mean of the null distribution.
null.sd	standard deviation of the null distribution.
z.alpha	critical value(s).
power	statistical power $(1 - \beta)$
n	sample size.

References

Demidenko, E. (2007). Sample size determination for logistic regression revisited. Statistics in Medicine, 26(18), 3385-3397. doi: 10.1002/sim.2771

Signorini, D. F. (1991). Sample size for poisson regression. Biometrika, 78(2), 446-450.

Examples

```
# predictor X follows normal distribution

## regression coefficient specification
power.z.poisson(beta0 = 0.50, beta1 = -0.10,
                alpha = 0.05, power = 0.80,
                dist = "normal")

## rate ratio specification
power.z.poisson(base.rate = exp(0.50),
                rate.ratio = exp(-0.10),
                alpha = 0.05, power = 0.80,
                dist = "normal")

## change parameters associated with predictor X
dist.x <- list(dist = "normal", mean = 10, sd = 2)
power.z.poisson(base.rate = exp(0.50),
                rate.ratio = exp(-0.10),
                alpha = 0.05, power = 0.80,
                dist = dist.x)


# predictor X follows Bernoulli distribution (such as treatment/control groups)

## regression coefficient specification
power.z.poisson(beta0 = 0.50, beta1 = -0.10,
                alpha = 0.05, power = 0.80,
                dist = "bernoulli")

## rate ratio specification
power.z.poisson(base.rate = exp(0.50),
                rate.ratio = exp(-0.10),
                alpha = 0.05, power = 0.80,
                dist = "bernoulli")

## change parameters associatied with predictor X
dist.x <- list(dist = "bernoulli", prob = 0.30)
power.z.poisson(base.rate = exp(0.50),
                rate.ratio = exp(-0.10),
                alpha = 0.05, power = 0.80,
                dist = dist.x)
```

`power.z.twocors` *Power Analysis for Independent Correlations*

Description

Calculates power or sample size (only one can be NULL at a time) to test difference between two independent (Pearson) correlations using Fisher's z transformation.

Formulas are validated using PASS and G*Power.

Usage

```
power.z.twocors(rho1, rho2,
                n2 = NULL, n.ratio = 1,
                power = NULL, alpha = 0.05,
                alternative = c("two.sided", "one.sided"),
                ceiling = TRUE, verbose = TRUE, pretty = FALSE)
```

Arguments

rho1	correlation in the first group.
rho2	correlation in the second group.
n2	sample size in the second group. Sample size in the first group can be calculated as n2*kappa. By default, n1 = n2 because kappa = 1.
n.ratio	n1/n2 ratio.
power	statistical power, defined as the probability of correctly rejecting a false null hypothesis, denoted as $1 - \beta$.
alpha	type 1 error rate, defined as the probability of incorrectly rejecting a true null hypothesis, denoted as α.
alternative	character; direction or type of the hypothesis test: "two.sided" or "one.sided".
ceiling	logical; whether sample size should be rounded up. TRUE by default.
verbose	logical; whether the output should be printed on the console. TRUE by default.
pretty	logical; whether the output should show Unicode characters (if encoding allows for it). FALSE by default.

Value

parms	list of parameters used in calculation.
test	type of the statistical test (Z-Test)
mean	mean of the alternative distribution.
sd	standard deviation of the alternative distribution.
null.mean	mean of the null distribution.
null.sd	standard deviation of the null distribution.
z.alpha	critical value(s).
power	statistical power $(1 - \beta)$
n	sample size for the first and second groups, in the form of c(n1, n2).

References

Bulus, M., & Polat, C. (2023). pwrss R paketi ile istatistiksel guc analizi [Statistical power analysis with pwrss R package]. Ahi Evran Universitesi Kirsehir Egitim Fakultesi Dergisi, 24(3), 2207-2328. doi: 10.29299/kefad.1209913

Chow, S. C., Shao, J., Wang, H., & Lokhnygina, Y. (2018). Sample size calculations in clinical research (3rd ed.). Taylor & Francis/CRC.

Cohen, J. (1988). Statistical power analysis for the behavioral sciences (2nd ed.). Lawrence Erlbaum Associates.

Examples

```
# difference between r1 and r2 is different from zero
# it could be -0.10 as well as 0.10
power.z.twocors(rho1 = .20, rho2 = 0.30,
               alpha = 0.05, power = .80,
               alternative = "two.sided")

# difference between r1 and r2 is greater than zero
power.z.twocors(rho1 = .30, rho2 = 0.20,
               alpha = 0.05, power = .80,
               alternative = "one.sided")
```

`power.z.twocors.steiger` *Power Analysis for Dependent Correlations (Steiger's Z-Test)*

Description

Calculates power or sample size (only one can be NULL at a time) to test difference between paired correlations (Pearson) using Fisher's Z-transformation.

Validated via PASS and G*Power.

Usage

```
power.z.twocors.steiger(rho12, rho13, rho23,
                        rho14 = NULL, rho24 = NULL, rho34 = NULL,
                        n = NULL, power = NULL, alpha = 0.05,
                        alternative = c("two.sided", "one.sided"),
                        pooled = TRUE, common.index = FALSE,
                        ceiling = TRUE, verbose = TRUE, pretty = FALSE)
```

Arguments

`rho12`	correlation between variable V1 and V2 (one common index and no common index). Check examples below.
`rho13`	correlation between variable V1 and V3 (one common index and no common index). Check examples below.
`rho23`	correlation between variable V2 and V3 (one common index and no common index). Check examples below.
`rho14`	correlation between variable V1 and V4 (no common index only). Check examples below.
`rho24`	correlation between variable V2 and V4 (no common index only). Check examples below.
`rho34`	correlation between variable V3 and V4 (no common index only). Check examples below.

n	integer; sample size.
power	statistical power, defined as the probability of correctly rejecting a false null hypothesis, denoted as $1 - \beta$.
alpha	type 1 error rate, defined as the probability of incorrectly rejecting a true null hypothesis, denoted as α.
alternative	character; the direction or type of the hypothesis test: "two.sided" or "one.sided".
pooled	logical; whether standard error should be pooled. TRUE by default.
common.index	logical; whether calculations pertain to one common index. TRUE means calculations involve correlations with a common index (where both correlations share one variable). FALSE (default) means calculations pertain to correlations with no common index (where all relevant correlations must be explicitly specified). Check examples below.
ceiling	logical; if TRUE rounds up sample size.
verbose	logical; if FALSE no output is printed on the console.
pretty	logical; whether the output should show Unicode characters (if encoding allows for it). FALSE by default.

Value

parms	list of parameters used in calculation.
test	type of the statistical test (Z-Test)
mean	mean of the alternative distribution.
sd	standard deviation of the alternative distribution.
null.mean	mean of the null distribution.
null.sd	standard deviation of the null distribution.
z.alpha	critical value(s).
power	statistical power $(1 - \beta)$.
n	sample size for the first and second groups, in the form of c(n1, n2).

References

Steiger, J. H. (1980). Tests for comparing elements of a correlation matrix. Psychological Bulletin, 87(2), 245-251. doi: 10.1037/00332909.87.2.245

Examples

```
# example data for one common index
# compare cor(V1, V2) to cor(V1, V3)

# subject    V1       V2      V3
# <int>    <dbl>    <dbl>    <dbl>
#   1       1.2      2.3      0.8
#   2      -0.0      1.1      0.7
#   3       1.9     -0.4     -2.3
#   4       0.7      1.3      0.4
```

```
#   5       2.1     -0.1      0.8
#   ...     ...      ...      ...
#   1000   -0.5      2.7     -1.7

# V1: socio-economic status (common)
# V2: pretest
# V3: post-test

power.z.twocors.steiger(rho12 = 0.35, rho13 = 0.45, rho23 = 0.05,
                        n = 1000, power = NULL, alpha = 0.05,
                        alternative = "two.sided",
                        common.index = TRUE)


# example data for no common index
# compare cor(V1, V2) to cor(V3, V4)

# subject    V1       V2       V3       V4
# <int>    <dbl>    <dbl>    <dbl>    <dbl>
#   1       1.2      2.3      0.8      1.2
#   2      -0.0      1.1      0.7      0.9
#   3       1.9     -0.4     -2.3     -0.1
#   4       0.7      1.3      0.4     -0.3
#   5       2.1     -0.1      0.8      2.7
#   ...     ...      ...      ...      ...
#   1000   -0.5      2.7     -1.7      0.8

# V1: pretest reading
# V2: pretest math
# V3: post-test reading
# V4: post-test math

power.z.twocors.steiger(rho12 = 0.45, rho13 = 0.45, rho23 = 0.50,
                        rho14 = 0.50, rho24 = 0.80, rho34 = 0.55,
                        n = 1000, power = NULL, alpha = 0.05,
                        alternative = "two.sided",
                        common.index = FALSE)
```

`power.z.twoprops` *Power Analysis for Testing Difference Between Two Proportions (Normal Approximation and Exact Methods)*

Description

Calculates power or sample size (only one can be NULL at a time) for two proportions using normal approximation method.

Validated via G*Power and PASS documentation.

NOTE: The `pwrss.z.2props()` function is deprecated, but it will remain available as a wrapper for the `power.z.twoprops()` function during the transition period.

Usage

```
power.z.twoprops(prob1, prob2, margin = 0,
                 n2 = NULL, n.ratio = 1,
                 power = NULL, alpha = 0.05,
                 alternative = c("two.sided", "one.sided", "two.one.sided"),
                 arcsine = FALSE, correct = FALSE,
                 paired = FALSE, rho.paired = 0.50,
                 std.error = c("pooled", "unpooled"),
                 ceiling = TRUE, verbose = TRUE, pretty = FALSE)

power.exact.twoprops(prob1, prob2, n2 = NULL, n.ratio = 1,
                     power = NULL, alpha = 0.05,
                     alternative = c("two.sided", "one.sided"),
                     paired = FALSE, rho.paired = 0.50,
                     method = c("exact", "approximate"),
                     ceiling = TRUE, verbose = TRUE, pretty = FALSE)
```

Arguments

prob1	probability of success in the first group.
prob2	probability of success in the second group.
margin	ignorable prob1 - prob2 difference. For two one-sided tests provide lower and upper margins in the form of c(lower, upper).
n2	integer; sample size for the second group.
n.ratio	sample size ratio (n1 / n2).
power	statistical power, defined as the probability of correctly rejecting a false null hypothesis, denoted as $1 - \beta$.
alpha	type 1 error rate, defined as the probability of incorrectly rejecting a true null hypothesis, denoted as α.
paired	logical; if TRUE samples are paired. FALSE by default.
rho.paired	correlation between paired observations.
method	character; whether to use "approximate" or "exact" method. Default is "exact" (only in the power.exact.twoprops() function).
arcsine	logical; whether arcsine transformation should be applied. Note that this only applies to independent proportions without continuity correction.
correct	logical; whether Yates' continuity correction should be applied to the test statistic. Ignored for the paired test.
std.error	character; whether to calculate standard error using "pooled" or "unpooled" standard deviation. Ignored for the paired test.
alternative	character; direction or type of the hypothesis test: "two.sided", "one.sided", or "two.one.sided".
ceiling	logical; TRUE rounds up sample size in each group.
verbose	logical; TRUE prints the output on the console.
pretty	logical; whether the output should show Unicode characters (if encoding allows for it). FALSE by default.

Value

parms	list of parameters used in calculation.
test	type of the test, which is "z" or "exact".
power	statistical power $(1 - \beta)$.
mean	mean of the alternative distribution.
sd	standard deviation of the alternative distribution.
null.mean	mean of the null distribution.
null.sd	standard deviation of the null distribution.
z.alpha	critical value(s).
n	sample size in the form of c(n1, n2) (applies to independent proportions).
n.total	total sample size (applies to independent proportions).
n.paired	paired sample size (applies to paired proportions).

References

Bulus, M., & Polat, C. (2023). pwrss R paketi ile istatistiksel guc analizi [Statistical power analysis with pwrss R package]. Ahi Evran Universitesi Kirsehir Egitim Fakultesi Dergisi, 24(3), 2207-2328. doi: 10.29299/kefad.1209913

Examples

```
# power
power.z.twoprops(prob1 = 0.65, prob2 = 0.60,
                 alpha = 0.05, n2 = 500,
                 alternative = "one.sided")

# sample size
power.z.twoprops(prob1 = 0.65, prob2 = 0.60,
                 alpha = 0.05, power = 0.80,
                 alternative = "one.sided")
```

---

probs.to.h *Conversion from Probability Difference to Cohen's h*

Description

Helper function to convert probability difference to Cohen's h (and vice versa).

Usage

```
probs.to.h(prob1, prob2 = 0.50, verbose = TRUE)
```

Arguments

prob1	probability of success in the first group, or under the alternative hypothesis in the one-sample case).
prob2	probability of success in the second group, or under the null hypothesis in the one-sample case).
h	Cohen's h effect size.
verbose	logical; whether the output should be printed on the console. TRUE by default.

Value

p1	probability of success in the first group, or under the alternative hypothesis in the one-sample case).
p2	probability of success in the second group, or under the null hypothesis in the one-sample case).
h	Cohen's h effect size.

Examples

```
probs.to.h(prob1 = 0.56, prob2 = 0.50)
```

---

probs.to.w *Conversion from Probabilities to Cohen's w*

Description

Helper function to convert (multinomial or product-multinomial) probabilities to Cohen's w.

Usage

```
probs.to.w(prob.matrix,
           null.prob.matrix = NULL,
           verbose = TRUE)
```

Arguments

prob.matrix	a vector or matrix of cell probabilities under alternative hypothesis
null.prob.matrix	a vector or matrix of cell probabilities under null hypothesis. Calculated automatically when prob.matrix is specified. The default can be overwritten by the user via providing a vector of the same size or matrix of the same dimensions as prob.matrix
verbose	logical; whether the output should be printed on the console. TRUE by default.

Value

w	Cohen's w effect size. It can be any of Cohen's W, Phi coefficient, Cramer's V. Phi coefficient is defined as `sqrt(X2/n)` and Cramer's V is defined as `sqrt(X2/(n*v))` where v is `min(nrow - 1, ncol - 1)` and X2 is the chi-square statistic.
df	degrees of freedom.

References

Cohen, J. (1988). Statistical power analysis for the behavioral sciences (2nd ed.). Lawrence Erlbaum Associates.

Examples

```
# ------------------------------------------------------------#
# Example 1: Cohen's W                                        #
# goodness-of-fit test for 1 x k or k x 1 table               #
# How many subjects are needed to claim that                  #
# girls choose STEM related majors less than males?          #
# ------------------------------------------------------------#

## from https://www.aauw.org/resources/research/the-stem-gap/
## 28 percent of the  workforce in STEM field is women
prob.vector <- c(0.28, 0.72)
null.prob.vector <- c(0.50, 0.50)
probs.to.w(prob.vector, null.prob.vector)

power.chisq.gof(w = 0.44, df = 1,
                alpha = 0.05, power = 0.80)


# ------------------------------------------------------------#
# Example 2: Phi Coefficient (or Cramer's V or Cohen's W)     #
# test of independence for 2 x 2 contingency tables           #
# How many subjects are needed to claim that                  #
# girls are underdiagnosed with ADHD?                         #
# ------------------------------------------------------------#

## from https://time.com/growing-up-with-adhd/
## 5.6 percent of girls and 13.2 percent of boys are diagnosed with ADHD
prob.matrix <- rbind(c(0.056, 0.132),
                     c(0.944, 0.868))
colnames(prob.matrix) <- c("Girl", "Boy")
rownames(prob.matrix) <- c("ADHD", "No ADHD")
prob.matrix

probs.to.w(prob.matrix)

power.chisq.gof(w = 0.1302134, df = 1,
                alpha = 0.05, power = 0.80)
```

```
# ------------------------------------------------------#
# Example 3: Cramer's V (or Cohen's W)                  #
# test of independence for j x k contingency tables     #
# How many subjects are needed to detect the relationship #
# between depression severity and gender?               #
# ------------------------------------------------------#

## from https://doi.org/10.1016/j.jad.2019.11.121
prob.matrix <- cbind(c(0.6759, 0.1559, 0.1281, 0.0323, 0.0078),
                     c(0.6771, 0.1519, 0.1368, 0.0241, 0.0101))
rownames(prob.matrix) <- c("Normal", "Mild", "Moderate",
                           "Severe", "Extremely Severe")
colnames(prob.matrix) <- c("Female", "Male")
prob.matrix

probs.to.w(prob.matrix)

power.chisq.gof(w = 0.03022008, df = 4,
                alpha = 0.05, power = 0.80)
```

`pwrss.t.contrasts` *Power Analysis for One-, Two-, Three-Way ANCOVA Contrasts and Multiple Comparisons (T-Tests)*

Description

Calculates power or sample size for one-, two-, three-Way ANCOVA contrasts and multiple comparisons. The `pwrss.t.contrast()` function allows a single contrast. For multiple contrast testing (multiple comparisons) use the `pwrss.t.contrasts()` function. The `pwrss.t.contrasts()` function also allows adjustment to alpha due to multiple testing. All arguments are the same as with the earlier function except that it accepts an object returned from the `pwrss.f.ancova.shieh()` function for convenience. Beware that, in this case, all other arguments are ignored except `alpha` and `adjust.alpha`.

The `pwrss.t.contrasts()` function returns a data frame with as many rows as number of contrasts and eight columns with names 'contrast', 'comparison', 'psi', 'd', 'ncp', 'df', 'n.total', and 'power'. 'psi' is the contrast estimate. 'd' is the standardized contrast estimate. Remaining parameters are explained elsewhere.

Note that R has a partial matching feature which allows you to specify shortened versions of arguments, such as `mu` or `mu.vec` instead of `mu.vector`, or such as `k` or `k.cov` instead of `k.covariates`.

Formulas are validated using examples and tables in Shieh (2017).

Usage

```
power.t.contrast(mu.vector,
                 sd.vector,
                 contrast.vector,
                 n.vector = NULL, p.vector = NULL,
```

```
                  r.squared = 0, k.covariates = 1,
                  power = NULL, alpha = 0.05,
                  tukey.kramer = FALSE,
                  ceiling = TRUE, verbose = TRUE, pretty = FALSE)

power.t.contrasts(x = NULL,
                  mu.vector = NULL,
                  sd.vector = NULL,
                  n.vector = NULL, p.vector = NULL,
                  r.squared = 0, k.covariates = 1,
                  contrast.matrix = NULL,
                  power = NULL, alpha = 0.05,
                  adjust.alpha = c("none", "tukey", "bonferroni",
                                   "holm", "hochberg", "hommel",
                                   "BH", "BY", "fdr"),
                  ceiling = TRUE, verbose = TRUE, pretty = FALSE)
```

Arguments

x	object; an object returned from pwrss.f.ancova.shieh() function.
mu.vector	vector; adjusted means (or estimated marginal means) for each level of a factor. Ignored when 'x' is specified.
sd.vector	vector; unadjusted standard deviations for each level of a factor. Ignored when 'x' is specified.
n.vector	vector; sample sizes for each level of a factor. Ignored when 'x' is specified.
p.vector	vector; proportion of total sample size in each level of a factor. These proportions should sum to one. Ignored when 'x' is specified.
r.squared	explanatory power of covariates (R-squared) in the ANCOVA model. Ignored when 'x' is specified.
k.covariates	Number of covariates in the ANCOVA model. Ignored when 'x' is specified.
contrast.vector	vector; a single contrast in the form of a vector with as many elements as number of levels or groups (or cells in factorial designs). Ignored when 'x' is specified.
contrast.matrix	vector or matrix; contrasts should not be confused with the model (design) matrix. Rows of contrast matrix indicate independent vector of contrasts summing to zero. The default contrast matrix is constructed using deviation coding scheme (a.k.a. effect coding). Columns in the contrast matrix indicate number of levels or groups (or cells in factorial designs). Ignored when 'x' is specified.
power	statistical power, defined as the probability of correctly rejecting a false null hypothesis, denoted as $1 - \beta$. Ignored when 'x' is specified.
alpha	type 1 error rate, defined as the probability of incorrectly rejecting a true null hypothesis, denoted as α. Note that this should be specified even if 'x' is specified. The 'alpha' value within the 'x' object pertains to the omnibus test, NOT the test of contrasts.

tukey.kramer	logical; TRUE by default. If FALSE no adjustment will be made to control Type 1 error.
adjust.alpha	character; one of the methods in c("none", "tukey", "bonferroni", "holm", "hochberg", "hommel", "BH", "BY", "fdr") to control Type 1 error. See ?stats::p.adjust.
ceiling	logical; TRUE by default. If FALSE sample size in each cell is NOT rounded up.
verbose	logical; TRUE by default. If FALSE no output is printed on the console.
pretty	logical; whether the output should show Unicode characters (if encoding allows for it). FALSE by default.

Value

parms	list of parameters used in calculation.
test	type of the statistical test (T-Test).
psi	contrast-weighted mean difference.
d	contrast-weighted standardized mean difference.
df	degrees of freedom.
t.alpha	critical values.
ncp	non-centrality parameter for the alternative.
null.ncp	non-centrality parameter for the null.
power	statistical power $(1 - \beta)$.
n.total	total sample size.

References

Shieh, G. (2017). Power and sample size calculations for contrast analysis in ANCOVA. Multivariate Behavioral Research, 52(1), 1-11. doi: 10.1080/00273171.2016.1219841

Examples

```
######################################################################
######################  planned contrasts  ###########################
######################################################################

#########################
## dummy coding scheme ##
#########################

contrast.object <- factorial.contrasts(factor.levels = 3, # one factor w/ 3 levels
                                       coding = "treatment") # use dummy coding scheme

# get contrast matrix from the contrast object
contrast.matrix <- contrast.object$contrast.matrix

# calculate sample size given design characteristics
ancova.design <- power.f.ancova.shieh(mu.vector = c(0.15, 0.30,  0.20), # marginal means
                                 sd.vector = c(1, 1, 1), # unadjusted standard deviations
                               p.vector = c(1/3, 1/3, 1/3), # allocation, should sum to 1
```

```
                                      contrast.matrix = contrast.matrix,
                                      r.squared = 0.50,
                                      k.covariates = 1,
                                      alpha = 0.05, power = 0.80)

# power of planned contrasts, adjusted for alpha level
power.t.contrasts(ancova.design, adjust.alpha = "fdr")

############################
## Helmert coding scheme ##
############################

contrast.object <- factorial.contrasts(factor.levels = 3, # one factor w/ 4 levels
                                       coding = "helmert") # use helmert coding scheme

# get contrast matrix from the contrast object
contrast.matrix <- contrast.object$contrast.matrix

# calculate sample size given design characteristics
ancova.design <- power.f.ancova.shieh(mu.vector = c(0.15, 0.30,  0.20), # marginal means
                                sd.vector = c(1, 1, 1), # unadjusted standard deviations
                               p.vector = c(1/3, 1/3, 1/3), # allocation, should sum to 1
                                      contrast.matrix = contrast.matrix,
                                      r.squared = 0.50,
                                      k.covariates = 1,
                                      alpha = 0.05, power = 0.80)

# power of planned contrasts
power.t.contrasts(ancova.design)

##############################
## polynomial coding scheme ##
##############################

contrast.object <- factorial.contrasts(factor.levels = 3, # one factor w/ 4 levels
                                       coding = "poly") # use polynomial coding scheme

# get contrast matrix from the contrast object
contrast.matrix <- contrast.object$contrast.matrix

# calculate sample size given design characteristics
ancova.design <- power.f.ancova.shieh(mu.vector = c(0.15, 0.30, 0.20), # marginal means
                                sd.vector = c(1, 1, 1), # unadjusted standard deviations
                               p.vector = c(1/3, 1/3, 1/3), # allocation, should sum to 1
                                      contrast.matrix = contrast.matrix,
                                      r.squared = 0.50,
                                      k.covariates = 1,
                                      alpha = 0.05, power = 0.80)

# power of the planned contrasts
power.t.contrasts(ancova.design)

######################
```

```
## custom contrasts ##
######################

# custom contrasts
contrast.matrix <- rbind(
  cbind(A1 = 1, A2 = -0.50, A3 = -0.50),
  cbind(A1 = 0.50, A2 = 0.50, A3 = -1)
)
# labels are not required for custom contrasts,
# but they make it easier to understand power.t.contrasts() output

# calculate sample size given design characteristics
ancova.design <- power.f.ancova.shieh(mu.vector = c(0.15, 0.30, 0.20), # marginal means
                                sd.vector = c(1, 1, 1), # unadjusted standard deviations
                               p.vector = c(1/3, 1/3, 1/3), # allocation, should sum to 1
                                       contrast.matrix = contrast.matrix,
                                       r.squared = 0.50,
                                       k.covariates = 1,
                                       alpha = 0.05, power = 0.80)

# power of the planned contrasts
power.t.contrasts(ancova.design)
```

Index