Package: plansize (via r-universe)

November 16, 2026

Title Plan and Justify Sample Size Before Data Collection

Version 0.0.0.9000

Description Plans and documents sample size for a study before data collection, using validated power engines.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

Config/testthat/edition 3

Imports pwrss (>= 1.3.3)

Suggests DMAR, pwr, testthat (>= 3.2.0), withr

Config/roxygen2/version 8.1.1

Repository https://emresaglamci0.r-universe.dev

Date/Publication 2026-11-15 17:26:18 UTC

RemoteUrl https://github.com/emresaglamci0/r-paketi

RemoteRef HEAD

RemoteSha 32e091f64fb5d307a45a72625e04f72b78116dde

Contents

plan_precision 2
plan_study 3
print.plansize_methods 6
print.plansize_plan 6
print.plansize_precision 7
print.plansize_sensitivity 7
print.plansize_simulation 8
sensitivity 8
simulate_power 9
write_methods 10

Index **12**

plan_precision *Plan a sample size for estimating one proportion*

Description

Sample size so that the confidence interval of one proportion has half-width `e` (Cochran's normal-approximation formula, `n0 = z^2 p (1 - p) / e^2`). Corrections are applied in this order, with one final rounding up: finite population correction `n0 / (1 + (n0 - 1) / N)` (only if `N` is given), then `* deff`, then `/ (1 - dropout)`. The order is a method decision from the package design; no published source for putting the finite population correction before the design effect was found. Applying the design effect first would give a smaller n (407 instead of 463 for `e = .05`, `N = 1000`, `deff = 1.5`, `dropout = .1`), so this order errs on the safe side. Some sources use `n0 / (1 + n0 / N)`; before `deff` and `dropout` the difference is at most 1 participant.

Usage

```
plan_precision(p = 0.5, e, conf_level = 0.95, N = NULL, dropout = 0, deff = 1)
```

Arguments

`p`	Expected proportion, strictly between 0 and 1. Default 0.5.
`e`	Half-width of the confidence interval, above 0 and below `min(p, 1 - p)`. Required.
`conf_level`	Confidence level, strictly between 0 and 1.
`N`	Population size (a whole number, at least 2) or `NULL` for an infinite population. A warning is given if the plan exceeds `N`.
`dropout`	Expected nonresponse or dropout proportion, in [0, 1).
`deff`	Design effect, at least 1.

Details

`p = 0.5` is the most conservative value (largest n). The Wald (normal) approximation is weak when p is near 0 or 1 and when n is small: for p = .1 and n = 139 the true coverage of a nominal 95% interval is about .93, and for p = .02 and n = 31 about .47. `e` is the half-width of the interval, not the full width. Precision is not power: this plan does not test a hypothesis, so `sensitivity()` does not accept it.

Value

A list of class `plansize_precision`; `n_required` is the integer recruitment target.

plan_study *Plan a study's sample size*

Description

Computes the analytic n for an independent two-sample t test, a paired (repeated-measures) t test, the overall F test of a one-way ANOVA with k equal groups, a comparison of two independent proportions (pooled-SE z test) or a Pearson correlation (Fisher z test of H0: rho = 0) or a chi-square test of independence (Cohen's w), then applies the design effect and dropout correction with one final ceiling.

Usage

```
plan_study(
  test = "t_independent",
  effect,
  r = NULL,
  k = NULL,
  p1 = NULL,
  p2 = NULL,
  rows = NULL,
  cols = NULL,
  predictors = NULL,
  tested = NULL,
  r2_covariates = NULL,
  effect_type = c("f", "eta2"),
  effect_basis = c("unknown", "prior_study", "pilot", "smallest_of_interest",
    "convention", "observed"),
  effect_source = NULL,
  alpha = 0.05,
  power = 0.8,
  alternative = c("two.sided", "one.sided"),
  dropout = 0,
  deff = 1
)
```

Arguments

test	`"t_independent"` (default), `"t_paired"`, `"anova_oneway"`, `"prop_two"`, `"cor_pearson"`, `"chisq_indep"`, `"reg_overall"` (overall F test of a multiple regression) or `"reg_partial"` (F test of the R squared change from a subset of the predictors).
effect	Effect size, a single number: positive for the t tests and ANOVA; not used with `"prop_two"`; for `"cor_pearson"` the expected Pearson r, between -1 and 1 and not 0 (its sign sets the direction of a one-sided test); for `"chisq_indep"` Cohen's w, above 0 and at most 1 (the test has no direction, so `alternative` must not be set). Cohen's d for the t tests; for "anova_oneway", Cohen's f or eta squared

as set by effect_type. For "reg_overall", R squared or Cohen f squared as set by effect_type; for "reg_partial", the R squared change or the partial Cohen f squared of the tested predictors, as set by effect_type. For `"t_paired"`, d is the mean change divided by the SD of one measurement (pre and post SDs assumed equal), not by the SD of the difference scores; do not enter a dz here. For "chisq_indep", w is not Cramer's V: for tables larger than 2 x 2, w = V * sqrt(min(rows, cols) - 1). w above 1 is possible when rows and cols are both at least 3, but pwrss does not support it.

`r` Only for `"t_paired"`: the correlation between the two repeated measurements, strictly between -1 and 1 (d is converted to dz = d / sqrt(2 * (1 - r))). An error for the other tests. For `"cor_pearson"`, give the expected correlation as `effect`, not as `r`.

`k` Number of equal-sized groups, a single whole number of at least 2. Required for `"anova_oneway"`; an error for the t tests.

`p1`, `p2` Expected proportions in group 1 and group 2, each strictly between 0 and 1 and different. Required for `"prop_two"` (pooled-SE z test; the effect is the difference p1 - p2, so `effect` must not be given); an error for the other tests.

`rows`, `cols` Number of rows and columns of the contingency table, each a single whole number from 2 to 1000. Required for `"chisq_indep"` (df = (rows - 1) * (cols - 1)); an error for the other tests.

`predictors` Number of predictors in the regression model, a single whole number from 1 to 1000. Required for `"reg_overall"` and `"reg_partial"` (for `"reg_partial"` the total, tested predictors included); an error for the other tests. Count a categorical predictor by its dummy variables (a 4-level factor is 3 predictors).

`tested` Only for `"reg_partial"`: number of predictors tested together, a whole number from 1 to `predictors` - 1 (to test all predictors use `"reg_overall"`). With `tested = 1` the test equals the two-sided t test of one predictor. An error for the other tests.

`r2_covariates` Only for `"reg_partial"` with `effect_type = "r2_change"`: R squared of the model without the tested predictors, from 0 up to but not including 1, with `effect` + `r2_covariates` at most 0.99. An error for the other tests and for `"f2"`. It is not the R squared of the full model: with full-model R squared .30 and an R squared change of .10, enter `effect = 0.10, r2_covariates = 0.20`. Do not enter the partial R squared (.10 / (1 - .20) = .125) as `effect`; with `"r2_change"` that gives a smaller n.

`effect_type` For `"anova_oneway"`: `"f"` (default) or `"eta2"`, whether `effect` is Cohen's f or eta squared (converted by f = sqrt(eta2 / (1 - eta2))). The F test has no direction, so `alternative` must not be set for `"anova_oneway"`. Use eta squared from a one-way design; partial eta squared from a design with other factors or covariates is not the same quantity. For `"reg_overall"`: required, `"r2"` or `"f2"` (exact match, no default). `"r2"` is the R squared of the whole model (share of variance explained by all predictors together, intercept excluded), above 0 and at most 0.99; `"f2"` is Cohen's f squared = R2 / (1 - R2), above 0 and at most 99 (not the partial f squared of one predictor). Cohen's f squared benchmarks .02, .15 and .35 (small, medium, large) are not R squared values: as R squared they are .02, .13 and .26. Entering .15 as `"r2"` instead of `"f2"` gives a different plan without

an error, so check which measure your source reports. Do not enter an adjusted R squared (papers label it "adjusted" or "R2adj"); use the plain R squared. An error for the other tests. For `"reg_partial"`: required, `"r2_change"` or `"f2"` (exact match, no default). `"r2_change"` is the increase in R squared (Delta R squared) when the tested predictors are added to the model with the other predictors (the covariates); `r2_covariates`, the R squared of the model without the tested predictors, is then required. `"f2"` is Cohen's f squared of the tested set = Delta R squared / (1 - R squared of the full model); `r2_covariates` must not be given because f squared already accounts for the covariates. pwrss is given the partial R squared = Delta R squared / (1 - `r2_covariates`) (equal to f2 / (1 + f2)), because the argument pwrss calls `r.squared.change` is used as f2 = r.squared.change / (1 - r.squared.change); entering Delta R squared directly would overstate n whenever the covariates explain variance.

`effect_basis` Where the effect came from: "unknown" (default), "prior_study", "pilot", "smallest_of_interest", "convention" or "observed". "unknown" and "convention" give a warning of class `plansize_basis_warning`; "observed" is an error, use `sensitivity()` instead. The chosen basis is stated in the methods text.

`effect_source` Optional free-text source of the effect. In the methods text, parentheses inside it are shown as square brackets.

`alpha` Significance level, in (0, 1).

`power` Target power, from 0.01 to 0.99 and greater than alpha.

`alternative` `"two.sided"` or `"one.sided"`.

`dropout` Expected dropout proportion, in [0, 1).

`deff` Design effect, at least 1.

Details

For `reg_overall`, print also shows the participants needed if predictors are random (Sampson's random model via DMAR). Measured predictors, as in most observational studies, are random, and recruiting that number keeps the target power. The line is absent when the suggested package DMAR is not installed, when the plan n is above 100,000, or when DMAR fails. The plan n and the methods text keep the fixed-predictor assumption.

For "prop_two", pwrss returns an already rounded n, so the dropout and design effect correction is applied to that rounded n; the recruitment target can be a few participants higher than a single final rounding would give.

For "cor_pearson", n uses the Fisher z approximation without the bias correction of pwr::pwr.r.test (about 1 participant higher), so results can differ slightly from pwr or G*Power; the approximation is rough below about 10 participants but errs on the safe side.

For "chisq_indep", n is for Pearson's chi-square test without continuity correction; in R use chisq.test(x, correct = FALSE) for 2 x 2 tables, as the default Yates correction lowers power.

For "reg_overall", n is the total for the overall F test of a multiple regression (H0: R squared = 0), with `groups` 1. The F model treats the predictors as fixed; with random predictors the power is slightly lower (about 0.78 instead of 0.80 in a simulation with R2 = 0.13 and 3 predictors). Tests of single predictors are covered only through `"reg_partial"` with `tested` = 1.

For "reg_partial", n is the total for the F test of the R squared change of `tested` of `predictors` predictors (H0: Delta R squared = 0), with `groups` 1. pwrss uses the noncentrality f squared * n

(fixed predictors, as in G*Power's "R2 increase, fixed model"); this was checked against a fixed-design simulation. pwr::pwr.f2.test uses Cohen's f squared * (u + v + 1) instead, a different method that gives 0 to `predictors - tested` more participants; do not treat it as a reference for this test. The F model treats the predictors as fixed; with random predictors the power is lower (about 0.77 instead of 0.80 in a simulation, and lower still with a small n and a large R squared).

Value

A list of class `plansize_plan`.

`print.plansize_methods` *Print a methods paragraph*

Description

Print a methods paragraph

Usage

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

Arguments

`x`	A `plansize_methods` from `write_methods()`.
`...`	Unused.

Value

`x`, invisibly.

`print.plansize_plan` *Print a sample size plan*

Description

Print a sample size plan

Usage

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

Arguments

x	A plansize_plan from plan_study().
...	Unused.

Value

x, invisibly.

---

print.plansize_precision *Print a precision plan*

Description

Print a precision plan

Usage

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

Arguments

x	A plansize_precision from plan_precision().
...	Unused.

Value

x, invisibly.

---

print.plansize_sensitivity *Print a sensitivity table*

Description

Print a sensitivity table

Usage

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

Arguments

x	A plansize_sensitivity from sensitivity().
...	Unused.

Value

x, invisibly.

---

print.plansize_simulation *Print a simulation check*

Description

Print a simulation check

Usage

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

Arguments

x	A plansize_simulation from simulate_power().
...	Unused.

Value

x, invisibly.

---

sensitivity *Smallest detectable effect for given sample sizes*

Description

For each combination of recruited per-group n and power, returns the smallest standardized effect detectable after dropout and design effect.

Usage

```
sensitivity(plan, n = NULL, power = c(0.8, 0.9, 0.95))
```

Arguments

plan	A plansize_plan from plan_study(). A precision plan from plan_precision() is an error (it has no hypothesis test).
n	Recruited per-group sample sizes (participants for "t_paired", "cor_pearson" and "chisq_indep"). Defaults to plan$n_per_group.
power	Target powers, each from 0.01 to 0.99 and greater than the plan's alpha.

Details

n_effective is floor(n * (1 - dropout) / deff), rounded down, so the minimum detectable effect for a plan with dropout can be slightly above the planned effect (conservative). With a very large planned effect, the higher powers in the default grid may be infeasible; pass a smaller power grid.

Value

A data frame of class plansize_sensitivity with columns n_recruited, n_effective, power, mde. For "anova_oneway" mde is in the plan's effect_type (f or eta squared); the unit is kept in attr(, "effect_type"). For "prop_two" mde is the detectable p1 with p2 fixed, in the direction of the planned difference. For "cor_pearson" mde is the detectable r with the sign of the planned correlation. For "chisq_indep" mde is the detectable Cohen's w. For "reg_partial" mde is in the plan's effect_type (R squared change or f squared of the tested predictors); the unit is kept in attr(, "effect_type").

---

simulate_power *Check a plan's power by simulation*

Description

Draws reps studies with the planned effect, runs the planned test on each and compares the share of significant results (empirical power) with the analytic power pwrss gives for the same n. This checks that the formula holds at this n under normal, binomial or multinomial data and a correct effect; it says nothing about real data and it is not the observed power.

Usage

```
simulate_power(
  plan,
  n = NULL,
  reps = 10000,
  seed = NULL,
  predictor_type = c("fixed", "random")
)
```

Arguments

plan	A plansize_plan from plan_study(), for any of its eight tests.
n	Sample size to simulate: per group (the whole sample for "t_paired" and "cor_pearson", the total n for "chisq_indep", "reg_overall" and "reg_partial"), before dropout and design effect. Regressions need at least predictors + 2. Default: the analytic n of the plan rounded up, not plan$n_per_group. Dropout and design effect are not simulated. With your own n, status compares only with the formula (the target rule belongs to the planned n) and a note says so.
reps	Number of simulated studies, a whole number of at least 100. The rule of the package needs at least 10000; with fewer, pass and status are NA because the gap can be noise.
seed	Optional whole number. With a seed the function leaves your random number stream unchanged; without one it uses and advances your stream like any random R function.
predictor_type	"fixed" (default, the pwrss model of the plan) or "random" (predictors drawn anew in every study), for "reg_overall" and "reg_partial" only. With "random", "reg_overall" is compared with the DMAR random-predictor power (without the target rule, since the plan n is for fixed predictors); without DMAR, and always for "reg_partial", status is NA.

Details

status is "pass" when empirical power is within 0.02 of the analytic power and at least the target minus 0.02, "warn" when it is more than 0.02 above the analytic power (the plan is safe; the formula is conservative at this n) and "fail" otherwise. "warn" is expected for two-sided correlations from r = .6 and one-sided from r = .5 (Fisher z is conservative; for example a correlation of .8 with n = 10), and for two proportions with a small p (such as .05 and .01), where exact binomial power exceeds the normal approximation, and for a small chi-square table with w from about .5 (such as 2 x 2 or 3 x 3 with w = .5, or 2 x 3 with w = .8 and n = 16). Do not widen the 0.02.

Value

A plansize_simulation list: test, n, reps, seed, predictor_type, power_sim, mc_se, ci, power_analytic, ref (reference is "pwrss" or "dmar"; notes are codes the print method explains).

---

write_methods *Write a methods paragraph*

Description

Turns a plan into a copyable methods paragraph in English or Turkish.

Usage

```
write_methods(plan, lang = NULL)
```

Arguments

plan	A plansize_plan from plan_study() or a plansize_precision from plan_precision().
lang	"en" or "tr". Defaults to the plansize.lang option, then "en".

Details

The paragraph is valid only for a plan made before data collection.

Value

A single string of class plansize_methods.

Index

plan_precision, 2
plan_precision(), *7*, *9*, *11*
plan_study, 3
plan_study(), *7*, *9–11*
print.plansize_methods, 6
print.plansize_plan, 6
print.plansize_precision, 7
print.plansize_sensitivity, 7
print.plansize_simulation, 8

sensitivity, 8
sensitivity(), *2*, *5*, *8*
simulate_power, 9
simulate_power(), *8*

write_methods, 10
write_methods(), *6*