

Package ‘OptOTrials’

September 23, 2026

Type Package

Title Optimal Two-Stage Designs for Ordered Categorical Outcomes

Version 1.1.0

Date 2026-08-28

Description Functions to design and simulate optimal two-stage randomized controlled trials (RCTs) with ordered categorical outcomes, using rank-based tests and interim decision rules for futility and superiority, following Park (2025) <[doi:10.1371/journal.pone.0318211](https://doi.org/10.1371/journal.pone.0318211)>. The functions 'rule()', 'op()' and 'design_table()' provide a single entry point for constructing designs, evaluating their operating characteristics, and tabulating several designs at once. The separate functions provided for each combination of test statistic and stopping rule up to version 1.0.2 were deprecated in 1.0.3 and are removed here. Please see the package reference manual and the vignette for details.

License GPL-3

Encoding UTF-8

Depends R (>= 4.0)

Imports stats

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

RoxygenNote 7.3.2

NeedsCompilation no

Author Yeonhee Park [aut, cre],
Yudi Wang [aut],
Zhanpeng Xu [aut]

Maintainer Yeonhee Park <yeonheepark@skku.edu>

Repository CRAN

Date/Publication 2026-09-23 02:30:02 UTC

Contents

OptOTrials-package	2
design_table	3
op	4
p2_fun	6
Proportional_odds_assumption	7
QR_fun	8
rule	8
theta	11
V_S.over.nk	12
W_W	13
Index	14

OptOTrials-package	<i>Optimal Two-Stage Designs for Ordered Categorical Outcomes</i>
--------------------	---

Description

Functions to design and simulate optimal two-stage randomized controlled trials (RCTs) with ordered categorical outcomes, using rank-based tests and interim decision rules for futility and superiority, following Park (2025) <doi:10.1371/journal.pone.0318211>. The functions 'rule()', 'op()' and 'design_table()' provide a single entry point for constructing designs, evaluating their operating characteristics, and tabulating several designs at once. The separate functions provided for each combination of test statistic and stopping rule up to version 1.0.2 were deprecated in 1.0.3 and are removed here. Please see the package reference manual and the vignette for details.

Details

The workflow runs through two functions. `rule` constructs a decision rule: the test statistic, the monitoring scheme and the optimality criterion are chosen by argument, and the result is a named object with a `print` method. `op` takes that object and estimates the operating characteristics by simulation, reporting the type I error rate, the power, the interim stopping probabilities and the expected sample size under both hypotheses in a single call.

`design_table` sweeps a set of tests and criteria and returns a data frame with one row per design. `Proportional_odds_assumption` checks whether the score test is appropriate, `p2_fun` constructs the experimental-group probabilities from a specified odds ratio, and `theta` returns the win probability.

For advanced calculations, `V_S.over.nk`, `QR_fun`, and `W_W` evaluate the asymptotic variance quantities for the score, Mann–Whitney–Wilcoxon, and win odds statistics, respectively. These helpers reproduce the Appendix calculations in the accompanying article and permit boundary calculations at a prespecified sample size.

Versions up to 1.0.2 provided a separate function for each combination of test and stopping rule. Those functions were deprecated in 1.0.3 and removed in 1.1.0; `NEWS.md` and the vignette tabulate the correspondence.

Author(s)

Yeonhee Park [aut, cre], Yudi Wang [aut], Zhanpeng Xu [aut]
Maintainer: Yeonhee Park <yeonheepark@skku.edu>

References

Park, Y. (2025). Optimal two-stage group sequential designs based on Mann-Whitney-Wilcoxon test. PloS one, 20(2), e0318211.

design_table	<i>Table of designs and operating characteristics</i>
--------------	---

Description

Builds a data frame of decision rules and simulated operating characteristics over a set of tests and optimality criteria, replacing the `expand.grid()` and `apply()` idiom that required positional indexing of returned vectors. The random number generator is re-seeded for every row, so each row is reproducible on its own.

Usage

```
design_table(alpha, beta, p1, p2, tests, criteria,
             stopping = c("F", "FS"), lambda = 1, nsim = 10000,
             seed = 1234, on_degenerate = c("fallback", "warn", "error", "none"),
             digits = NULL)
```

Arguments

alpha, beta, p1, p2, lambda	As in rule .
tests	Character vector of tests, a subset of <code>c("S", "M", "W")</code> .
criteria	Numeric vector of optimality criteria, a subset of 1 to 5.
stopping	"F" or "FS".
nsim	Number of simulation replicates per row.
seed	Seed applied before each scenario of each row.
on_degenerate	Passed to rule , with the same default, "fallback": a degenerate optimum is replaced by the corresponding single-stage design and the substitution is reported. Set to "none" to report the raw optimum instead, in which case the degenerate column flags the affected rows and their operating characteristics are returned as NA, because a stage-1 total below two patients cannot be allocated to two arms and so cannot be simulated. That is the only case in which the operating characteristics are omitted; any other error from op is passed on.
digits	Optional number of digits for rounding the estimated probabilities. When supplied, EN0 and ENa are rounded to two decimals and EN is recomputed from the rounded values, so that a reader checking $EN = (EN0 + ENa)/2$ against the printed table reproduces the printed EN exactly. The default NULL returns full precision.

Value

A data frame with one row per (test, criterion) combination, containing the decision rule, the estimated type I error rate and power with Monte Carlo standard errors, the futility and superiority stopping probabilities under each hypothesis, the expected sample sizes, and a degenerate flag.

See Also

[rule](#), [op](#)

Examples

```
p1 <- c(1/3, 1/3, 1/3)
p2 <- c(1/2, 1/3, 1/6)
design_table(0.05, 0.2, p1, p2, tests = "M", criteria = 1,
            stopping = "FS", nsim = 500)
```

op	<i>Operating characteristics of a design</i>
----	--

Description

Estimates the operating characteristics of a design produced by [rule](#) by Monte Carlo simulation. Both hypotheses are evaluated in a single call, and every returned quantity is named, so the meaning of the output does not depend on how the function was called.

Usage

```
op(design, nsim = 10000, seed = NULL, p1 = NULL, p2 = NULL,
   allow_degenerate = FALSE)
```

Arguments

design	An object returned by rule .
nsim	Number of simulation replicates (default 10000).
seed	Optional integer. If supplied, the random number generator is seeded immediately before each scenario, so the result is reproducible independently of the state of the stream when <code>op()</code> is reached.
p1, p2	Optional category probabilities overriding those stored in <code>design</code> , for evaluating a design under a scenario other than the one it was built for. Both vectors are validated on every call: each must be numeric, cover at least two outcome categories, contain no missing or negative values, and sum to one within a tolerance of 1e-6; the two must have the same length. A violation raises an error naming the condition that failed. The tolerance admits floating-point representation error, so exact fractions such as <code>c(1/3, 1/3, 1/3)</code> pass, while a transcription mistake does not.

`allow_degenerate`

Logical. A degenerate two-stage optimum that has not been replaced by its single-stage substitute cannot be simulated meaningfully, because its stage-1 size is too small to allocate patients to both arms; such a design is refused by default. Set to TRUE to override the check.

Value

An object of class "OptOTrialsOC" containing alpha and power with their Monte Carlo standard errors `se_alpha` and `se_power`, the expected sample sizes `EN0`, `ENa` and `EN`, and a data frame table reporting, under each hypothesis, the rejection probability, the probability of stopping early for futility, the probability of stopping early for superiority, the probability of continuing to the second stage, and the expected sample size. Futility and superiority stopping are reported separately rather than pooled.

See Also

[rule, design_table](#)

Examples

```
p1 <- c(0.075, 0.182, 0.319, 0.243, 0.015, 0.166)
p2 <- p2_fun(p1, log(3.06))
d <- rule(0.05, 0.2, p1, p2, test = "M", stopping = "F", criterion = 1)

## Printing gives an aligned table: both hypotheses in one call, with the
## futility and superiority stopping probabilities reported separately.
o <- op(d, nsim = 1000, seed = 1234)
o                                     # same as print(o)

## Every quantity is named, so which is which does not depend on how the
## function was called.
o$alpha                             # type I error rate, with
o$se_alpha                           # its Monte Carlo standard error
o$power
o$se_power
c(o$EN0, o$ENa, o$EN)                # expected sample sizes
o$table                             # the full table, one row per hypothesis
names(o)

## The seed is applied to each hypothesis separately, so a call reproduces
## on its own whatever the state of the random number stream.
runif(100)
identical(op(d, nsim = 1000, seed = 1234)$power, o$power)

## Evaluating the same design under a different alternative
op(d, nsim = 1000, seed = 1234, p2 = p2_fun(p1, log(2)))$power
```

p2_fun

*Experimental-group probabilities from a specified odds ratio***Description**

Constructs the category probabilities for the experimental arm from those of the control arm and a specified log odds ratio, under the proportional odds model. It is a convenience for setting up an alternative hypothesis in terms of a clinically meaningful effect size rather than by writing out a second probability vector.

Usage

```
p2_fun(p1, theta)
```

Arguments

p1	A vector containing the probabilities of the outcome falling into each level for the control arm. Validated on every call: numeric, at least two categories, no missing or negative values, and summing to one within 1e-6.
theta	The log odds ratio representing the expected effect of the experimental treatment. A single finite number.

Value

A numeric vector of the same length as p1, giving the probability of each outcome level for the *experimental* arm.

See Also

[Proportional_odds_assumption](#) to check that a pair of vectors is consistent with a common odds ratio.

Examples

```
## Example 1 of the article: six ordered categories, odds ratio 3.06
p1 <- c(0.075, 0.182, 0.319, 0.243, 0.015, 0.166)
p2 <- p2_fun(p1, log(3.06))
p2
sum(p2)

## The construction is consistent by definition, so the diagnostic recovers
## the odds ratio it was given
exp(Proportional_odds_assumption(p1, p2, verbose = FALSE))
```

`Proportional_odds_assumption`*Checking proportional odds assumption*

Description

This is the function to check the proportional odds assumption for the score test.

Usage

```
Proportional_odds_assumption(p1, p2, verbose = TRUE)
```

Arguments

<code>p1</code>	A vector containing the probabilities of the outcome falling into each level of the control arm.
<code>p2</code>	A vector containing the probabilities of the outcome falling into each level of the experimental arm.
<code>verbose</code>	Logical. If TRUE (the default) an explanatory message is printed stating whether the assumption holds and, if so, the implied common odds ratio.

Validated as for [rule](#): numeric, equal length, at least two categories, no missing or negative values, and summing to one within 1e-6.

Value

Indicates whether the proportional odds assumption holds. If the assumption holds, the function returns the log-odds ratio from the score test. If the assumption does not hold, the function returns NA and additionally emits a message and a warning explaining that the score test is not advisable and suggesting the Mann-Whitney-Wilcoxon or win odds test, so that the condition is not missed in scripted use.

Examples

```
## The assumption holds when p2 is built from p1 by a common odds ratio
p1 <- c(0.075, 0.182, 0.319, 0.243, 0.015, 0.166)
p2 <- p2_fun(p1, log(3.06))
Proportional_odds_assumption(p1, p2)

## and fails otherwise, with a message and a warning rather than a bare NA
q1 <- c(1/3, 1/3, 1/3)
q2 <- c(1/2, 1/3, 1/6)
try(Proportional_odds_assumption(q1, q2))
```

QR_fun

Variance component of the Mann-Whitney-Wilcoxon statistic

Description

Returns the quantity Q (equivalently R , by symmetry of the arguments) appearing in the asymptotic variance of the Mann-Whitney-Wilcoxon statistic, $\text{Var}(T_k^{(M)}) \approx Q/n_{1\cdot k} + R/n_{2\cdot k}$.

Usage

```
QR_fun(p1, p2)
```

Arguments

p1 A vector of outcome-category probabilities for the control arm.

p2 A vector of outcome-category probabilities for the experimental arm; supply p1 twice for the null value.

Value

A single number.

See Also

[V_S.over.nk, W_W.](#)

Examples

```
p1 <- c(0.075, 0.182, 0.319, 0.243, 0.015, 0.166)
lambda <- 1

## Null variance of the statistic at a total size of 28, and the threshold it
## implies for a nominal one-sided level of 0.05
v <- ((1 + lambda) * QR_fun(p1, p1) + (1 + 1/lambda) * QR_fun(p1, p1)) / 28
qnorm(0.05, lower.tail = FALSE) * sqrt(v)
```

rule

Optimal design for a trial with an ordered categorical outcome

Description

Constructs the decision rule for a one- or two-stage randomised trial with an ordered categorical outcome. This is the recommended entry point to the package: the test statistic and the monitoring scheme are selected by argument, so a single function replaces the eleven separate decision-rule functions provided up to version 1.0.2.

Usage

```
rule(alpha, beta, p1, p2, test = c("S", "M", "W"),
      stopping = c("F", "FS", "none"), criterion = 1, lambda = 1,
      on_degenerate = c("fallback", "warn", "error", "none"), min_n1 = 10)
```

Arguments

alpha	Target type I error rate.
beta	Target type II error rate; power is 1 - beta.
p1	Numeric vector of outcome category probabilities for the control group. Must be non-negative and sum to 1.
p2	Numeric vector of outcome category probabilities for the experimental group. Must be the same length as p1, non-negative, and sum to 1. Both vectors are validated on every call: each must be numeric, cover at least two outcome categories, contain no missing or negative values, and sum to one within a tolerance of 1e-6; the two must have the same length. A violation raises an error naming the condition that failed. The tolerance admits floating-point representation error, so exact fractions such as c(1/3, 1/3, 1/3) pass, while a transcription mistake does not.
test	Test statistic: "S" the score test under the proportional odds model, "M" the Mann-Whitney-Wilcoxon test, "W" the win odds test.
stopping	Monitoring scheme: "F" allows early stopping for futility only, "FS" for futility and superiority, "none" gives a single-stage design.
criterion	Optimality criterion, 1 to 5. 1: minimise the expected total sample size under the null; 2: under the alternative; 3: assuming $\Pr(H_0) = \Pr(H_a)$; 4: balance the two stages prioritising EN0; 5: balance the two stages prioritising the maximum sample size. Ignored when stopping = "none".
lambda	Ratio of experimental to control sample sizes (default 1).
on_degenerate	How to handle a degenerate optimum (see Details). "fallback" (default) warns and returns the single-stage design; "warn" warns and returns the degenerate design; "error" raises an error; "none" returns the raw optimum silently.
min_n1	Smallest stage-1 sample size regarded as usable when detecting degeneracy (default 10).

Details

Minimising the expected sample size under the alternative (criterion = 2) can drive the optimum to a design with a stage-1 sample size of 1 and an interim boundary so far into the lower tail that the interim analysis cannot change any decision: the design is then two-stage in name only. Intuitively, the criterion rewards designs under which continuation to the final analysis is almost certain when the treatment works, so the optimiser commits essentially all information to the final analysis. rule() detects this and, by default, substitutes the corresponding single-stage design, recording the substitution in the returned object rather than making it silently.

Value

An object of class "OptOTrialsRule": a list with components test, stopping, criterion, alpha, beta, lambda, p1, p2, the stage-1 size n1, the interim futility and superiority boundaries t1f and t1s, the maximum size n2, the final boundary t2, a data frame effect giving each boundary on a clinically interpretable effect-size scale (odds ratio for the score test, win odds for the rank-based tests), and the flags degenerate, degenerate_reason and substituted. A print method displays these with labels.

Note

The name rule is also used by the **cli** package, which **MAMS** and several other packages attach. If **cli** is attached after **OptOTrials**, `cli::rule` masks this function for the rest of the session and a call to `rule()` will fail with an error about unused arguments. Attach **OptOTrials** last, or call `OptOTrials::rule()` explicitly.

See Also

[op](#), [design_table](#)

Examples

```
p1 <- c(0.075, 0.182, 0.319, 0.243, 0.015, 0.166)
p2 <- p2_fun(p1, log(3.06))

## Printing the object gives a labelled summary of the decision rule,
## including each boundary on the effect-size scale.
d <- rule(0.05, 0.2, p1, p2, test = "M", stopping = "F", criterion = 1)
d                                     # same as print(d)

## The components are named, so nothing has to be indexed by position.
d$n1                                # stage-1 size
d$t1f                               # interim futility boundary
d$n2                                # cumulative size at the final analysis
d$t2                                # final boundary
d$effect                            # boundaries as win odds (or odds ratios)
names(d)

## The object is passed straight to op(); the sample sizes and boundaries
## never have to be transcribed between the two calls.
op(d, nsim = 1000, seed = 1234)

## An FS design adds an interim superiority boundary.
rule(0.05, 0.2, p1, p2, test = "M", stopping = "FS", criterion = 1)

## Minimising E(N | Ha) can return a degenerate optimum. By default it is
## detected and replaced by the single-stage design, with a warning.
d2 <- rule(0.05, 0.2, p1, p2, test = "M", stopping = "F", criterion = 2)
d2$substituted
```

theta

Win probability for two ordered categorical distributions

Description

Computes the win probability $\theta = \Pr(\eta < \xi) + 0.5 \Pr(\eta = \xi)$, the expectation of the win odds statistic, where ξ is a draw from the control distribution and η a draw from the experimental distribution. Categories are ordered from most to least desirable, so $\theta > 0.5$ favours the experimental treatment. The win odds are $\theta/(1 - \theta)$.

Usage

```
theta(p1, p2)
```

Arguments

p1	A vector containing the probabilities of the outcome falling into each level for the control arm. It must be numeric, contain at least two categories and no missing or negative values, and sum to one within 1e-6.
p2	A vector containing the probabilities of the outcome falling into each level for the experimental arm. It is validated as for p1 and must have the same length.

Value

A single number, the win probability θ .

See Also

[rule](#) for designs based on the win odds test.

Examples

```
p1 <- c(1/3, 1/3, 1/3)
p2 <- c(1/2, 1/3, 1/6)

## Win probability and the corresponding win odds
th <- theta(p1, p2)
th
th / (1 - th)

## Under the null the two distributions coincide and theta is 0.5
theta(p1, p1)
```

V_S.over.nk

Variance factor of the score statistic

Description

Returns the asymptotic variance of the score test statistic per patient, that is $\text{Var}(T_k^{(S)})/n_{..k}$, under the proportional odds model. Because the factor does not depend on the sample size, the variance at any total size n is obtained by multiplying by n , which is what makes the sample-size equations solvable in closed form.

Usage

```
V_S.over.nk(p1, p2, lambda = 1)
```

Arguments

p1	A vector of outcome-category probabilities for the control arm.
p2	A vector of outcome-category probabilities for the experimental arm; supply p1 twice for the null value.
lambda	A finite positive allocation ratio, experimental to control (default 1).

Value

A single number, the variance per patient.

See Also

[QR_fun](#) and [W_W](#) for the rank-based tests.

Examples

```
p1 <- c(0.075, 0.182, 0.319, 0.243, 0.015, 0.166)
p2 <- p2_fun(p1, log(3.06))

## Null variance factor, and the threshold a fixed total size n implies for a
## nominal one-sided level of 0.05
v <- V_S.over.nk(p1, p1)
qnorm(0.05, lower.tail = FALSE) * sqrt(v * 28)

## Under the alternative the factor differs
V_S.over.nk(p1, p2)
```

W_W

Variance factor of the win odds statistic

Description

Returns the variance factor $W_k^{(W)}$ of the win odds statistic, so that its asymptotic variance at total size n is $W_k^{(W)}/n$.

Usage

```
W_W(p1, p2, lambda = 1)
```

Arguments

p1	A vector of outcome-category probabilities for the control arm.
p2	A vector of outcome-category probabilities for the experimental arm; supply p1 twice for the null value.
lambda	A finite positive allocation ratio, experimental to control (default 1).

Value

A single number.

See Also

[theta](#) for the win probability itself, [V_S.over.nk](#) and [QR_fun](#) for the other tests.

Examples

```
p1 <- c(0.075, 0.182, 0.319, 0.243, 0.015, 0.166)

## The win odds statistic is centred at 0.5 under the null; the threshold at a
## total size of 28 for a nominal one-sided level of 0.05 is
0.5 + qnorm(0.05, lower.tail = FALSE) * sqrt(W_W(p1, p1) / 28)
```

Index

`design_table`, [2](#), [3](#), [5](#), [10](#)

`op`, [2–4](#), [4](#), [10](#)

`OptOTrials` (`OptOTrials`-package), [2](#)

`OptOTrials`-package, [2](#)

`p2_fun`, [2](#), [6](#)

`print.OptOTrialsOC` (`op`), [4](#)

`print.OptOTrialsRule` (`rule`), [8](#)

`Proportional_odds_assumption`, [2](#), [6](#), [7](#)

`QR_fun`, [2](#), [8](#), [12](#), [13](#)

`rule`, [2–5](#), [7](#), [8](#), [11](#)

`theta`, [2](#), [11](#), [13](#)

`V_S.over.nk`, [2](#), [8](#), [12](#), [13](#)

`W_W`, [2](#), [8](#), [12](#), [13](#)