Package 'adaptDiag'

June 28, 2026

Type Package

Title Bayesian Adaptive Designs for Diagnostic Trials

Version 0.1.1

Description Simulate clinical trials for diagnostic test devices and evaluate the operating characteristics under an adaptive design with futility assessment determined via the posterior predictive probabilities.

License GPL-3

Encoding UTF-8

URL https://graemeleehickey.github.io/adaptDiag/, https://github.com/graemeleehickey/adaptDiag, https://CRAN.R-project.org/package=adaptDiag

BugReports https://github.com/graemeleehickey/adaptDiag/issues

Depends R (>= 3.6.0)

Imports doParallel, doRNG, extraDistr, foreach, parallel, stats

Language en-US

RoxygenNote 7.3.3

Suggests rmarkdown, knitr, testthat (>= 3.0.0), VGAM, covr

Config/testthat/edition 3

VignetteBuilder knitr

NeedsCompilation no

Author Graeme L. Hickey [cre, aut],
Yongqiang Zhang [aut],
Becton, Dickinson and Company [cph]

Maintainer Graeme L. Hickey <graemeleehickey@gmail.com>

Repository CRAN

Date/Publication 2026-06-28 19:20:02 UTC

Contents

binom_sample_size 2
multi_trial 3
summarise_trials 7

Index **9**

`binom_sample_size` *Calculate the minimum number of samples required for a one-sided exact binomial test*

Description

Calculate the minimum number of samples required for a one-sided exact binomial test to distinguish between two success probabilities with specified alpha and power.

Usage

```
binom_sample_size(alpha = 0.05, power = 0.9, p0 = 0.9, p1 = 0.95)
```

Arguments

`alpha`	scalar. The desired false positive rate (probability of incorrectly rejecting the null). Must be be between 0 and 1. Default value is `alpha = 0.05`.
`power`	scalar. The minimum probability of correctly rejecting the null when the alternate is true.
`p0`	scalar. The expected proportion of successes under the null.
`p1`	scalar. The proportion of successes under the alternate hypothesis.

Details

This is a one-sided function, such that $p_0 < p_1$. It determines the minimum sample size to evaluate the hypothesis test:

$$H_0 : p_1 \leq p_0, \, vs.$$

$$H_1 : p_1 > p_0$$

Value

A list containing the required sample size and the number of successful trials required.

References

Chow S-C, Shao J, Wang H, Lokhnygina Y. (2017) *Sample Size Calculations in Clinical Research*, Boca Raton, FL: CRC Press.

Examples

```
# The minimum number of reference positive cases required to demonstrate
# the true sensitivity is >0.7, assuming that the true value is 0.824, with
# 90% power is

binom_sample_size(alpha = 0.05, power = 0.9, p0 = 0.7, p1 = 0.824)

# With a sample size of n = 104, if the true prevalence is 0.2, we would
# require a sample size of at least n = 520 randomly sampled subjects to
# have adequate power to demonstrate the sensitivity of the new test.

# The minimum number of reference negative cases required to demonstrate
# the true specificity is >0.9, assuming that the true value is 0.963, with
# 90% power is

binom_sample_size(alpha = 0.05, power = 0.9, p0 = 0.9, p1 = 0.963)

# The proposed total sample size of n = 520 would be sufficient to
# demonstrate both endpoint goals are met.
```

---

multi_trial *Simulate and analyse multiple trials*

Description

Multiple trials and simulated and analysed up to the final analysis stage, irrespective of whether it would have been stopped for early success or expected futility. The output of the trials is handled elsewhere.

Usage

```
multi_trial(
  sens_true,
  spec_true,
  prev_true,
  endpoint = "both",
  sens_pg = 0.8,
  spec_pg = 0.8,
  prior_sens = c(0.1, 0.1),
  prior_spec = c(0.1, 0.1),
  prior_prev = c(0.1, 0.1),
  succ_sens = 0.95,
  succ_spec = 0.95,
  n_at_looks,
  n_mc = 10000,
  n_trials = 1000,
  seed = NULL,
```

```
  ncores
)
```

Arguments

sens_true	scalar. True assumed sensitivity (must be between 0 and 1).
spec_true	scalar. True assumed specificity (must be between 0 and 1).
prev_true	scalar. True assumed prevalence as measured by the gold-standard reference test (must be between 0 and 1).
endpoint	character. The endpoint(s) that must meet a performance goal criterion. The default is `endpoint = "both"`, which means that the endpoint is based simultaneously on sensitivity and specificity. Alternative options are to specify `endpoint = "sens"` or `endpoint = "spec"` for sensitivity and specificity, respectively. If only a single endpoint is selected (e.g. sensitivity), then the PG and success probability threshold of the other statistic are set to 1, and ignored for later analysis.
sens_pg	scalar. Performance goal (PG) for the sensitivity endpoint, such that the posterior probability that the PG is exceeded is calculated. Must be between 0 and 1.
spec_pg	scalar. Performance goal (PG) for the specificity endpoint, such that the posterior probability that the PG is exceeded is calculated. Must be between 0 and 1.
prior_sens	vector. A vector of length 2 specifying the shape parameters (α, β) of the Beta prior for sensitivity. `c(1, 1)` gives a uniform (non-informative) prior. `c(0.1, 0.1)` is a near-Jeffreys prior that is bimodal, placing most mass near 0 and 1; this can make the posterior sensitive to early data. An informative prior derived from a pilot study is recommended when available.
prior_spec	vector. A vector of length 2 specifying the shape parameters (α, β) of the Beta prior for specificity. See `prior_sens` for guidance on prior choice.
prior_prev	vector. A vector of length 2 specifying the shape parameters (α, β) of the Beta prior for prevalence. See `prior_sens` for guidance on prior choice.
succ_sens	scalar. Probability threshold for the sensitivity to exceed in order to declare a success. Must be between 0 and 1.
succ_spec	scalar. Probability threshold for the specificity to exceed in order to declare a success. Must be between 0 and 1.
n_at_looks	vector. Sample sizes for each interim look. The final value (or only value if no interim looks are planned) is the maximum allowable sample size for the trial.
n_mc	integer. Number of Monte Carlo draws to use for sampling from the Beta-Binomial distribution.
n_trials	integer. The number of clinical trials to simulate overall, which will be used to evaluate the operating characteristics.
seed	integer. Optional random seed passed to `set.seed` before simulations begin, to ensure reproducibility. Default is `NULL` (no seed set).
ncores	integer. The number of cores to use for parallel processing. If ‘ncores‘ is missing, it defaults to the maximum number of cores available (spare 1).

Details

This function simulates multiple trials and analyses each stage of the trial (i.e. at each interim analysis sample size look) irrespective of whether a stopping rule was triggered or not. The operating characteristics are handled by a separate function, which accounts for the stopping rules and any other trial constraints. By enumerating each stage of the trial, additional insights can be gained such as: for a trial that stopped early for futility, what is the probability that it would eventually go on to be successful if the trial had not stopped. The details on how each trial are simulated here are described below.

Simulating a single trial

Given true values for the test sensitivity (`sens_true`), specificity (`spec_true`), and the prevalence (`prev_true`) of disease, along with a sample size look strategy (`n_at_looks`), it is straightforward to simulate a complete dataset using the binomial distribution. That is, a data frame with true disease status (reference test), and the new diagnostic test result.

Posterior probability of exceeding PG at current look

At a given sample size look, the posterior probability of an endpoint (e.g. sensitivity) exceeding the pre-specified PG (`sens_pg`) can be calculated as follows.

If we let θ be the test property of interest (e.g. sensitivity), and if we assume a prior distribution of the form

$$\theta \; Beta(\alpha, \beta),$$

then with $X|\theta \sim Bin(n, \theta)$, where X is the number of new test positive cases from the reference positive cases, the posterior distribution of θ is

$$\theta|X = x \; Beta(\alpha + x, \beta + n - x).$$

The posterior probability of exceeding the PG is then calculated as

$P[\theta \geq sens_pg|X = x, n]$.

A similar calculation can be performed for the specificity, with corresponding PG, `spec_pg`.

Posterior predictive probability of eventual success

When at an interim sample size that is less the maximum (i.e. `max(n_at_looks)`), we can calculate the probability that the trial will go on to eventually meet the success criteria.

At the j-th look, we have observed n_j tests, with $n_j^* = n_{max} - n_j$ subjects yet to be enrolled for testing. For the n_j^* subjects remaining, we can simulate the number of reference positive results, y_j^*, using the posterior predictive distribution for the prevalence (reference positive tests), which is off the form

$$y_j^*|y_j, n_j, n_j^* \; Beta - Bin(n_j^*, \alpha_0 + y_j, \beta + n_j - y_j),$$

where y_j is the observed number of reference positive cases. Conditional on the number of subjects with a positive reference test in the remaining sample together with n_j^*, one can simulate the complete 2x2 contingency table by using the posterior predictive distributions for sensitivity and specificity, each of which has a Beta-Binomial form. Combining the observed n_j subjects' data with a sample of the n_j^* subjects' data drawn from the predictive distribution, one can then calculate the posterior probability of trial success (exceeding a PG) for a specific endpoint. Repeating

this many times and calculating the proportion of probabilities that exceed the probability success threshold yields the probability of eventual trial success at the maximum sample size.

As well as calculating the predictive posterior probability of eventual success for sensitivity and specificity, separately, we can also calculate the probability for both endpoints simultaneously.

Value

A list containing a data frame with rows for each stage of the trial (i.e. each sample size look), irrespective of whether the trial meets the stopping criteria. Multiple trial simulations are stacked longways and indicated by the 'trial' column. The data frame has the following columns:

- `stage`: Trial stage.
- `pp_sens`: Posterior probability of exceeding the performance goal for sensitivity.
- `pp_spec`: Posterior probability of exceeding the performance goal for specificity.
- `ppp_succ_sens`: Posterior predictive probability of eventual success for sensitivity at the maximum sample size.
- `ppp_succ_spec`: Posterior predictive probability of eventual success for specificity at the maximum sample size.
- `ppp_succ_both`: Posterior predictive probability of eventual success for *both* sensitivity and specificity at the maximum sample size.
- `tp`: True positive count.
- `tn`: True negative count.
- `fp`: False positive count.
- `fn`: False negative count.
- `sens_hat`: Posterior median estimate of the test sensitivity.
- `sens_CrI2.5`: Lower bound of the 95 the test sensitivity.
- `sens_CrI97.5`: Upper bound of the 95 the test sensitivity.
- `spec_hat`: Posterior median estimate of the test specificity.
- `spec_CrI2.5`: Lower bound of the 95 the test specificity.
- `spec_CrI97.5`: Upper bound of the 95 the test specificity.
- `n`: The sample size at the given look for the row.
- `trial`: The trial number, which will range from 1 to 'n_trials'.

The list also contains the arguments used and the call.

Parallelization

To use multiple cores (where available), the argument `ncores` can be increased from the default of 1. Parallelization uses `registerDoParallel` with a PSOCK cluster, which works on all platforms including Windows. The `%dorng%` operator from the **doRNG** package is used in place of the standard `%dopar%`, ensuring that results are fully reproducible across all backends when a `seed` is supplied.

Examples

```
multi_trial(
  sens_true = 0.9,
  spec_true = 0.95,
  prev_true = 0.1,
  endpoint = "both",
  sens_pg = 0.8,
  spec_pg = 0.8,
  prior_sens = c(0.1, 0.1),
  prior_spec = c(0.1, 0.1),
  prior_prev = c(0.1, 0.1),
  succ_sens = 0.95,
  succ_spec = 0.95,
  n_at_looks = c(200, 400, 600, 800, 1000),
  n_mc = 10000,
  n_trials = 2,
  ncores = 1
)
```

`summarise_trials` *Summarise results of multiple simulated trials to give the operating characteristics*

Description

Summarise results of multiple simulated trials to give the operating characteristics

Usage

```
summarise_trials(data, min_pos = 1, fut = 0)
```

Arguments

`data`	list. Output from the `multi_trial` function.
`min_pos`	integer. The minimum number of reference positive cases before stopping is allowed. Default is `min_pos = 1`.
`fut`	scalar. A probability threshold at which the posterior predictive probability of eventual success is compared to. If the probability is less than `fut`, the trial stops for binding futility. Default is `fut = 0`, which corresponds to no stopping for futility.

Value

A data frame of row length 1, with the following columns:

- power: Power is defined as the proportion of trials that result in success, irrespective of whether it is an early stop for success or not. Trials that stop for futility, but which subsequently go on to be successful, are not considered as a success. In other words, the futility decision is binding, and in practice, if a trial triggered a futility rule, the sponsor would not see the eventual outcome if the trial were to continue enrolling. When the performance goals are set equal to the respective true values, the power returned is the type I error.
- stop_futility: The proportion of trials that stopped early for expected futility.
- n_avg: The average sample size for trials at the stage they stopped.
- sens: The average sensitivity for trials at the stage they stopped.
- spec: The average specificity for trials at the stage they stopped.
- mean_pos: The average number of reference positive cases for trials at the stage they stopped.

Examples

```
data <- multi_trial(
    sens_true = 0.9,
    spec_true = 0.95,
    prev_true = 0.1,
    endpoint = "both",
    sens_pg = 0.8,
    spec_pg = 0.8,
    prior_sens = c(1, 1),
    prior_spec = c(1, 1),
    prior_prev = c(1, 1),
    succ_sens = 0.95,
    succ_spec = 0.95,
    n_at_looks = c(200, 400, 600, 800, 1000),
    n_mc = 10000,
    n_trials = 20,
    ncores = 1
    )

summarise_trials(data, fut = 0.05, min_pos = 10)
```

Index

%dorng%, *6*

binom_sample_size, 2

multi_trial, 3, *7*

registerDoParallel, *6*

set.seed, *4*
summarise_trials, 7