# Package 'RCTCovAdj'

September 24, 2026

**Type** Package

**Title** Covariate Adjustment for Randomized Controlled Trials

**Version** 0.1.0

**Description** Implements covariate-adjusted estimation and inference for marginal mean differences in two-arm randomized controlled trials with continuous outcomes. Methods include the unadjusted difference in means, common-slope analysis of covariance, interacted standardization with joint influence-function inference, and cross-fitted augmentation. The package also provides analytic power and sample-size calculations, reproducible simulation designs, and article replication workflows, including tools for verifying and analyzing an authorized local trial-data file. The adjustment methods build on Tsiatis et al. (2008) <doi:10.1002/sim.3113> and Lin (2013) <doi:10.1214/12-AOAS583>.

**License** MIT + file LICENSE

**Encoding** UTF-8

**Depends** R (>= 4.1.0)

**Imports** digest, graphics, grDevices, parallel, stats, tools, utils

**Suggests** knitr, medicaldata (>= 0.2.0), rmarkdown, testthat (>= 3.0.0), withr

**VignetteBuilder** knitr

**Config/testthat/edition** 3

**LazyData** true

**LazyDataCompression** xz

**RoxygenNote** 7.3.3

**NeedsCompilation** no

**Author** Se Yoon Lee [aut, cre, cph]

**Maintainer** Se Yoon Lee <seyoonlee.stat.math@gmail.com>

**Repository** CRAN

**Date/Publication** 2026-09-24 14:50:11 UTC

# Contents



---

`RCTCovAdj-package` *RCTCovAdj: Covariate Adjustment for Randomized Controlled Trials*

---

## Description

RCTCovAdj implements estimation, inference, simulation, and prospective design calculations for marginal mean differences in two-arm randomized controlled trials with continuous outcomes.

## Main functions

- `rct_adjust()` computes unadjusted, common-slope, and interacted estimates.
- `rct_crossfit()` computes cross-fitted augmented estimates.
- `rct_power()` and `rct_sample_size()` connect variance to trial design.
- `run_paper_simulations()` reproduces the article's Monte Carlo study.
- `analyze_licorice()` implements the real-data analysis for an authorized copy of the licorice-gargle trial records.

The package distributes neither the trial records nor results derived from them. Data access, analysis, publication, and redistribution require the permission of the relevant data rights holder.

### Author(s)

**Maintainer**: Se Yoon Lee <seyoonlee.stat.math@gmail.com> [copyright holder]

---

`analyze_licorice` *Analyze the Licorice-Gargle Trial Application*

---

### Description

Reproduces the observed-score analyses and the bounded two-score completion exercise in the article. The function analyzes the 233 stored records with observed 30-minute throat-pain scores. It then assigns every requested pair of values to the two unavailable scores among all 235 stored records and refits `DIM` (difference in means), `ANCOVA-HC0` (common-slope analysis of covariance with heteroskedasticity-consistent type 0 inference), `ANHECOVA-joint-IF` (interacted standardization with joint influence-function inference), and `ANHECOVA-coefficient-HC2` (the coefficient-only type 2 diagnostic).

### Usage

```
analyze_licorice(data, completion_values = 0:10)
```

### Arguments

| | |
|---|---|
| `data` | The verified or prepared 235-record analysis data frame. |
| `completion_values` | Nonempty numeric vector of distinct finite values used for each unavailable score. |

### Details

The returned normal-reference intervals are working calculations. They do not recover the participant absent from the file, identify the effect among all 236 randomized participants without additional assumptions, or adjust for the trial's interim monitoring.

### Value

An object of class "licorice_analysis" containing estimates, outcome summaries, baseline summaries, the completion grid and ranges, model diagnostics, arm-specific coefficients, and the underlying adjustment object.

### Examples

```
# Synthetic records illustrate the analysis; these are not trial results.
set.seed(42)
n <- 235L
d <- data.frame(
  participant_id = seq_len(n),
  tx = rep(0:1, c(117, 118)),
  age_bl = runif(n, 18, 80),
```

```
  bmi_bl = runif(n, 20, 35),
  gender = factor(sample(0:1, n, replace = TRUE),
                  levels = 0:1, labels = c("0. Female", "1. Male")),
  asa_physical_status_bl = factor(sample(1:3, n, replace = TRUE),
    levels = 1:3, labels = c("1. Healthy", "2. Mild Systemic Disease",
                             "3. Severe Systemic Disease")),
  mallampati_bl = factor(sample(1:3, n, replace = TRUE),
    levels = 1:3, labels = c("1. 1", "2. 2", "3. 3+")),
  pacu_30_min_throat_pain = sample(0:10, n, replace = TRUE)
)
d$pacu_30_min_throat_pain[c(1, 118)] <- NA
# Four completions keep this example short; the full grid uses 0:10.
result <- analyze_licorice(d, completion_values = c(0, 10))
result$estimates
```

---

`paper_population_benchmarks` *Population Benchmarks for the Simulation Cases*

---

## Description

Exact variance factors and projection slopes for the four simulation laws.

## Usage

```
paper_population_benchmarks
```

## Format

A data frame with 4 rows and 13 variables: case labels and parameters, unadjusted variance VU, efficiency bound Vstar, interacted linear variance Vlinear, pooled analysis of covariance (ANCOVA) variance Vancova, coefficient-only heteroskedasticity-consistent type 2 (HC2) variance limit Vhc2_limit, and the optimal and pooled slopes.

## Source

Calculated analytically from simulation_cases.

## paper_power_design *Reproduce the Paper's Power and Sample-Size Application*

### Description

Builds the article's deterministic two-sided normal-approximation calculations for simulation Cases A and C. The result contains the planning table and power curves for the analyses compared in each case.

### Usage

```
paper_power_design(
  effect = 0.5,
  alpha = 0.05,
  target_powers = c(0.8, 0.9),
  n_grid = seq.int(20L, 1600L)
)
```

### Arguments

| | |
|---|---|
| effect | Planning marginal mean difference. |
| alpha | Two-sided type I error probability. |
| target_powers | Target powers used for sample-size calculations. |
| n_grid | Total sample sizes used to construct power curves. |

### Value

An object of class "rct_power_design" with components sample_sizes, curves, and settings.

### Examples

```
design <- paper_power_design(n_grid = seq(50, 1200, by = 10))
design$sample_sizes
```

## paper_power_design_results *Power and Sample-Size Results from the Article*

### Description

Exact two-sided normal-approximation calculations for analytic Cases A and C at a marginal mean difference of 0.5 and level 0.05.

## Usage

```
paper_power_design_results
```

## Format

A data frame with 6 rows and 12 variables, including the variance factor, relative variance, smallest total sample sizes attaining 80 and 90 percent power, achieved powers after rounding, and power at total sample sizes 300, 500, and 800.

## Source

Calculated by `paper_power_design()`.

---

# paper_simulation_results

*Monte Carlo Results from the Article*

---

## Description

Aggregate results from 4,000 replications in each of four cases and two sample sizes. Replicate-level records are omitted from the package because they can be regenerated with `run_paper_simulations()`.

## Usage

```
paper_simulation_results
```

## Format

A data frame with 56 rows and 23 variables. It contains the case, sample size, allocation probability, method, replication count, bias and its Monte Carlo standard error, empirical and estimated standard errors, coverage, normalized variance summaries, population limits, and numerical safeguard counts.

## Source

Generated by the article's simulation workflow with master seed 20260903 and L'Ecuyer-CMRG random-number streams.

`plot_licorice_results` *Plot Aggregate Results from the Licorice Trial Application*

## Description

Draws the observed score distribution by assigned arm alongside the three distinct mean-difference estimates, their working intervals, and the two-score completion ranges.

## Usage

```
plot_licorice_results(x, ...)
```

## Arguments

| | |
|---|---|
| `x` | A result from `analyze_licorice()` or an aggregate result list with the same plotting components. |
| `...` | Unused. |

## Value

The input object, invisibly.

## Examples

```
# Entirely fabricated aggregate values, not results from the trial.
distribution <- expand.grid(score = 0:10, arm = 0:1)
distribution$arm_label <- rep(c("Sugar", "Licorice"), each = 11)
distribution$count <- c(8, 2, rep(0, 9), 9, 1, rep(0, 9))
distribution$n_observed <- 10
distribution$proportion <- distribution$count / 10
methods <- c("DIM", "ANCOVA-HC0", "ANHECOVA-joint-IF")
x <- list(
  outcome_distribution = distribution,
  estimates = data.frame(method = methods,
    estimate = c(-0.10, -0.12, -0.11),
    lower_working = c(-0.40, -0.39, -0.37),
    upper_working = c(0.20, 0.15, 0.15)),
  completion_summary = data.frame(method = methods,
    min_estimate = c(-0.20, -0.22, -0.21),
    max_estimate = c(0.00, -0.02, -0.01))
)
figure <- tempfile(fileext = ".pdf")
grDevices::pdf(figure, width = 7, height = 3.2)
tryCatch(plot_licorice_results(x), finally = grDevices::dev.off())
unlink(figure)
```

plot_power_design *Plot the Paper's Power-Design Application*

## Description

Plots the two-sided normal-approximation power curves and target-power sample sizes for Cases A and C in an object returned by paper_power_design().

## Usage

```
plot_power_design(
  x = paper_power_design(),
  target_power = 0.8,
 colors = c(unadjusted = "#333333", interacted = "#0072B2", efficient = "#009E73",
    pooled = "#D55E00"),
  ...
)
```

## Arguments

| | |
|---|---|
| x | An object returned by paper_power_design(). |
| target_power | Horizontal reference power. |
| colors | Named colors for the three curves in each panel. |
| ... | Additional graphical parameters passed to plot.default(). |

## Value

The input object, invisibly.

## Examples

```
design <- paper_power_design(n_grid = seq(50, 1200, by = 10))
figure <- tempfile(fileext = ".pdf")
grDevices::pdf(figure, width = 7, height = 3.4)
tryCatch(plot_power_design(design), finally = grDevices::dev.off())
unlink(figure)
```

plot_simulation_results *Plot the Paper's Simulation Results*

## Description

Plots archived or newly generated simulation summaries as either empirical variance relative to the semiparametric bound or confidence-interval coverage, together with the corresponding analytic benchmarks.

## Usage

```
plot_simulation_results(
  results = NULL,
  benchmarks = NULL,
  metric = c("efficiency", "coverage"),
  sample_size = 800L,
  ...
)
```

## Arguments

| | |
|---|---|
| results | A simulation summary data frame or an object returned by run_paper_simulations(). With NULL, the packaged complete-run summary is used. |
| benchmarks | Population benchmark data. With NULL, the packaged analytic benchmarks are used. |
| metric | Either "efficiency" or "coverage". |
| sample_size | Sample size displayed in the efficiency plot. |
| ... | Unused. |

## Value

The supplied results, invisibly.

## Examples

```
figure <- tempfile(fileext = ".pdf")
grDevices::pdf(figure, width = 7.1, height = 5.2)
tryCatch(
plot_simulation_results(
  paper_simulation_results,
  paper_population_benchmarks,
  metric = "efficiency"
), finally = grDevices::dev.off()
)
unlink(figure)
```

`prepare_licorice_data` *Prepare the Licorice-Gargle Trial Records*

## Description

Converts the source-format licorice-gargle data distributed by the optional medicaldata package into the analysis-ready structure used by the article. This function does not distribute participant records. Users must review the source data terms and obtain any permission required for their intended use.

## Usage

```
prepare_licorice_data(data = NULL)
```

## Arguments

| | |
|---|---|
| `data` | A source-format data frame. When NULL, the function retrieves licorice_gargle from the installed medicaldata package. |

## Value

The 25-column analysis-ready data frame used by `analyze_licorice()`.

## References

Ruetzler K, Fleck M, Nabecker S, et al. (2013). A randomized, double-blind comparison of licorice versus sugar-water gargle for prevention of postoperative sore throat and postextubation coughing. Anesthesia & Analgesia, 117(3), 614-621. doi:10.1213/ANE.0b013e318299a650

## Examples

```
# Synthetic records illustrate the required source format, not trial results.
set.seed(42)
n <- 235L
source_data <- data.frame(
  preOp_gender = sample(0:1, n, replace = TRUE),
  preOp_asa = sample(1:3, n, replace = TRUE),
  preOp_calcBMI = runif(n, 20, 35),
  preOp_age = sample(18:80, n, replace = TRUE),
  preOp_mallampati = sample(1:4, n, replace = TRUE),
  preOp_smoking = sample(1:3, n, replace = TRUE),
  preOp_pain = sample(0:1, n, replace = TRUE),
  treat = rep(0:1, c(117, 118)),
  intraOp_surgerySize = sample(1:3, n, replace = TRUE),
  extubation_cough = sample(0:3, n, replace = TRUE),
  pacu30min_cough = sample(0:3, n, replace = TRUE),
  pacu30min_throatPain = sample(0:10, n, replace = TRUE),
  pacu30min_swallowPain = sample(0:10, n, replace = TRUE),
  pacu90min_cough = sample(0:3, n, replace = TRUE),
  pacu90min_throatPain = sample(0:10, n, replace = TRUE),
```

```
  postOp4hour_cough = sample(0:3, n, replace = TRUE),
  postOp4hour_throatPain = sample(0:10, n, replace = TRUE),
  pod1am_cough = sample(0:3, n, replace = TRUE),
  pod1am_throatPain = sample(0:10, n, replace = TRUE)
)
source_data$pacu30min_throatPain[c(1, 118)] <- NA
d <- prepare_licorice_data(source_data)
dim(d)
```

---

`print.licorice_analysis` *Print a Licorice-Application Result*

---

## Description

Print a Licorice-Application Result

## Usage

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

## Arguments

| | |
|---|---|
| `x` | An object returned by `analyze_licorice()`. |
| `...` | Additional arguments passed to `print.data.frame()`. |

## Value

The input object, invisibly.

---

`print.rct_adjustment` *Print a Covariate-Adjustment Result*

---

## Description

Print a Covariate-Adjustment Result

## Usage

```
## S3 method for class 'rct_adjustment'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
```

### Arguments

| | |
|---|---|
| x | An object returned by rct_adjust() or rct_crossfit(). |
| digits | Number of significant digits to print. |
| ... | Additional arguments passed to print.data.frame(). |

### Value

The input object, invisibly.

---

## print.rct_power_design *Print a Power-Design Result*

---

### Description

Print a Power-Design Result

### Usage

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

### Arguments

| | |
|---|---|
| x | An object returned by paper_power_design(). |
| ... | Additional arguments passed to print.data.frame(). |

### Value

The input object, invisibly.

---

## print.rct_simulation *Print a Paper Simulation Result*

---

### Description

Print a Paper Simulation Result

### Usage

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

### Arguments

| | |
|---|---|
| x | An object returned by run_paper_simulations(). |
| ... | Additional arguments passed to print.data.frame(). |

### Value

The input object, invisibly.

---

## rct_adjust *Estimate a Marginal Mean Difference with Covariate Adjustment*

---

### Description

Estimates the treatment-minus-control marginal mean difference in a two-arm randomized controlled trial with a continuous outcome. The adjusted methods use the same empirical distribution of baseline covariates for both arms.

### Usage

```
rct_adjust(
  outcome,
  treatment,
  covariates = NULL,
  allocation = NULL,
  methods = c("unadjusted", "ancova", "interacted"),
  conf_level = 0.95,
  diagnostic_hc2 = FALSE
)
```

### Arguments

| | |
|---|---|
| outcome | Complete, finite numeric continuous outcome. |
| treatment | Complete numeric or logical treatment indicator, coded 1 for treatment and 0 for control. Both arms must be represented. |
| covariates | Baseline covariates. Supply NULL, a data frame, a numeric matrix, or a numeric vector with one row per outcome. Data-frame factors are expanded with model.matrix(); the resulting columns must be complete and finite. |
| allocation | Known probability of assignment to treatment. When NULL, the realized treatment fraction is used, and the influence calculation accounts for estimating that fraction. |
| methods | Any subset of "unadjusted", "ancova", and "interacted". |
| conf_level | Confidence level for normal-reference intervals. |
| diagnostic_hc2 | If TRUE, include the coefficient-only heteroskedasticity-consistent type 2 (HC2) diagnostic for interacted standardization. |

## Details

The common-slope method is ordinary least squares with a treatment indicator and centered baseline features. Its standard error is the heteroskedasticity-consistent type 0 coefficient sandwich. The interacted method fits a separate linear projection in each arm, standardizes both fits over all participants, and uses the joint influence function that includes empirical-standardization variation. Setting diagnostic_hc2 to TRUE adds a coefficient-only heteroskedasticity-consistent type 2 (HC2) calculation for the interacted point estimate. That calculation is provided as a diagnostic and is not generally valid for the superpopulation marginal estimand when treatment effects vary with covariates.

## Value

An object of class "rct_adjustment". Its estimates component contains point estimates, standard errors, and normal-reference confidence intervals. The influence component contains participant-level influence contributions.

## Examples

```
set.seed(1)
n <- 200
x <- rnorm(n)
a <- rbinom(n, 1, 0.5)
y <- 0.4 * a + x + rnorm(n)
fit <- rct_adjust(y, a, data.frame(x = x), allocation = 0.5)
fit
confint(fit)
```

---

# rct_adjustment_methods

*Methods for Covariate-Adjustment Results*

---

## Description

Standard methods extract the estimate table, coefficient vector, influence-function covariance matrix, or normal-reference confidence intervals from an object returned by `rct_adjust()` or `rct_crossfit()`.

## Usage

```
## S3 method for class 'rct_adjustment'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)

## S3 method for class 'rct_adjustment'
coef(object, ...)

## S3 method for class 'rct_adjustment'
vcov(object, ...)
```

```
## S3 method for class 'rct_adjustment'
confint(object, parm, level = 0.95, ...)

## S3 method for class 'rct_adjustment'
summary(object, ...)
```

## Arguments

| | |
|---|---|
| x, object | An object of class "rct_adjustment". |
| row.names, optional | Arguments passed to as.data.frame(). |
| ... | Additional arguments for the corresponding generic. |
| parm | Optional method names or numeric indices. |
| level | Confidence level. |

## Value

as.data.frame and summary return the estimate table; coef returns a named vector; vcov returns the covariance matrix estimated from the stored influence contributions; confint returns a two-column matrix.

---

# rct_crossfit *Cross-Fitted Covariate Adjustment*

---

## Description

Computes a cross-fitted augmented estimator of the treatment-minus-control marginal mean difference. Outcome regressions are trained separately within each arm and evaluated only on held-out participants. Both fitted means are averaged through the same augmented score.

## Usage

```
rct_crossfit(
  outcome,
  treatment,
  covariates,
  allocation = NULL,
  folds = 2L,
  fold_id = NULL,
  learner = c("linear", "quadratic"),
  seed = NULL,
  conf_level = 0.95
)
```

## Arguments

| | |
|---|---|
| outcome | Complete, finite numeric continuous outcome. |
| treatment | Complete numeric or logical treatment indicator, coded 1 for treatment and 0 for control. Both arms must be represented. |
| covariates | Baseline covariates. Supply NULL, a data frame, a numeric matrix, or a numeric vector with one row per outcome. Data-frame factors are expanded with `model.matrix()`; the resulting columns must be complete and finite. |
| allocation | Known probability of assignment to treatment. When NULL, the realized treatment fraction is used, and the influence calculation accounts for estimating that fraction. |
| folds | Number of folds. Ignored when fold_id is supplied. |
| fold_id | Optional integer or factor vector defining the fold of each participant. Every fold must be represented. Fold membership should be fixed without using outcomes. |
| learner | Either "linear", "quadratic", or a custom prediction function with the interface described in Details. |
| seed | Optional seed used only to randomly permute otherwise balanced fold labels. With NULL, folds follow participant order deterministically. |
| conf_level | Confidence level for normal-reference intervals. |

## Details

Built-in linear and quadratic learners use ordinary least squares. The quadratic learner adds a squared term for each model-matrix column; redundant columns are removed using baseline covariates alone. If a built-in fold-by-arm regression is rank deficient or has fewer observations than the number of coefficients plus two, it falls back to the corresponding training-arm mean.

A custom learner must be a function with arguments x_train, y_train, and x_new. It must return one finite numeric prediction for every row of x_new. Training and prediction are called once per arm within each fold.

## Value

An object of class "rct_adjustment".

## Examples

```
d <- simulate_rct_case("A", n = 120, seed = 11)
rct_crossfit(
  d$outcome, d$treatment, d["x"],
  allocation = 0.5, folds = 2, learner = "quadratic"
)
```

`rct_power` *Normal-Approximation Power for a Marginal Mean Difference*

## Description

Computes the rejection probability of a normal-reference test for a treatment-minus-control marginal mean difference. The `"greater"` and `"less"` alternatives correspond to positive and negative effects, respectively.

## Usage

```
rct_power(
  n,
  effect,
  variance,
  alpha = 0.05,
  alternative = c("two.sided", "greater", "less")
)
```

## Arguments

| | |
|---|---|
| `n` | Total sample size. |
| `effect` | Planning marginal mean difference. |
| `variance` | Asymptotic variance factor on the square-root-n scale. |
| `alpha` | Type I error probability. |
| `alternative` | Test direction: `"two.sided"`, `"greater"` for a positive effect, or `"less"` for a negative effect. |

## Value

A numeric vector of normal-approximation rejection probabilities.

## Examples

```
rct_power(n = 300, effect = 0.5, variance = c(12.5, 8.5, 4))
```

`rct_sample_size` *Required Sample Size for a Marginal Mean Difference*

## Description

Inverts the exact one- or two-tail normal power expression. With round_up equal to TRUE, the result is the smallest integer total sample size attaining the requested normal-approximation power.

## Usage

```
rct_sample_size(
  effect,
  variance,
  power = 0.8,
  alpha = 0.05,
  alternative = c("two.sided", "greater", "less"),
  round_up = TRUE
)
```

## Arguments

| | |
|---|---|
| `effect` | Planning marginal mean difference. |
| `variance` | Asymptotic variance factor on the square-root-n scale. |
| `power` | Target power. |
| `alpha` | Type I error probability. |
| `alternative` | Test direction: `"two.sided"`, `"greater"` for a positive effect, or `"less"` for a negative effect. |
| `round_up` | Return the smallest attaining integer if TRUE, or the continuous solution if FALSE. |

## Value

A numeric vector of total sample sizes.

## Examples

```
rct_sample_size(effect = 0.5, variance = c(12.5, 8.5, 4), power = 0.8)
```

`rct_variance_factors` *Analytic Variance Factors for the Paper's Gaussian Designs*

## Description

Computes the variance factors used in the article's one-covariate Gaussian examples. The data-generating law is

$$Y = \Delta A + \{b_0 + (b_1 - b_0)A\}X + c(X^2 - 1) + \epsilon,$$

where $X \sim N(0,1)$, $A \sim \text{Bernoulli}(\pi)$, and $\epsilon \sim N(0, \sigma^2)$ are mutually independent. The value of $\sigma^2$ is supplied through `noise_variance`; the marginal effect $\Delta$ does not enter the variance factors.

## Usage

```
rct_variance_factors(
  allocation,
  beta_control,
  beta_treatment,
  quadratic = 0,
  noise_variance = 1
)
```

## Arguments

| | |
|---|---|
| `allocation` | Probability of assignment to treatment. |
| `beta_control` | Linear slope in the control arm. |
| `beta_treatment` | Linear slope in the treatment arm. |
| `quadratic` | Coefficient of the centered quadratic term. |
| `noise_variance` | Error variance, common to both arms. |

## Value

A named numeric vector containing variance factors for the unadjusted estimator, the semiparametric efficiency bound, interacted linear adjustment, pooled analysis of covariance (ANCOVA), and the coefficient-only heteroskedasticity-consistent type 2 (HC2) variance limit. The optimal and pooled slopes are returned as well.

## Examples

```
rct_variance_factors(
  allocation = 0.5, beta_control = 1,
  beta_treatment = 1, quadratic = 0.75
)
```

## read_licorice_data *Read an Authorized Licorice-Gargle Data File*

### Description

Verifies the MD5 and SHA-256 fingerprints of the exact reviewed R data file before loading it into an isolated environment. The file itself is not included in RCTCovAdj.

### Usage

```
read_licorice_data(path)
```

### Arguments

| | |
|---|---|
| path | Path to the authorized licorice_gargle RDA file. |

### Value

The verified analysis-ready data frame.

### Examples

```
# An unverified file is rejected before it can be loaded.
path <- tempfile(fileext = ".rda")
writeLines("This is not the reviewed trial-data file.", path)
try(read_licorice_data(path))
unlink(path)

# Set this environment variable only for a copy you are authorized to use.
path <- Sys.getenv("RCTCOVADJ_LICORICE_RDA", unset = "")
if (nzchar(path)) {
  d <- read_licorice_data(path)
}
```

## reproduce_paper_simulations *Write a Reproducible Simulation Run*

### Description

Runs `run_paper_simulations()` and writes portable comma-separated summaries, optional replicate data, PDF and PNG figures, session information, and file checksums to an explicitly chosen directory.

## Usage

```
reproduce_paper_simulations(
  output_dir,
  reps = 4000L,
  sample_sizes = c(200L, 800L),
  cases = c("A", "B", "C", "D"),
  seed = 20260903L,
  keep_replicates = TRUE,
  progress = interactive(),
  figures = TRUE,
  overwrite = FALSE
)
```

## Arguments

| | |
|---|---|
| output_dir | Destination directory, supplied explicitly by the user. There is no default. Use a path under tempdir() for temporary output. |
| reps | Replications per case and sample-size cell. |
| sample_sizes | A subset of 200 and 800. |
| cases | A subset of "A", "B", "C", and "D". |
| seed | Master seed. |
| keep_replicates | Retain replicate-level estimates and standard errors. |
| progress | Print one message after each completed cell. |
| figures | Whether to draw efficiency and coverage figures in both PDF and PNG formats. |
| overwrite | Whether an existing package-defined output bundle may be replaced. Outputs from an earlier bundle that are not requested in the new run are removed only after the new bundle has been generated. |

## Value

A named character vector of written paths, invisibly.

## Examples

```
out <- tempfile("rctcovadj-")
reproduce_paper_simulations(
  out, reps = 3, sample_sizes = 200, cases = "A",
  figures = FALSE
)
unlink(out, recursive = TRUE)
```

`run_paper_simulations` *Run the Paper's Monte Carlo Experiment*

## Description

Reproduces the four continuous-outcome designs, seven estimators, independent random-number streams, and deterministic two-fold split used in the article. The default is the complete 32,000-trial experiment. Use a small value of reps for a quick workflow check.

## Usage

```
run_paper_simulations(
  reps = 4000L,
  sample_sizes = c(200L, 800L),
  cases = c("A", "B", "C", "D"),
  seed = 20260903L,
  keep_replicates = FALSE,
  progress = interactive()
)
```

## Arguments

| | |
|---|---|
| `reps` | Replications per case and sample-size cell. |
| `sample_sizes` | A subset of 200 and 800. |
| `cases` | A subset of "A", "B", "C", and "D". |
| `seed` | Master seed. |
| `keep_replicates` | Retain replicate-level estimates and standard errors. |
| `progress` | Print one message after each completed cell. |

## Value

An object of class "rct_simulation" with aggregate summary, population benchmarks, settings, and optionally replicate-level results.

## Examples

```
quick <- run_paper_simulations(
  reps = 4, sample_sizes = 200, cases = "A"
)
quick$summary
```

## simulate_rct_case *Simulate One Continuous-Outcome Randomized Controlled Trial*

### Description

Generates one trial under a named Gaussian data-generating law in simulation_cases. The marginal treatment mean difference is 0.5 in every case.

### Usage

```
simulate_rct_case(case = c("A", "B", "C", "D"), n = 200L, seed = NULL)
```

### Arguments

| | |
|---|---|
| case | One of "A", "B", "C", or "D". |
| n | Number of participants. |
| seed | Optional nonnegative integer seed. |

### Value

A data frame with columns x, treatment, and outcome. Design parameters are stored in the case_parameters attribute.

### Examples

```
d <- simulate_rct_case("A", n = 100, seed = 42)
head(d)
```

## simulation_cases *Simulation Cases Used in the Article*

### Description

Four one-covariate Gaussian data-generating laws used to study nonlinear prognostic structure, treatment-effect heterogeneity, unequal allocation, and opposing arm-specific slopes.

### Usage

```
simulation_cases
```

## Format

A data frame with 4 rows and 6 variables:

**case** Case identifier.

**label** Descriptive case label.

**pi** Probability of assignment to treatment.

**b0** Control-arm linear slope.

**b1** Treatment-arm linear slope.

**quad** Coefficient of the centered quadratic term.

## Source

Generated from the analytic designs specified in the accompanying manuscript.

# Index