

Package ‘rdborrow’

October 8, 2026

Type Package

Title External Control Borrowing for Rare Disease Trials

Description Implements causal inference methods for incorporating external control data into randomized controlled trials (RCTs) with longitudinal outcomes. Provides an analysis module supporting weighting-based methods such as inverse probability weighting (IPW) and augmented inverse probability weighting (AIPW), difference-in-differences (DID), and synthetic control approaches for borrowing external control information, as well as a simulation module for generating trial and external control data, evaluating estimator performance via Monte Carlo studies, and conducting power analyses for sample size determination. Methods are based on Zhou et al. (2025) <[doi:10.1093/jrsssa/qnae075](https://doi.org/10.1093/jrsssa/qnae075)> and Zhou et al. (2024) <[doi:10.1080/10543406.2024.2330209](https://doi.org/10.1080/10543406.2024.2330209)>.

Version 0.0.5.0

License Apache License (>= 2)

Depends R (>= 4.1.0)

Imports checkmate, futile.logger, mvtnorm, boot, Matrix, CVXR (>= 1.8.1), copula, progress, stats, utils, methods

Suggests covr, ECOSolveR, knitr, pkgdown, rmarkdown, lintr, spelling, styler, testthat (>= 3.3.0)

Config/testthat/edition 3

Collate 'method_class.R' 'analysis_class.R' 'method_SCM_class.R'
'method_DID_class.R' 'analysis_OLE_class.R'
'method_weighting_class.R' 'analysis_primary_class.R' 'data.R'
'ec_ipw.R' 'did_ec_ipw.R' 'did_ec_aipw.R' 'did_ec_or.R'
'ec_aipw.R' 'ec_weights.R' 'legacy.R' 'package.R'
'rdborrow-package.R' 'run_analysis.R' 'run_simulation.R'
'scm.R' 'simulate_X_copula.R' 'simulate_X_dct_mvnorm.R'
'simulate_X_mixture.R' 'simulate_outcome_from_model.R'
'simulate_trial.R' 'simulate_trial_status.R'
'simulate_trt_assign.R' 'simulation_class.R'
'simulation_report_class.R'

URL <https://genentech.github.io/rdborrow/>,
<https://github.com/Genentech/rdborrow>

BugReports <https://github.com/Genentech/rdborrow/issues>

VignetteBuilder knitr

LazyData true

Encoding UTF-8

Language en-US

Config/roxygen2/version 8.1.0

Config/Needs/website rmarkdown

NeedsCompilation no

Author Lei Shi [aut],
 Matt Secrest [cre, aut] (ORCID:
[<https://orcid.org/0000-0002-0939-4902>](https://orcid.org/0000-0002-0939-4902)),
 Herbert Pang [aut],
 Chen Chen [aut],
 Jiawen Zhu [aut],
 Genentech, Inc. [cph]

Maintainer Matt Secrest <secrmatt@gmail.com>

Repository CRAN

Date/Publication 2026-10-08 18:30:07 UTC

Contents

rdborrow-package	3
did_ec_aipw	4
did_ec_ipw	5
did_ec_or	6
ec_aipw	8
ec_ipw	10
estimate	11
run_analysis	13
run_simulation	15
scm	16
setup_analysis_OLE	17
setup_analysis_primary	19
setup_bootstrap	20
setup_method_DID	21
setup_method_SCM	22
setup_method_weighting	22
setup_simulation_OLE	23
setup_simulation_primary	24
simulate_outcome_from_model	26
simulate_trial	27

<i>rdborrow-package</i>	3
simulate_trial_status	28
simulate_trt_assign	29
simulate_X_copula	30
simulate_X_dct_mvnorm	31
simulate_X_mixture	32
SyntheticData	33
SyntheticDataII	34
Index	35

rdborrow-package	<i>The rdborrow package</i>
------------------	-----------------------------

Description

rdborrow is a package that implements causal inference methods for analyzing and designing clinical trials with external controls.

Authors

- The following authors contribute to the development and maintenance of the package:
- Lei Shi, University of California Berkeley, leishi1998@gmail.com
 - Herbert Pang, Genentech Inc.
 - Chen Chen, Genentech Inc.
 - Jiawen Zhu, Genentech Inc.

See Also

- Useful links:
- GitHub Repo for rdborrow: <https://github.com/Genentech/rdborrow>
 - Causal estimators for incorporating external controls in randomized trials with longitudinal outcomes (ec_ipw(), ec_aipw()): [doi:10.1093/jrsssa/qnae075](https://doi.org/10.1093/jrsssa/qnae075)
 - Estimating treatment effect in randomized trial after control to treatment crossover using external controls (did_ec_ipw(), did_ec_aipw(), did_ec_or(), scm()): [doi:10.1080/10543406.2024.2330209](https://doi.org/10.1080/10543406.2024.2330209)
 - Shi L, Pang H, Chen C, Zhu J. rdborrow: an R package for causal inference incorporating external controls in randomized controlled trials with longitudinal outcomes. J Biopharm Stat. 2025 Oct; 35(6):1043-1066. [doi:10.1080/10543406.2025.2489283](https://doi.org/10.1080/10543406.2025.2489283)

did_ec_aipw

*DID-EC-AIPW method***Description**

Creates a method object for difference-in-differences augmented inverse probability weighting (DID-EC-AIPW) estimation with external control borrowing for the open-label extension phase (Zhou et al., 2024, Eq. 5). Doubly robust: consistent if either the propensity score model or the outcome model is correct.

Usage

```
did_ec_aipw(
  ps_formula,
  trt_formula = NULL,
  outcome_formula,
  bootstrap = 500L,
  bootstrap_ci_type = NULL
)
```

Arguments

- | | |
|-----------------|---|
| ps_formula | Formula string for the propensity score model predicting trial participation. The right-hand side should use only columns in covariates_col_name. . or any other column is an error; an outcome gives a warning, because adjusting for an outcome measured after randomization can bias the treatment effect. |
| trt_formula | Formula string for the treatment assignment model, or NULL (default) for the marginal probability of treatment in the trial. The model is fit on trial patients only, and its predicted probabilities set the inverse-probability weights of trial treated and trial control patients. The left-hand side is replaced by the treatment column, so it can be any name. The right-hand side should use only columns in covariates_col_name. . or any other column is an error; an outcome gives a warning, because adjusting for an outcome measured after randomization can bias the treatment effect. |
| outcome_formula | Character vector of outcome model formulas, one per outcome. Each formula is matched to an outcome by its left-hand side, so the order does not matter. The left-hand side must be the outcome name itself; to model a transformed outcome, transform the column first. The right-hand side should use only columns in covariates_col_name. . or any other column is an error; an outcome gives a warning, because adjusting for an outcome measured after randomization can bias the treatment effect. |
| bootstrap | Number of bootstrap replicates (at least 2). Defaults to 500. Use about 1000 or more for reported intervals; small values are for quick checks only, and their interval can exclude the point estimate. |

bootstrap_ci_type
Bootstrap CI type: one of "perc" (default), "bca", "norm", or "basic". "bca" is slow when bootstrap is smaller than the number of patients: boot::boot.ci() then refits the estimator once for each patient.

Value

An S4 object of class did_ec_aipw_method.

References

Zhou et al. (2024). Estimating treatment effect in randomized trial after control to treatment crossover using external controls. *Journal of Biopharmaceutical Statistics*. doi:10.1080/10543406.2024.2330209

Examples

```
did_ec_aipw(  
  ps_formula = "S ~ x1 + x2 + x3 + x4 + x5",  
  trt_formula = "A ~ x1 + x2 + x3 + x4 + x5",  
  outcome_formula = c(  
    "y1 ~ x1 + x2 + x3 + x4 + x5",  
    "y2 ~ x1 + x2 + x3 + x4 + x5",  
    "y3 ~ x1 + x2 + x3 + x4 + x5",  
    "y4 ~ x1 + x2 + x3 + x4 + x5"  
  ),  
  bootstrap = 500  
)
```

did_ec_ipw	DID-EC-IPW method
------------	-------------------

Description

Creates a method object for difference-in-differences inverse probability weighting (DID-EC-IPW) estimation with external control borrowing for the open-label extension phase (Zhou et al., 2024, Eq. 4).

Usage

```
did_ec_ipw(  
  ps_formula,  
  trt_formula = NULL,  
  bootstrap = 500L,  
  bootstrap_ci_type = NULL  
)
```

Arguments

ps_formula	Formula string for the propensity score model predicting trial participation. The right-hand side should use only columns in covariates_col_name. . or any other column is an error; an outcome gives a warning, because adjusting for an outcome measured after randomization can bias the treatment effect.
trt_formula	Formula string for the treatment assignment model, or NULL (default) for the marginal probability of treatment in the trial. The model is fit on trial patients only, and its predicted probabilities set the inverse-probability weights of trial treated and trial control patients. The left-hand side is replaced by the treatment column, so it can be any name. The right-hand side should use only columns in covariates_col_name. . or any other column is an error; an outcome gives a warning, because adjusting for an outcome measured after randomization can bias the treatment effect.
bootstrap	Number of bootstrap replicates, at least 2 (required for DID methods). Defaults to 500. Use about 1000 or more for reported intervals; small values are for quick checks only, and their interval can exclude the point estimate.
bootstrap_ci_type	Bootstrap CI type: one of "perc" (default), "bca", "norm", or "basic". "bca" is slow when bootstrap is smaller than the number of patients: boot::boot.ci() then refits the estimator once for each patient.

Value

An S4 object of class did_ec_ipw_method.

References

Zhou et al. (2024). Estimating treatment effect in randomized trial after control to treatment crossover using external controls. *Journal of Biopharmaceutical Statistics*. doi:10.1080/10543406.2024.2330209

Examples

```
did_ec_ipw(  
  ps_formula = "S ~ x1 + x2 + x3 + x4 + x5",  
  trt_formula = "A ~ x1 + x2 + x3 + x4 + x5",  
  bootstrap = 500  
)
```

did_ec_or	DID-EC-OR method constructor
-----------	------------------------------

Description

Creates a method object for difference-in-differences outcome regression (DID-EC-OR) estimation with external control borrowing for the open-label extension phase (Zhou et al., 2024, Eq. 3). Uses outcome models only (no propensity score model).

Usage

```
did_ec_or(
  outcome_formula_ext,
  outcome_formula_rct_ctrl,
  outcome_formula_rct_trt,
  bootstrap = 500L,
  bootstrap_ci_type = NULL
)
```

Arguments

outcome_formula_ext	Character vector of outcome model formulas for external controls, one per outcome.
outcome_formula_rct_ctrl	Character vector of outcome model formulas for RCT control subjects, one per outcome.
outcome_formula_rct_trt	Character vector of outcome model formulas for RCT treated subjects, one per outcome. In all three arguments, each formula is matched to an outcome by its left-hand side, so the order does not matter. The left-hand side must be the outcome name itself; to model a transformed outcome, transform the column first. The right-hand side should use only columns in <code>covariates_col_name</code> . <code>.</code> or any other column is an error; an outcome gives a warning, because adjusting for an outcome measured after randomization can bias the treatment effect.
bootstrap	Number of bootstrap replicates (at least 2). Defaults to 500. Use about 1000 or more for reported intervals; small values are for quick checks only, and their interval can exclude the point estimate.
bootstrap_ci_type	Bootstrap CI type: one of "perc" (default), "bca", "norm", or "basic". "bca" is slow when bootstrap is smaller than the number of patients: <code>boot::boot.ci()</code> then refits the estimator once for each patient.

Value

An S4 object of class `did_ec_or_method`.

References

Zhou et al. (2024). Estimating treatment effect in randomized trial after control to treatment crossover using external controls. *Journal of Biopharmaceutical Statistics*. doi:10.1080/10543406.2024.2330209

Examples

```
model_forms <- c(
  "y1 ~ x1 + x2 + x3 + x4 + x5",
  "y2 ~ x1 + x2 + x3 + x4 + x5",
  "y3 ~ x1 + x2 + x3 + x4 + x5",
  "y4 ~ x1 + x2 + x3 + x4 + x5"
```

```

)
did_ec_or(
  outcome_formula_ext = model_forms,
  outcome_formula_rct_ctrl = model_forms,
  outcome_formula_rct_trt = model_forms,
  bootstrap = 500
)

```

ec_aipw

EC-AIPW method

Description

Creates a method object for augmented IPW estimation with external control borrowing (Zhou et al., 2025). Augments the IPW estimator with an outcome regression model for improved efficiency. Pass to [setup_analysis_primary](#) and [run_analysis](#).

Usage

```

ec_aipw(
  ps_formula,
  outcome_formula,
  weight = NULL,
  bootstrap = NULL,
  bootstrap_ci_type = NULL
)

```

Arguments

ps_formula	Formula string for the propensity score model predicting trial participation. The right-hand side should use only columns in covariates_col_name. . or any other column is an error; an outcome gives a warning, because adjusting for an outcome measured after randomization can bias the treatment effect.
outcome_formula	Character vector of outcome model formulas, one per outcome (e.g., c("y1 ~ x1 + x2", "y2 ~ x1 + x2")). Each formula is matched to an outcome by its left-hand side, so the order does not matter. The left-hand side must be the outcome name itself; to model a transformed outcome, transform the column first. The right-hand side should use only columns in covariates_col_name. . or any other column is an error; an outcome gives a warning, because adjusting for an outcome measured after randomization can bias the treatment effect.
weight	Borrowing weight. NULL (default) for data-adaptive optimal weight, 0 for no direct borrowing, or a value in (0, 1]. At 0 the external controls get no weight, but the outcome model is still fit on all controls, trial and external (as in Zhou et al., Theorem 2). The estimate stays valid by randomization alone; the external data affect only its precision. Unlike ec_ipw , it therefore does not use trial data only.

bootstrap Number of bootstrap replicates (at least 2), or NULL (default) for sandwich variance with normal CIs. Use about 1000 or more for reported intervals; small values are for quick checks only, and their interval can exclude the point estimate.

bootstrap_ci_type Bootstrap CI type, or NULL (default) which resolves to "perc" when bootstrap is set. One of "perc", "bca", "norm", or "basic". Needs bootstrap. "bca" is slow when bootstrap is smaller than the number of patients: `boot::boot.ci()` then refits the estimator once for each patient.

Value

An S4 object of class `ec_aipw_method`.

References

Zhou et al. (2025). Causal estimators for incorporating external controls in randomized trials with longitudinal outcomes. *JRSS-A*, 188(3), 791-818. [doi:10.1093/jrssa/qnae075](https://doi.org/10.1093/jrssa/qnae075)

Examples

```
# optimal weight, sandwich SE
ec_aipw(
  ps_formula = "S ~ x1 + x2 + x3 + x4 + x5",
  outcome_formula = c(
    "y1 ~ x1 + x2 + x3 + x4 + x5",
    "y2 ~ x1 + x2 + x3 + x4 + x5"
  )
)

# no direct borrowing
ec_aipw(
  ps_formula = "S ~ x1 + x2 + x3 + x4 + x5",
  outcome_formula = c(
    "y1 ~ x1 + x2 + x3 + x4 + x5",
    "y2 ~ x1 + x2 + x3 + x4 + x5"
  ),
  weight = 0
)

# fixed weight with bootstrap
ec_aipw(
  ps_formula = "S ~ x1 + x2 + x3 + x4 + x5",
  outcome_formula = c(
    "y1 ~ x1 + x2 + x3 + x4 + x5",
    "y2 ~ x1 + x2 + x3 + x4 + x5"
  ),
  weight = 0.3,
  bootstrap = 500
)
```

ec_ipw	<i>EC-IPW method constructor</i>
--------	----------------------------------

Description

Creates a method object for IPW estimation with external control borrowing (Zhou et al., 2025). Pass to `setup_analysis_primary` and `run_analysis`.

Usage

```
ec_ipw(ps_formula, weight = NULL, bootstrap = NULL, bootstrap_ci_type = NULL)
```

Arguments

ps_formula	Formula string for the propensity score model predicting trial participation. The left-hand side is replaced internally (e.g., "S ~ x1 + x2 + x3"). The right-hand side should use only columns in <code>covariates_col_name</code> . . or any other column is an error; an outcome gives a warning, because adjusting for an outcome measured after randomization can bias the treatment effect.
weight	Borrowing weight. NULL (default) for data-adaptive optimal weight, 0 for RCT-only, or a value in (0, 1].
bootstrap	Number of bootstrap replicates (at least 2), or NULL (default) for sandwich variance with normal CIs. Use about 1000 or more for reported intervals; small values are for quick checks only, and their interval can exclude the point estimate.
bootstrap_ci_type	Bootstrap CI type, or NULL (default) which resolves to "perc" when bootstrap is set. One of "perc", "bca", "norm", or "basic". Needs bootstrap. "bca" is slow when bootstrap is smaller than the number of patients: <code>boot::boot.ci()</code> then refits the estimator once for each patient.

Value

An S4 object of class `ec_ipw_method`.

References

Zhou et al. (2025). Causal estimators for incorporating external controls in randomized trials with longitudinal outcomes. *JRSS-A*, 188(3), 791-818. doi:10.1093/jrsssa/qnae075

Examples

```
# optimal weight, sandwich SE
ec_ipw(ps_formula = "S ~ x1 + x2 + x3 + x4 + x5")

# no borrowing
ec_ipw(ps_formula = "S ~ x1 + x2 + x3 + x4 + x5", weight = 0)
```

```

# optimal weight with bootstrap
ec_ipw(
  ps_formula = "S ~ x1 + x2 + x3 + x4 + x5",
  bootstrap = 500
)

# fixed weight with bootstrap
ec_ipw(
  ps_formula = "S ~ x1 + x2 + x3 + x4 + x5",
  weight = 0.3,
  bootstrap = 500
)

```

estimate

Run estimation for a method object

Description

S4 generic that dispatches to the appropriate estimation logic based on the method class. Each method subclass (e.g., `ec_ipw_method`, `did_ec_ipw_method`) implements its own `estimate()` method containing the full estimation pipeline.

Usage

```

estimate(method, ...)

## S4 method for signature 'ec_ipw_method'
estimate(
  method,
  data,
  outcomes,
  treatment,
  trial_status,
  covariates,
  alpha = 0.05,
  quiet = TRUE
)

## S4 method for signature 'did_ec_ipw_method'
estimate(
  method,
  data,
  outcomes,
  treatment,
  trial_status,
  covariates,
  alpha = 0.05,

```

```
    quiet = TRUE,
    T_cross
  )

## S4 method for signature 'did_ec_aipw_method'
estimate(
  method,
  data,
  outcomes,
  treatment,
  trial_status,
  covariates,
  alpha = 0.05,
  quiet = TRUE,
  T_cross
)

## S4 method for signature 'did_ec_or_method'
estimate(
  method,
  data,
  outcomes,
  treatment,
  trial_status,
  covariates,
  alpha = 0.05,
  quiet = TRUE,
  T_cross
)

## S4 method for signature 'ec_aipw_method'
estimate(
  method,
  data,
  outcomes,
  treatment,
  trial_status,
  covariates,
  alpha = 0.05,
  quiet = TRUE
)

## S4 method for signature 'scm_method'
estimate(
  method,
  data,
  outcomes,
  treatment,
```

```

    trial_status,
    covariates,
    alpha = 0.05,
    quiet = TRUE,
    T_cross
  )

```

Arguments

method	An S4 method object (e.g., from ec_ipw).
...	Additional method-specific arguments.
data	Data frame with all subjects (RCT + external controls).
outcomes	Character vector of outcome column names.
treatment	Name of the treatment column.
trial_status	Name of the trial participation column.
covariates	Character vector of covariate column names.
alpha	Significance level, more than 0 and less than 1 (default 0.05).
quiet	Logical. Suppress output (default TRUE).
T_cross	Integer crossover time point (OLE methods only).

Value

For primary methods, a list with results and borrow_weight; for OLE methods, a data frame. See [run_analysis](#) for the columns and row names.

Examples

```

method <- ec_ipw(ps_formula = "S ~ x1 + x2 + x3 + x4 + x5")
estimate(
  method,
  data = SyntheticData,
  outcomes = c("y1", "y2"),
  treatment = "A",
  trial_status = "S",
  covariates = c("x1", "x2", "x3", "x4", "x5")
)

```

run_analysis

Run an analysis with external control borrowing

Description

Estimates treatment effects by combining randomized trial data with external controls. Choose a method, wrap it in an analysis object, and pass it here.

Usage

```
run_analysis(analysis_obj, quiet = TRUE)
```

Arguments

`analysis_obj` An analysis object created by [setup_analysis_primary](#) or [setup_analysis_OLE](#).
`quiet` Logical. If TRUE, suppress printed output.

Details

Six borrowing methods are available:

[ec_ipw](#) Inverse probability weighting (primary analysis).
[ec_aipw](#) Augmented inverse probability weighting (primary analysis).
[did_ec_ipw](#) Difference-in-differences with IPW (open-label extension).
[did_ec_aipw](#) Difference-in-differences with AIPW (open-label extension).
[did_ec_or](#) Difference-in-differences with outcome regression (open-label extension).
[scm](#) Synthetic control method (open-label extension).

Value

For primary methods ([ec_ipw\(\)](#), [ec_aipw\(\)](#)), a list with:

`results` A data frame with one row for each outcome, named `tau1`, `tau2`, and so on, in the order of `outcome_col_name`. The columns are `point_estimates`, `standard_deviation`, and either `lower_CI_normal` and `upper_CI_normal` (sandwich variance, when `bootstrap` is NULL) or `lower_CI_boot` and `upper_CI_boot`. `standard_deviation` is the sandwich standard error, or the standard deviation of the bootstrap replicates.

`borrow_weight` The borrowing weight that was used.

For OLE methods ([did_ec_ipw\(\)](#), [did_ec_aipw\(\)](#), [did_ec_or\(\)](#), [scm\(\)](#)), a data frame with one row for each open-label visit and the columns `point_estimates`, `standard_deviation` (of the bootstrap replicates), `lower_CI_boot`, and `upper_CI_boot`. The rows are named `tau<k>` for `k` from `T_cross + 1` to the number of outcomes, where `k` is the position of the outcome in `outcome_col_name`. List the outcomes in visit order.

See Also

[run_simulation](#) for evaluating operating characteristics via Monte Carlo simulation.

Examples

```
method <- ec_ipw(ps_formula = "S ~ x1 + x2 + x3 + x4 + x5")
analysis <- setup_analysis_primary(
  data = SyntheticData,
  trial_status_col_name = "S",
  treatment_col_name = "A",
  outcome_col_name = c("y1", "y2"),
```

```

    covariates_col_name = c("x1", "x2", "x3", "x4", "x5"),
    method_weighting_obj = method
  )
run_analysis(analysis)

```

run_simulation

Evaluate operating characteristics via Monte Carlo simulation

Description

Runs repeated simulations under user-specified data-generating scenarios to estimate power, type I error rate, bias, and coverage for one or more borrowing methods.

Usage

```
run_simulation(simulation_obj, quiet = TRUE)
```

Arguments

`simulation_obj` A simulation object created by [setup_simulation_primary](#) or [setup_simulation_OLE](#).
`quiet` Logical. If TRUE, suppress iteration output.

Details

Six borrowing methods are available:

[ec_ipw](#) Inverse probability weighting (primary analysis).
[ec_aipw](#) Augmented inverse probability weighting (primary analysis).
[did_ec_ipw](#) Difference-in-differences with IPW (open-label extension).
[did_ec_aipw](#) Difference-in-differences with AIPW (open-label extension).
[did_ec_or](#) Difference-in-differences with outcome regression (open-label extension).
[scm](#) Synthetic control method (open-label extension).

Value

A simulation report object containing estimated power, type I error rate, and related operating characteristics for each method.

See Also

[run_analysis](#) for analyzing a single dataset.

Examples

```
method <- ec_ipw(ps_formula = "S ~ x1 + x2 + x3 + x4 + x5")
sim <- setup_simulation_primary(
  data_matrix_list_null = list(SyntheticData, SyntheticData),
  trial_status_col_name = "S",
  treatment_col_name = "A",
  outcome_col_name = c("y1", "y2"),
  covariates_col_name = c("x1", "x2", "x3", "x4", "x5"),
  method_obj_list = list(method),
  true_effect = 0,
  method_description = "IPW"
)
run_simulation(sim)
```

scm	<i>SCM method constructor</i>
-----	-------------------------------

Description

Creates a method object for synthetic control estimation with external control borrowing for the open-label extension phase (Zhou et al., 2024). Constructs a weighted combination of external controls matching each RCT control subject on covariates and pre-crossover outcomes. The penalty is chosen by leave-one-out cross-validation over the external controls, so the data needs at least 2 external controls. The covariates must be numeric or logical; code a factor as 0/1 indicators first.

Usage

```
scm(
  lambda_min = 0,
  lambda_max = 0.1,
  nlambdas = 2L,
  parallel = "no",
  ncpus = 1L,
  bootstrap = 200L,
  bootstrap_ci_type = NULL
)
```

Arguments

<code>lambda_min</code>	Minimum penalty parameter for LOOCV. A finite number of at least 0.
<code>lambda_max</code>	Maximum penalty parameter for LOOCV. A finite number of at least <code>lambda_min</code> .
<code>nlambdas</code>	Number of lambda values to evaluate in LOOCV, evenly spaced from <code>lambda_min</code> to <code>lambda_max</code> . With <code>nlambdas = 1</code> , <code>lambda_max</code> must equal <code>lambda_min</code> .
<code>parallel</code>	Parallelization type for bootstrap ("no", "multicore", or "snow").
<code>ncpus</code>	Number of CPUs for parallel bootstrap.

bootstrap Number of bootstrap replicates (at least 2). Defaults to 200. Use about 1000 or more for reported intervals; small values are for quick checks only, and their interval can exclude the point estimate.

bootstrap_ci_type Bootstrap CI type: one of "perc" (default), "bca", "norm", or "basic". "bca" is slow when bootstrap is smaller than the number of patients: `boot::boot.ci()` then refits the estimator once for each patient.

Details

The analysis result from `run_analysis()` has an attribute "lambda", the penalty that cross-validation selected: `attr(result, "lambda")`.

Value

An S4 object of class `scm_method`.

References

Zhou et al. (2024). Estimating treatment effect in randomized trial after control to treatment crossover using external controls. *Journal of Biopharmaceutical Statistics*. doi:10.1080/10543406.2024.2330209

Examples

```
scm(lambda_min = 0, lambda_max = 0.001, nlambda = 2, bootstrap = 50)
```

setup_analysis_OLE	<i>Set up an open-label extension (OLE) analysis</i>
--------------------	--

Description

Bundles data, column mappings, crossover time, and a method object into an analysis object ready to be passed to `run_analysis`.

Usage

```
setup_analysis_OLE(
  data,
  trial_status_col_name,
  treatment_col_name,
  outcome_col_name,
  covariates_col_name,
  method_OLE_obj,
  T_cross,
  alpha = 0.05
)
```

Arguments

<code>data</code>	A data frame containing all subject-level data. It must have trial treated patients, trial controls, and external controls.
<code>trial_status_col_name</code>	Name of the trial status column: 1 for trial patients, 0 for external controls. Must be numeric or logical.
<code>treatment_col_name</code>	Name of the treatment column: 1 for treated, 0 for control. Must be numeric or logical, not a factor. External controls must have 0.
<code>outcome_col_name</code>	Character vector of outcome column names covering both placebo-controlled and OLE periods, in visit order. The position of each outcome sets its period and its result row name. The columns must have no missing values.
<code>covariates_col_name</code>	Character vector of covariate column names. The columns must have no missing values. Outcome and covariate columns cannot be named S or A, or be the trial-status or treatment column.
<code>method_OLE_obj</code>	A method object created by did_ec_ipw , did_ec_aipw , did_ec_or , or scm .
<code>T_cross</code>	Integer crossover time point (column index boundary). The first <code>T_cross</code> outcome columns are from the placebo-controlled phase and are used as negative controls for bias correction. The remaining <code>length(outcome_col_name) - T_cross</code> columns are from the open-label extension phase and are used to estimate the treatment effect. Must be a positive integer strictly less than <code>length(outcome_col_name)</code> .
<code>alpha</code>	Significance level, more than 0 and less than 1 (default 0.05).

Details

Available OLE methods:

[did_ec_ipw](#) Difference-in-differences with IPW.

[did_ec_aipw](#) DID with augmented IPW.

[did_ec_or](#) DID with outcome regression.

[scm](#) Synthetic control method.

Value

An object of class `analysis_OLE_obj`, to be passed to [run_analysis](#).

See Also

[run_analysis](#), [setup_analysis_primary](#)

Examples

```

method <- did_ec_ipw(
  ps_formula = "S ~ x1 + x2 + x3 + x4 + x5",
  trt_formula = "A ~ x1 + x2 + x3 + x4 + x5",
  bootstrap = 50
)
setup_analysis_OLE(
  data = SyntheticData,
  trial_status_col_name = "S",
  treatment_col_name = "A",
  outcome_col_name = c("y1", "y2", "y3", "y4"),
  covariates_col_name = c("x1", "x2", "x3", "x4", "x5"),
  method_OLE_obj = method,
  T_cross = 2
)

```

setup_analysis_primary

Set up a primary analysis with external control borrowing

Description

Bundles data, column mappings, and a method object into an analysis object ready to be passed to [run_analysis](#).

Usage

```

setup_analysis_primary(
  data,
  trial_status_col_name,
  treatment_col_name,
  outcome_col_name,
  covariates_col_name,
  method_weighting_obj,
  alpha = 0.05
)

```

Arguments

data	A data frame containing all subject-level data. It must have trial treated patients and trial controls. It must also have external controls, unless the method is <code>ec_ipw(weight = 0)</code> .
trial_status_col_name	Name of the trial status column: 1 for trial patients, 0 for external controls. Must be numeric or logical.
treatment_col_name	Name of the treatment column: 1 for treated, 0 for control. Must be numeric or logical, not a factor. External controls must have 0.

outcome_col_name	Character vector of outcome column names. The columns must have no missing values.
covariates_col_name	Character vector of covariate column names. The columns must have no missing values. Outcome and covariate columns cannot be named S or A, or be the trial-status or treatment column.
method_weighting_obj	A method object created by ec_ipw or ec_aipw .
alpha	Significance level, more than 0 and less than 1 (default 0.05).

Details

Available primary methods:

[ec_ipw](#) Inverse probability weighting.

[ec_aipw](#) Augmented inverse probability weighting (doubly robust).

Value

An object of class `analysis_primary_obj`, to be passed to [run_analysis](#).

See Also

[run_analysis](#), [setup_analysis_OLE](#)

Examples

```
method <- ec_ipw(ps_formula = "S ~ x1 + x2 + x3 + x4 + x5")
setup_analysis_primary(
  data = SyntheticData,
  trial_status_col_name = "S",
  treatment_col_name = "A",
  outcome_col_name = c("y1", "y2"),
  covariates_col_name = c("x1", "x2", "x3", "x4", "x5"),
  method_weighting_obj = method
)
```

setup_bootstrap	<i>Setup bootstrap (Deprecated)</i>
-----------------	-------------------------------------

Description

‘r lifecycle::badge("deprecated")’

Bootstrap settings are now specified directly in each method constructor (e.g., `[ec_ipw()]`, `[did_ec_ipw()]`). The separate bootstrap object is no longer needed.

Usage

```
setup_bootstrap(...)
```

Arguments

```
... Ignored.
```

Value

This function is defunct and always signals an error; it does not return a value.

Examples

```
try(setup_bootstrap())
```

setup_method_DID	<i>Setup method DID (Deprecated)</i>
------------------	--------------------------------------

Description

```
‘r lifecycle::badge("deprecated")‘
```

This function has been removed. Use [did_ec_ipw()] for inverse probability weighting, [did_ec_aipw()] for augmented inverse probability weighting, or [did_ec_or()] for outcome regression.

Usage

```
setup_method_DID(...)
```

Arguments

```
... Ignored.
```

Value

This function is defunct and always signals an error; it does not return a value.

Examples

```
try(setup_method_DID())
```

setup_method_SCM	<i>Setup method SCM (Deprecated)</i>
------------------	--------------------------------------

Description

```
‘r lifecycle::badge("deprecated")‘
```

This function has been removed. Use [scm()] instead.

Usage

```
setup_method_SCM(...)
```

Arguments

... Ignored.

Value

This function is defunct and always signals an error; it does not return a value.

Examples

```
try(setup_method_SCM())
```

setup_method_weighting	<i>Setup method weighting (Deprecated)</i>
------------------------	--

Description

```
‘r lifecycle::badge("deprecated")‘
```

This function has been removed. Use [ec_ipw()] for inverse probability weighting or [ec_aipw()] for augmented inverse probability weighting.

Usage

```
setup_method_weighting(...)
```

Arguments

... Ignored.

Value

This function is defunct and always signals an error; it does not return a value.

Examples

```
try(setup_method_weighting())
```

```
setup_simulation_OLE    Construct a simulation object for OLE analysis
```

Description

Construct a simulation object for OLE analysis

Usage

```
setup_simulation_OLE(
  data_matrix_list,
  trial_status_col_name,
  treatment_col_name,
  outcome_col_name,
  covariates_col_name,
  method_obj_list,
  T_cross,
  true_effect,
  method_description,
  alpha = 0.05
)
```

Arguments

data_matrix_list	List of simulated data frames.
trial_status_col_name	Name of the trial status column.
treatment_col_name	Name of the treatment column.
outcome_col_name	Character vector of outcome column names.
covariates_col_name	Character vector of covariate column names.
method_obj_list	List of method objects to evaluate.
T_cross	Crossover time point: a whole number of at least 1 and less than the number of outcomes.
true_effect	The true treatment effect at the final visit, a single number. [run_simulation()] evaluates estimates at the final visit only.
method_description	Character vector of method labels, one per method in 'method_obj_list'.
alpha	Significance level, more than 0 and less than 1.

Value

An object of class ‘simulation_OLE_obj’.

Examples

```
setup_simulation_OLE(
  data_matrix_list = list(SyntheticData),
  trial_status_col_name = "S",
  treatment_col_name = "A",
  outcome_col_name = c("y1", "y2", "y3", "y4"),
  covariates_col_name = c("x1", "x2", "x3", "x4", "x5"),
  method_obj_list = list(
    did_ec_ipw(
      ps_formula = "S ~ x1 + x2 + x3 + x4 + x5",
      trt_formula = "A ~ x1 + x2 + x3 + x4 + x5",
      bootstrap = 50
    )
  ),
  T_cross = 2,
  true_effect = 0,
  method_description = "IPW, DID"
)
```

```
setup_simulation_primary
```

Construct a simulation object for primary analysis

Description

Construct a simulation object for primary analysis

Usage

```
setup_simulation_primary(
  data_matrix_list_null,
  trial_status_col_name,
  treatment_col_name,
  outcome_col_name,
  covariates_col_name,
  method_obj_list,
  true_effect,
  method_description,
  data_matrix_list_alt = list(),
  alt_effect = numeric(0),
  alpha = 0.05
)
```

Arguments

<code>data_matrix_list_null</code>	List of data frames simulated under the null.
<code>trial_status_col_name</code>	Name of the trial status column.
<code>treatment_col_name</code>	Name of the treatment column.
<code>outcome_col_name</code>	Character vector of outcome column names.
<code>covariates_col_name</code>	Character vector of covariate column names.
<code>method_obj_list</code>	List of method objects to evaluate.
<code>true_effect</code>	The true treatment effect at the final visit, a single number. <code>[run_simulation()]</code> evaluates estimates at the final visit only.
<code>method_description</code>	Character vector of method labels, one per method in <code>'method_obj_list'</code> .
<code>data_matrix_list_alt</code>	List of data frames simulated under the alternative.
<code>alt_effect</code>	Numeric vector of alternative treatment effects.
<code>alpha</code>	Significance level, more than 0 and less than 1.

Value

An object of class `'simulation_primary_obj'`.

Examples

```
setup_simulation_primary(
  data_matrix_list_null = list(SyntheticData),
  trial_status_col_name = "S",
  treatment_col_name = "A",
  outcome_col_name = c("y1", "y2"),
  covariates_col_name = c("x1", "x2", "x3", "x4", "x5"),
  method_obj_list = list(
    ec_ipw(ps_formula = "S ~ x1 + x2 + x3 + x4 + x5")
  ),
  true_effect = 0,
  method_description = "IPW"
)
```

simulate_outcome_from_model

Simulate outcomes from additive linear models

Description

Generates longitudinal outcomes from a structural causal model of the form $Y_t = A * \text{effect} + X$ phase ($t \leq T_{\text{cross}}$), treatment assignment 'A' is used directly. In the OLE phase ($t > T_{\text{cross}}$), all RCT patients are assumed to receive treatment (effect multiplied by 1 instead of 'A').

Usage

```
simulate_outcome_from_model(X, A, outcome_model_specs, OLE_flag, T_cross)
```

Arguments

X	Data frame of covariates.
A	Numeric vector of treatment indicators (same length as <code>nrow(X)</code>).
outcome_model_specs	List of lists, one per time point. See Details above.
OLE_flag	Logical. If 'TRUE', time points after <code>T_cross</code> use the OLE model (all patients treated).
T_cross	Positive integer. The crossover time point separating the primary and OLE phases. Only used when <code>OLE_flag = TRUE</code> .

Details

Each element of `'outcome_model_specs'` is a list with:

- 'effect'** Numeric scalar. Treatment effect for this time point.
- 'model_form_x'** Named numeric vector of covariate coefficients. Names must include `"1"` (intercept) and match column names in `'X'`.
- 'noise_mean'** Numeric scalar. Mean of the normal noise.
- 'noise_sd'** Numeric scalar. SD of the normal noise.

Value

A data frame with `'n'` rows and one column per time point (`'y1'`, `'y2'`, ...).

Examples

```
X <- data.frame(x1 = rnorm(20), x2 = rnorm(20))
A <- rbinom(20, 1, 0.5)
specs <- list(
  list(
    effect = 1.5,
```

```

      model_form_x = c("1" = 2.0, "x1" = 0.5, "x2" = -0.3),
      noise_mean = 0, noise_sd = 1
    ),
    list(
      effect = 0,
      model_form_x = c("1" = 1.0, "x1" = 0.2, "x2" = 0.1),
      noise_mean = 0, noise_sd = 1
    )
  )
Y <- simulate_outcome_from_model(X, A, specs, OLE_flag = FALSE, T_cross = 2)

```

simulate_trial

*Simulate trials***Description**

Simulate trials

Usage

```

simulate_trial(
  X_int,
  X_ext,
  num_treated,
  OLE_flag,
  T_cross,
  outcome_model_specs
)

```

Arguments

<code>X_int</code>	Data frame of internal (RCT) covariates.
<code>X_ext</code>	Data frame of external control covariates.
<code>num_treated</code>	Number of treated subjects.
<code>OLE_flag</code>	Logical. Whether this is an OLE simulation.
<code>T_cross</code>	Integer crossover time point.
<code>outcome_model_specs</code>	List of outcome model specifications.

Value

a data frame for the simulated data

Examples

```

X_int <- data.frame(x1 = rnorm(20), x2 = rnorm(20))
X_ext <- data.frame(x1 = rnorm(30), x2 = rnorm(30))
specs <- list(
  list(
    effect = 1.5,
    model_form_x = c("1" = 2.0, "x1" = 0.5, "x2" = -0.3),
    noise_mean = 0, noise_sd = 1
  ),
  list(
    effect = 0,
    model_form_x = c("1" = 1.0, "x1" = 0.2, "x2" = 0.1),
    noise_mean = 0, noise_sd = 1
  )
)
Data <- simulate_trial(X_int,
  X_ext,
  num_treated = 10,
  OLE_flag = FALSE,
  T_cross = 2,
  outcome_model_specs = specs
)

```

simulate_trial_status *Simulate trial participation status*

Description

Simulates a binary trial participation indicator using a logistic model. The probability of participation is `inv.logit(X_intercept covariate matrix with an intercept column prepended)`.

Usage

```
simulate_trial_status(X, model_specs)
```

Arguments

<code>X</code>	Data frame of covariates. The number of columns must equal <code>'length(model_specs\$coef) - 1'</code> (the intercept is added automatically).
<code>model_specs</code>	List with: 'family' Character string. Currently only <code>"binomial"</code> is supported. 'coef' Numeric vector of length <code>'ncol(X) + 1'</code>. Logistic regression coefficients (intercept first).

Value

A single-column data frame with column `'S'` (1 = RCT participant, 0 = external control).

Examples

```
X <- data.frame(x1 = rnorm(20), x2 = rnorm(20))
S <- simulate_trial_status(X, model_specs = list(
  family = "binomial",
  coef = c(0, 0.5, -0.5)
))
```

simulate_trt_assign	<i>Simulate treatment assignment</i>
---------------------	--------------------------------------

Description

Randomly assigns treatment to RCT patients ('S = 1') with a given probability. External control patients ('S = 0') always receive control ('A = 0').

Usage

```
simulate_trt_assign(X, S, prob)
```

Arguments

X	Data frame of covariates. Must have the same number of rows as 'S'.
S	Data frame with a column 'S' indicating trial status (1 = RCT, 0 = external control).
prob	Numeric scalar between 0 and 1. Probability of treatment assignment for RCT patients.

Value

A single-column data frame with column 'A' indicating treatment status (1 = treated, 0 = control).

Examples

```
X <- SyntheticData[c("x1", "x2")]
S <- SyntheticData["S"]
A <- simulate_trt_assign(X, S, prob = 1 / 2)
```

simulate_X_copula	<i>Simulate covariates using a copula</i>
-------------------	---

Description

Couples several marginal distributions using a copula to generate correlated multivariate covariates.

Usage

```
simulate_X_copula(n, p, cp, margins, paramMargins)
```

Arguments

n	Positive integer. Number of units to simulate.
p	Positive integer. Number of covariates. Must equal the dimension of ‘cp’.
cp	A copula object (from the ‘copula’ package).
margins	Character vector of length ‘p’. Names of marginal distributions (e.g. “norm”, “binom”).
paramMargins	List of length ‘p’. Each element is a named list of parameters for the corresponding marginal distribution.

Value

A data frame with ‘n’ rows and ‘p’ columns named ‘x1’, ..., ‘xp’.

Examples

```
normal <- copula::normalCopula(param = c(0.8), dim = 4, dispstr = "ar1")
X <- simulate_X_copula(1000, 4, normal,
  margins = c("norm", "t", "norm", "binom"),
  paramMargins = list(
    list(mean = 2, sd = 3),
    list(df = 2),
    list(mean = 0, sd = 1),
    list(size = 10, prob = 0.5)
  )
cor(X, method = "spearman")
```

`simulate_X_dct_mvnorm` *Simulate covariates by discretizing a multivariate normal*

Description

Draws from a multivariate normal distribution and optionally discretizes selected columns into categorical variables based on specified probability cutpoints.

Usage

```
simulate_X_dct_mvnorm(
  n,
  p,
  mu = rep(0, p),
  sig = diag(p),
  cat_cols = c(),
  cat_prob = list()
)
```

Arguments

<code>n</code>	Positive integer. Number of units to simulate.
<code>p</code>	Positive integer. Number of covariates.
<code>mu</code>	Numeric vector of length ‘p’. Mean of the multivariate normal. Defaults to a zero vector.
<code>sig</code>	Numeric matrix of dimension ‘p x p’. Covariance matrix. Defaults to the identity matrix.
<code>cat_cols</code>	Integer vector. Column indices to discretize into categorical variables. Each index must be between 1 and ‘p’.
<code>cat_prob</code>	List of numeric vectors, one per element of ‘cat_cols’. Each vector gives the category probabilities (must sum to 1). The number of categories equals the length of the vector, and the resulting values are 0-indexed (0, 1, ..., K-1).

Value

A data frame with ‘n’ rows and ‘p’ columns named ‘x1’, ..., ‘xp’.

Examples

```
simulate_X_dct_mvnorm(
  20, 3,
  mu = rep(0, 3), sig = diag(3),
  cat_cols = c(1),
  cat_prob = list(c(0.3, 0.7))
)
simulate_X_dct_mvnorm(
```

```

20, 3,
mu = rep(0, 3), sig = diag(3),
cat_cols = c(1, 3),
cat_prob = list(c(0.2, 0.6, 0.2), c(0.3, 0.7))
)

```

simulate_X_mixture	<i>Simulate covariates from a Gaussian mixture model</i>
--------------------	--

Description

Generates covariates where categorical variables define mixture components and continuous variables are drawn from component-specific multivariate normal distributions. Each combination of categorical levels has its own probability and its own distribution for the continuous covariates.

Usage

```

simulate_X_mixture(
  n,
  p_cat,
  p_cont,
  cat_level_list,
  cat_comb_prob,
  cont_para_list
)

```

Arguments

<code>n</code>	Positive integer. Number of units to simulate.
<code>p_cat</code>	Non-negative integer. Number of categorical covariates.
<code>p_cont</code>	Non-negative integer. Number of continuous covariates. At least one of ‘ <code>p_cat</code> ’ or ‘ <code>p_cont</code> ’ must be positive.
<code>cat_level_list</code>	List of length ‘ <code>p_cat</code> ’. Each element is a vector of possible levels for that categorical variable. The total number of combinations is ‘ <code>prod(lengths(cat_level_list))</code> ’.
<code>cat_comb_prob</code>	Numeric vector of probabilities, one per combination of categorical levels (in the order produced by <code>[expand.grid()]</code>). Must sum to 1.
<code>cont_para_list</code>	List of parameter lists for the continuous covariates. When ‘ <code>p_cat > 0</code> ’, must have one element per combination of categorical levels; each element is a list with ‘ <code>mean</code> ’ (length ‘ <code>p_cont</code> ’) and ‘ <code>sigma</code> ’ (‘ <code>p_cont</code> x <code>p_cont</code> ’ matrix). When ‘ <code>p_cat == 0</code> ’, a single-element list.

Value

A data frame with ‘`n`’ rows and ‘`p_cat + p_cont`’ columns named ‘`x1`’, ..., ‘`xp`’.

Examples

```
# Continuous only
X <- simulate_X_mixture(
  n = 100, p_cat = 0, p_cont = 2,
  cat_level_list = list(),
  cat_comb_prob = c(),
  cont_para_list = list(list(mean = c(0, 0), sigma = diag(2)))
)

# Mixed categorical and continuous
X <- simulate_X_mixture(
  n = 100, p_cat = 1, p_cont = 2,
  cat_level_list = list(c(0, 1)),
  cat_comb_prob = c(0.4, 0.6),
  cont_para_list = list(
    list(mean = c(0, 0), sigma = diag(2)),
    list(mean = c(2, 2), sigma = diag(2))
  )
)
```

SyntheticData

Synthetic example dataset for rdborrow

Description

A simulated dataset containing covariates, treatment, trial status, and longitudinal outcomes for demonstrating external control borrowing methods.

Usage

```
data(SyntheticData)
```

Format

A data frame with 300 rows and 12 columns: 100 trial patients on treatment ($S = 1$, $A = 1$), 100 trial controls ($S = 1$, $A = 0$), and 100 external controls ($S = 0$, $A = 0$). The covariates mimic a spinal muscular atrophy (SMA) study.

x1 SMA type: 0 = type II, 1 = type III.

x2 SMN2 copy number: 0 = 3 copies, 1 = 4 copies.

x3 Scoliosis: 0 = no, 1 = yes.

x4 Age at enrollment, in years.

x5 Baseline outcome.

A Treatment: 1 = treated, 0 = control.

S Trial status: 1 = trial patient, 0 = external control.

T_cross Last visit before trial controls cross over to treatment: 2 for all rows.

y1, y2, y3, y4 Outcomes at visits 1 to 4.

Source

Simulated with `simulate_trial()`. The script is `data-raw/SyntheticData.R` in the package source on GitHub.

`SyntheticDataII`*Unbalanced synthetic example dataset for rdborrow*

Description

A companion to `SyntheticData` with unequal group sizes, for tests and examples where balanced data would hide an error. With equal arms, for example, exchanging the randomization probability and its complement gives the same numbers.

Usage

```
data(SyntheticDataII)
```

Format

A data frame with 380 rows and the same 12 columns as `SyntheticData`: 160 trial patients on treatment ($S = 1, A = 1$), 80 trial controls ($S = 1, A = 0$), and 140 external controls ($S = 0, A = 0$). The trial randomizes 2:1, so the probability of treatment is $2/3$. SMA type III ($x_1 = 1$) is more common in the trial than in the external controls. The participation model has good overlap. The outcome models are the same as in `SyntheticData`, and `T_cross` is 2 for all rows.

Source

Simulated with `simulate_trial()`. The script is `data-raw/SyntheticDataII.R` in the package source on GitHub.

Index

- * **datasets**
 - SyntheticData, [33](#)
 - SyntheticDataII, [34](#)
- * **rdborrow**
 - rdborrow-package, [3](#)
- did_ec_aipw, [4](#), [14](#), [15](#), [18](#)
- did_ec_ipw, [5](#), [14](#), [15](#), [18](#)
- did_ec_or, [6](#), [14](#), [15](#), [18](#)
- ec_aipw, [8](#), [14](#), [15](#), [20](#)
- ec_ipw, [8](#), [10](#), [13–15](#), [20](#)
- estimate, [11](#)
- estimate, did_ec_aipw_method-method (estimate), [11](#)
- estimate, did_ec_ipw_method-method (estimate), [11](#)
- estimate, did_ec_or_method-method (estimate), [11](#)
- estimate, ec_aipw_method-method (estimate), [11](#)
- estimate, ec_ipw_method-method (estimate), [11](#)
- estimate, scm_method-method (estimate), [11](#)
- rdborrow (rdborrow-package), [3](#)
- rdborrow-package, [3](#)
- run_analysis, [8](#), [10](#), [13](#), [13](#), [15](#), [17–20](#)
- run_simulation, [14](#), [15](#)
- scm, [14](#), [15](#), [16](#), [18](#)
- setup_analysis_OLE, [14](#), [17](#), [20](#)
- setup_analysis_primary, [8](#), [10](#), [14](#), [18](#), [19](#)
- setup_bootstrap, [20](#)
- setup_method_DID, [21](#)
- setup_method_SCM, [22](#)
- setup_method_weighting, [22](#)
- setup_simulation_OLE, [15](#), [23](#)
- setup_simulation_primary, [15](#), [24](#)
- simulate_outcome_from_model, [26](#)
- simulate_trial, [27](#), [34](#)
- simulate_trial_status, [28](#)
- simulate_trt_assign, [29](#)
- simulate_X_copula, [30](#)
- simulate_X_dct_mvnorm, [31](#)
- simulate_X_mixture, [32](#)
- SyntheticData, [33](#), [34](#)
- SyntheticDataII, [34](#)