Package: recoverlite (via r-universe)

July 5, 2026

Title Pre-Data Recovery Tests for Planned Study Designs

Version 0.2.0

Description A prototype implementation of the pre-data recovery test: a standardized simulation protocol for evaluating whether a planned design-analysis pair can recover its target estimand before data collection. Researchers declare the estimand, data-generating assumptions, data strategy, missing-data process, and analysis strategy; the package simulates a crossed scenario grid (null and target effects, each under declared and pessimistically perturbed nuisance assumptions), applies the planned analysis to each simulated dataset, and reports target-null rejection, power, target bias with its exact decomposition into estimator bias and estimand drift, coverage, Type S and Type M errors, precision, and classified model failure, each with Monte Carlo uncertainty and explicit inclusion rules. A versioned threshold profile converts the diagnosands into a PASS/RISK/FAIL verdict - a decision convention, not a validity classification. Built on the 'DeclareDesign' framework.

License Apache License (>= 2)

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Depends R (>= 4.1)

Imports DeclareDesign (>= 1.0.0), fabricatr, randomizr, stats, utils

Suggests lme4, lmerTest, pbkrtest, simr, estimatr, testthat (>= 3.0.0), knitr, rmarkdown

VignetteBuilder knitr

Config/testthat/edition 3

URL https://github.com/heidihelena/recoverlite

BugReports https://github.com/heidihelena/recoverlite/issues

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

Date/Publication 2026-07-04 21:37:56 UTC

RemoteUrl https://github.com/heidihelena/recoverlite

RemoteRef HEAD

RemoteSha cf27d33b3dde2eef9177c39f5e808dcb9a2707f3

Contents

`attrition_model` *Declare the attrition (missing-outcome) model*

Description

Dropout follows a declared response model rather than an unstated one. Under `mechanism = "differential"`, the probability of dropout is

$$logit Pr(dropout|Z, B) = alpha_Z + gamma_Z B$$

depending on assignment and the *observed baseline* — not on the unobserved post-test outcome. Because dropout depends only on observed quantities (Z, B), missingness is MAR given the baseline; this keeps the drift mechanism transparent and makes baseline-based repairs (e.g. `planned_analysis()` with `estimator = "mi_baseline_adjusted"`) identifiable for the ITT estimand. The arm-specific intercepts alpha_Z are calibrated numerically so that each arm's *marginal* attrition rate equals its declared value: arm imbalance in retention is a declared quantity, not a by-product.

Usage

```
attrition_model(
  rate,
  mechanism = c("differential", "mcar"),
  rate_control = NULL,
  rate_treated = NULL,
  baseline_slope_treated = -0.5,
  baseline_slope_control = 0,
  max_rate = 0.6,
  evidence = NULL
)
```

Arguments

`rate`	Numeric in [0, 1). Anticipated overall marginal attrition rate; used for both arms unless arm-specific rates are given.
`mechanism`	Character. `"differential"` (default) or `"mcar"`.
`rate_control`, `rate_treated`	Numeric or `NULL`. Arm-specific marginal attrition rates; default to `rate`.
`baseline_slope_treated`	Numeric. gamma_1, the log-odds change in treated-arm dropout per baseline SD (default -0.5: poorer baseline prognosis, higher dropout). Ignored under `"mcar"`.
`baseline_slope_control`	Numeric. gamma_0 (default 0). Ignored under `"mcar"`.
`max_rate`	Numeric. Field-relevant cap applied when the pessimistic scenario multiplies the declared rates by 1.5; must be stated and justified.
`evidence`	Character or `NULL`. Evidence tier / source for the declared attrition values (e.g. observed rates in comparable trials). Echoed in the report.

Details

With a negative treated-arm slope and higher scores indicating better functioning, intervention participants with poorer baseline prognosis are more likely to drop out. This mechanism persists when the treatment effect is zero, so under null scenarios the analyzable-data contrast is displaced from 0 and rejections are false claims about the target.

`mechanism = "mcar"` drops observations completely at random at the declared rate(s).

Value

An object of class `recovery_attrition`.

`cluster_trial` *Declare a cluster-randomized trial data strategy*

Description

Cluster-level random assignment with individual outcomes and cluster sizes fixed by design. For pupil i in cluster j,

$$Y_ij = tauZ_j + u_j + e_ij$$

with total individual-level variance standardized to 1 and ICC = var(u) / (var(u) + var(e)). The declared effect is expressed in individual-level (total) standard-deviation units. The simulated data contain a `cluster` id column for use in the analysis formula, e.g. `y_observed ~ treatment + (1 | cluster)`.

Usage

```
cluster_trial(
  n_clusters,
  n_per_cluster,
  icc,
  icc_pessimistic = NULL,
  allocation = 0.5,
  evidence = NULL
)
```

Arguments

`n_clusters`	Integer. Total number of clusters randomized.
`n_per_cluster`	Integer. Individuals measured per cluster.
`icc`	Numeric in [0, 1). Declared intraclass correlation.
`icc_pessimistic`	Numeric or `NULL`. The upper end of the empirically documented ICC range for the outcome domain, used in the pessimistic scenario rows. If `NULL`, the pessimistic rows default to `icc + 0.10` and the report records the value as a package default (the lowest tier of the evidence hierarchy). Declare it explicitly.
`allocation`	Numeric in (0, 1). Share of clusters assigned to treatment.
`evidence`	Character or `NULL`. Evidence tier / source for the declared and pessimistic ICC values (e.g. a published ICC range for the outcome domain). Echoed in the report.

Value

An object of class `recovery_data_strategy`.

declare_recovery *Declare a recovery design*

Description

Step 2 of the protocol: assemble the target estimand, the model and data strategy, and the answer strategy into a single executable declaration. Every major design feature is either declared here or recorded as omitted in the report.

Usage

```
declare_recovery(
  target,
  data_strategy,
  measurement = NULL,
  missingness = NULL,
  answer_strategy,
  effect = NULL
)
```

Arguments

target	A target_estimand() object.
data_strategy	A two_arm_trial() or cluster_trial() object.
measurement	A measured_outcome() object, or NULL to treat the outcome as perfectly reliable (recorded as an omission).
missingness	An attrition_model() object, or NULL to model no attrition (recorded as an omission).
answer_strategy	A planned_analysis() object.
effect	Numeric or NULL. The declared (expected) effect on the target scale. Defaults to the SESOI. Target scenario rows are always simulated at the SESOI; if the declared effect exceeds the SESOI, an informational expected-effect scenario is added outside the verdict (a study justified by an optimistic expected effect but unable to recover the smallest meaningful effect should not receive a clean PASS). Pessimistic rows never shrink the effect; fragility to the effect size itself is a separate curve, effect_fragility().

Value

An object of class recovery_design.

Examples

```
design <- declare_recovery(
  target = target_estimand(
    estimand = "ITT mean difference at 12 weeks",
    scale = "latent-outcome standardized mean difference",
    sesoi = 0.40
  ),
  data_strategy = two_arm_trial(n_per_arm = 100),
  measurement = measured_outcome(reliability = 0.70),
  missingness = attrition_model(rate = 0.15, mechanism = "differential"),
  answer_strategy = planned_analysis(
    estimator = "linear_model",
    formula = y_observed ~ treatment
  )
)
```

---

effect_fragility *Effect-size fragility curve*

Description

Shows how power, Type M exaggeration, and precision behave as the true effect shrinks below the declared target, with all nuisance assumptions held at their declared values. This is deliberately **outside the PASS/RISK/FAIL verdict**: shrinking the target effect changes the inferential target, so pessimistic scenario rows perturb nuisance assumptions only, and fragility to the effect size itself is reported as a separate continuous curve.

Usage

```
effect_fragility(design, effects = NULL, sims = 500, seed = NULL)
```

Arguments

design	A declare_recovery() object.
effects	Numeric vector of true effects to evaluate. Defaults to an even grid from a quarter of the declared effect up to the declared effect, with the SESOI included.
sims	Simulations per effect value (default 500; increase for smoother curves).
seed	Optional integer seed.

Value

A data frame of class recovery_fragility with one row per effect: power, Type M, and precision, each with its MCSE.

measured_outcome *Declare the measurement model for the observed outcome*

Description

Classical additive measurement error, per the manuscript (section 3.1): the observed outcome is `y_observed = y_true + e` with `Var(e) = 1/reliability - 1`, so that reliability = Var(y_true)/Var(y_observed). Under this model the raw treatment contrast is *not attenuated in expectation*: measurement error inflates residual variance (costing precision, power, and — downstream of low power — Type M exaggeration), and is charged to the variance account, not the bias account. (Attenuation by sqrt(reliability) would apply only if the estimator standardized by the observed-outcome SD.)

Usage

```
measured_outcome(reliability, evidence = NULL)
```

Arguments

`reliability`	Numeric in (0, 1]. Declared reliability (e.g. test-retest) of the outcome measure.
`evidence`	Character or `NULL`. Evidence tier / source for the declared reliability (e.g. a validation study). Echoed in the report.

Details

If measurement is omitted from `declare_recovery()`, the outcome is treated as perfectly reliable and the report states this explicitly (silence must not imply ideality).

Value

An object of class `recovery_measurement`.

nuisance_fragility *Nuisance-parameter fragility curve*

Description

Fragility curves over a nuisance parameter the diagnosis identifies as binding (manuscript, section 2.2, Step 3): the verdict uses the pre-specified point scenarios; these curves show where the verdict would change. The chosen diagnosands are power, target-null-relevant quantities being outside scope here, target bias, coverage, and estimand drift across the grid, at the target effect (the SESOI).

Usage

```
nuisance_fragility(
  design,
  parameter = c("attrition_rate", "reliability", "icc"),
  values,
  sims = 500,
  seed = NULL
)
```

Arguments

design	A declare_recovery() object.
parameter	One of "attrition_rate", "reliability", "icc".
values	Numeric grid of parameter values to evaluate.
sims	Simulations per grid point (default 500).
seed	Optional integer seed.

Value

A data frame of class recovery_fragility.

---

planned_analysis *Declare the planned analysis (answer strategy)*

Description

The analysis that will actually be applied to the observed data, stated exactly — including the specific inference method, since finite-sample behavior can differ sharply across methods that share a model formula (manuscript, section 3.2).

Usage

```
planned_analysis(
  estimator = c("linear_model", "lmm_random_intercept", "cluster_mean_ttest",
    "mi_baseline_adjusted"),
  formula,
  alpha = 0.05,
  inference = c("satterthwaite", "kenward_roger", "wald_z"),
  m_imputations = 20,
  degenerate_counts = FALSE
)
```

Arguments

estimator	Character; one of the estimators above.
formula	The planned model formula. The simulated data provide y_observed, treatment, baseline, and (for cluster designs) cluster. The treatment coefficient is the one whose name starts with "treatment".
alpha	Numeric. Two-sided significance level of the planned decision rule (default 0.05); also fixes the nominal 1 - alpha confidence interval whose coverage is diagnosed.
inference	Character. Inference method for "lmm_random_intercept"; ignored otherwise.
m_imputations	Integer. Number of imputations for "mi_baseline_adjusted" (default 20).
degenerate_counts	Logical. Whether degenerate fits (singular or boundary variance estimates; failure class (c) of the manuscript's taxonomy) count against the model-failure threshold. Design-specific and must be pre-specified; the choice is echoed in the report. Fatal errors and nonconvergence always count; diagnostic warnings are reported but never count. Default FALSE (degenerate fits are recorded, retained, and their marginal effect on coverage reported separately).

Details

Available estimators:

- "linear_model" — stats::lm() on the retained sample (complete-case), Wald t interval. Baseline adjustment is expressed through the formula, e.g. y_observed ~ treatment + baseline.
- "lmm_random_intercept" — linear mixed model via lmerTest::lmer() on the retained sample; inference selects "satterthwaite" (default), "kenward_roger" (requires pbkrtest), or "wald_z" (large-sample z interval).
- "cluster_mean_ttest" — two-sample t-test on equal-weighted cluster means (with cluster sizes fixed by design, equal and pupil-count weighting coincide), n_clusters - 2 degrees of freedom.
- "mi_baseline_adjusted" — multiple imputation of missing outcomes from a normal model drawing on (treatment, baseline), followed by the baseline-adjusted linear model on each completed dataset and Rubin's rules with Barnard-Rubin degrees of freedom. Identifiable for the ITT estimand when dropout is MAR given the observed baseline; per the manuscript (section 2.3), evaluated through bias against the target, not through drift.

Value

An object of class recovery_analysis.

`recovery_test` *Run the recovery test*

Description

Simulates the declared design over the crossed scenario grid (manuscript, section 2.2, Step 3): null and target effects, each under declared and pessimistically perturbed nuisance assumptions, plus an informational expected-effect row when the declared effect exceeds the SESOI. The planned analysis is applied to every simulated dataset and the diagnosands of section 2.3 are computed with Monte Carlo standard errors and explicit inclusion rules.

Usage

```
recovery_test(
  design,
  sims = 2000,
  scenarios = "confirmatory_grid",
  thresholds = recovery_thresholds(),
  seed = NULL
)
```

Arguments

`design`	A `declare_recovery()` object.
`sims`	Integer. Simulations per scenario row. The default 2000 is an initial working number, not a standard: the relevant stopping rule is whether the MCSE is small enough to support the verdict, and verdicts near a threshold may require many more.
`scenarios`	`"confirmatory_grid"` (default: the four verdict rows Null-declared, Null-pessimistic, Target-declared, Target-pessimistic) or `"target_grid"` (the two target rows only, for estimation-focused designs evaluated with the estimation profile).
`thresholds`	A `recovery_thresholds()` object. The profile and any deviations from it are echoed in the report.
`seed`	Optional integer seed for reproducibility.

Value

An object of class `recovery_result`. Use `verdict()` and `report()` on it.

`recovery_test_stable` *Run the recovery test under the algorithmic doubling stopping rule*

Description

Wraps `recovery_test()` with the pre-declared simulation stopping rule of the protocol (manuscript, section 2.4). The number of simulations `S` starts at `start_sims` and is doubled whenever a required threshold margin is still within `mcse_margin` (default 2) Monte Carlo standard errors of its threshold, up to a pre-declared ceiling `max_sims`. The rule stops as soon as the verdict is determined: a stable declared failure locks FAIL, or all required margins resolve (clean PASS or determined RISK). Anything still within `mcse_margin` MCSEs of a threshold at `max_sims` is reported as RISK: the run cannot resolve whether the design clears the bright line, and that irresolution is itself the finding.

Usage

```
recovery_test_stable(
  design,
  start_sims = 2000L,
  max_sims = 16000L,
  scenarios = "confirmatory_grid",
  thresholds = recovery_thresholds(),
  seed = NULL,
  verbose = FALSE
)
```

Arguments

`design`	A `declare_recovery()` object.
`start_sims`	Initial simulations per scenario row (default 2000).
`max_sims`	Pre-declared maximum simulations per scenario row (default 16000, i.e. up to three doublings from 2000).
`scenarios`	`"confirmatory_grid"` (default) or `"target_grid"`; passed through to `recovery_test()`.
`thresholds`	A `recovery_thresholds()` object.
`seed`	Optional integer seed for reproducibility.
`verbose`	Logical; if `TRUE`, report each doubling to the console.

Details

Doubling reuses the same `seed`; each `(seed, S)` run is fully reproducible.

Value

A `recovery_result` (as from `recovery_test()`) carrying an additional `"stopping"` attribute: a list with `start_sims`, `final_sims`, `max_sims`, `resolved` (whether the verdict was determined before the ceiling), `hit_ceiling`, and `unresolved` (a data frame of any required margins still within the MCSE band at the final `S`).

`recovery_thresholds` *Versioned verdict threshold profiles*

Description

The recovery-test protocol ships threshold *profiles* matched to design goals (manuscript, section 2.5). Thresholds are conventions: their function is not to be correct but to prevent post hoc negotiation with borderline results. A profile must be selected or justified before simulation; every deviation from a shipped profile is recorded and echoed in the report, and the report recomputes the verdict under the shipped strict and lenient profiles with signed margins to every threshold.

Usage

```
recovery_thresholds(
  profile = c("default", "strict", "lenient", "estimation"),
  null_rejection_mult = NULL,
  power = NULL,
  target_bias = NULL,
  coverage = NULL,
  type_s = NULL,
  type_m = NULL,
  model_failure = NULL,
  drift = NULL,
  overcoverage_flag = 0.975,
  mcse_margin = 2,
  min_conditional_n = 200,
  max_width = NULL
)
```

Arguments

`profile`	One of `"default"`, `"strict"`, `"lenient"`, `"estimation"`.
`null_rejection_mult`	Maximum target-null rejection rate as a multiple of the nominal alpha.
`power`	Minimum power at the SESOI.
`target_bias`	Maximum absolute target bias in units of the declared substantive scale Delta.
`coverage`	Minimum coverage of a nominal 95% interval (lower bound; one-sided).
`type_s`	Maximum Type S rate among significant estimates, when stably estimable.
`type_m`	Maximum Type M exaggeration among significant estimates, when stably estimable.
`model_failure`	Maximum counted model-failure rate (see `planned_analysis()` for which classes count).
`drift`	Maximum absolute estimand drift in Delta units; thresholded by the estimation profile only (reported, not thresholded, in confirmatory profiles, where its effect is captured by target bias and target-null rejection).

`overcoverage_flag`
: Coverage above this value is flagged as inefficiency (never failure).

`mcse_margin`
: All required threshold margins must exceed this many Monte Carlo standard errors for a PASS (default 2).

`min_conditional_n`
: Minimum contributing simulations for a conditional diagnosand (Type S, Type M) to be treated as stable (default 200 — a stability convention chosen so that the MCSE of a conditional proportion cannot exceed ~0.035).

`max_width`
: Maximum acceptable confidence-interval width for the estimation profile's precision criterion; usually inherited from `target_estimand()` by `recovery_test()`.

Details

The shipped confirmatory profiles are, exactly:

Criterion	Lenient	Default	Strict
Target-null rejection	<= 1.50 a	<= 1.25 a	<= 1.10 a
Power	>= 0.70	>= 0.80	>= 0.90
Target bias	<= 0.10 D	<= 0.05 D	<= 0.025 D
Coverage	>= 0.900	>= 0.925	>= 0.940
Type S	<= 0.05	<= 0.01	<= 0.005
Type M	<= 2.00	<= 1.50	<= 1.25
Model failure	<= 0.05	<= 0.01	<= 0.005

The rejection-rate threshold is one-sided: excess false claims trigger failure, while conservative behavior is flagged through power and precision. The coverage threshold is a lower bound; overcoverage above `overcoverage_flag` (default 0.975) is flagged as inefficiency and evaluated through precision, not treated as failure.

The `"estimation"` profile replaces the rejection-rate and Type S/M criteria with target bias, coverage, precision against a declared maximum acceptable width (see `target_estimand()`), and estimand drift, evaluated on the target rows only.

Value

An object of class `recovery_thresholds`, carrying the threshold-set version, the profile name, and a record of any values changed from the shipped profile.

`report` *Produce the standalone recovery report*

Description

Renders the full recovery report (protocol Step 6): target estimand and SESOI with scales, the declaration (including explicit statements of what was *not* modeled), the scenario grid with the null-world specification and the evidence tier of every pessimistic value, all diagnosands under every scenario with MCSEs, contributing counts, failure-class counts, and the bias decomposition, the threshold profile with signed margins, the verdict under the selected and the shipped strict/lenient profiles, the binding failure mode, and the computation settings. The report is standalone: a collaborator, reviewer, supervisor, or funder should be able to read it without running the code. A FAIL report is as complete as a PASS report.

Usage

```
report(result, file = NULL)
```

Arguments

`result`	A `recovery_result` from `recovery_test()`.
`file`	Optional path; if supplied, the report is also written there as plain text.

Value

The report, invisibly, as a character vector of lines.

`target_estimand` *Declare the target estimand*

Description

Step 1 of the recovery-test protocol: state, in one sentence, the quantity the study is designed to estimate, the scale on which it is expressed, and the smallest effect size of interest (SESOI). The SESOI should be justified on substantive grounds, not reverse-engineered from the available sample size.

Usage

```
target_estimand(
  estimand,
  scale,
  sesoi,
  bias_scale_unit = NULL,
  max_width = NULL
)
```

Arguments

estimand	Character. One-sentence statement of the target estimand (quantity, population, scale).
scale	Character. The scale on which the estimand is expressed, e.g. "latent-outcome standardized mean difference".
sesoi	Numeric. The smallest effect size of interest, on the estimand scale. Must be strictly positive. Target scenario rows are simulated at the SESOI.
bias_scale_unit	Numeric or NULL. The declared substantive unit Delta used to scale target bias, estimator bias, and estimand drift: each is (expectation - target) / bias_scale_unit. Defaults to the SESOI.
max_width	Numeric or NULL. Maximum acceptable confidence interval width, if one is declared. Used as the precision criterion by the estimation threshold profile; reported (not thresholded) otherwise.

Value

An object of class recovery_target.

See Also

declare_recovery()

Examples

```
target_estimand(
  estimand = "ITT mean difference at 12 weeks",
  scale = "latent-outcome standardized mean difference",
  sesoi = 0.40
)
```

---

two_arm_trial *Declare a two-arm randomized trial data strategy*

Description

Individual random assignment to two arms, with an observed baseline measure. The data-generating model follows the manuscript (section 3.1): for participant i with assignment Z,

$$B_i \; N(0, 1)$$

$$Y*_i = tauZ_i + rhoB_i + sqrt(1 - rho^2)eps_i$$

so the true (latent) outcome has unit SD given Z and the declared effect is expressed in latent standard-deviation units. The baseline is observed at randomization for every participant and is available to the answer strategy as the column baseline.

Usage

```
two_arm_trial(
  n_per_arm,
  allocation = 0.5,
  baseline_outcome_cor = 0.5,
  noncompliance = 0
)
```

Arguments

n_per_arm	Integer. Participants recruited per arm.
allocation	Numeric in (0, 1). Share assigned to treatment.
baseline_outcome_cor	Numeric in [0, 1). Correlation rho between the observed baseline and the true outcome (default 0.5).
noncompliance	Numeric in [0, 1). Share of treated participants who do not receive treatment (one-sided never-takers). The target estimand remains the declared ITT effect.

Details

n_per_arm is the number of participants *recruited* per arm (under equal allocation); attrition, if declared via attrition_model(), is applied after recruitment.

Value

An object of class recovery_data_strategy.

---

verdict *Apply the PASS/RISK/FAIL verdict rule*

Description

Applies the fixed verdict rule (manuscript, section 2.2, Step 5) mechanically to a recovery_test() result. Once thresholds and scenarios are fixed, the computation is mechanical — the judgment lives in the declaration and the scenario grid, not here.

Usage

```
verdict(result)
```

Arguments

result	A recovery_result from recovery_test().

Details

- **PASS** — all required thresholds met under all scenario rows the selected profile requires, with every margin exceeding `mcse_margin` (default 2) Monte Carlo standard errors.
- **RISK** — thresholds met under the declared-nuisance rows but not under a pessimistic row; or any margin within `mcse_margin` MCSEs of a threshold; or any required conditional diagnosand too unstable to evaluate; or required scenario rows missing from the run.
- **FAIL** — one or more required thresholds fail under a declared-nuisance row — including an inflated target-null rejection rate under Null-declared. The planned study cannot recover the target even if the declared assumptions are correct.

The verdict is recomputed under the shipped strict and lenient profiles (section 2.5): agreement across profiles carries more decision weight than any one profile, and a verdict that flips is itself a finding — the RISK category exists to hold it. The verdict is a decision convention, not a validity classification, and the full `report()` always travels with it.

Value

An object of class `recovery_verdict`.

Index