

# Package: bacipair (via r-universe)

October 16, 2026

**Type** Package

**Title** BACI / Paired-DiD / Synthetic-Control Estimators for Carbon-MRV Audits

**Version** 0.1.0

**Description** A reproducibility-by-construction toolkit for Before-After-Control-Impact (BACI) audits of carbon-MRV registers. Provides a staggered-treatment simulator with a known oracle ATT, a paired difference-in-differences estimator with proponent-cluster-robust standard errors, a time-varying synthetic-control comparator with simplex-constrained donor weights (Wang and Carreira-Perpinan, 2013), a proponent-level cluster bootstrap, a Goodman-Bacon (2021) staggered-DiD decomposition wrapper, and 'targets'-skeleton helpers for end-to-end pipeline reproducibility. Designed against the Macintosh et al. (2024) Australian Human-Induced Regeneration setting but applicable to any paired BACI register.

**License** MIT + file LICENSE

**URL** <https://github.com/max578/bacipair>

**BugReports** <https://github.com/max578/bacipair/issues>

**Encoding** UTF-8

**Language** en-AU

**LazyData** false

**Depends** R (>= 4.1)

**Imports** data.table (>= 1.15.0), stats, utils

**Suggests** bacondecomp, jsonlite, knitr, rmarkdown, targets, testthat (>= 3.0.0)

**VignetteBuilder** knitr

**Config/testthat/edition** 3

**RoxygenNote** 7.3.3

**Roxygen** list(markdown = TRUE)

**Repository** <https://max578.r-universe.dev>  
**Date/Publication** 2026-10-16 01:58:31 UTC  
**RemoteUrl** <https://github.com/max578/bacipair>  
**RemoteRef** main  
**RemoteSha** 5120f814be0a56769e4b422c3b36f8f216741eed

Contents

|                                       |           |
|---------------------------------------|-----------|
| baci_paired_did . . . . .             | 2         |
| bacipair_targets_plan . . . . .       | 3         |
| goodman_bacon_decomp . . . . .        | 5         |
| is_bacipair_abstention . . . . .      | 6         |
| proponent_cluster_bootstrap . . . . . | 7         |
| simulate_staggered_panel . . . . .    | 8         |
| synth_control_baci . . . . .          | 10        |
| write_oracle_recovery_log . . . . .   | 12        |
| <b>Index</b>                          | <b>13</b> |

---

|                 |                                                         |
|-----------------|---------------------------------------------------------|
| baci_paired_did | <i>Paired difference-in-differences on a BACI panel</i> |
|-----------------|---------------------------------------------------------|

---

Description

Estimates the average treatment effect on the treated (ATT) on a paired BACI panel via a two-way fixed-effects regression of the outcome on (a) a treated-arm-by-post-treatment interaction, plus (b) pair fixed effects via within-pair-and-arm demeaning, and (c) year fixed effects.

Usage

```
baci_paired_did(  
  panel,  
  outcome = "woody_cover",  
  treated_label = "credited",  
  comparison_label = "comparison",  
  cluster_col = "proponent_id"  
)
```

Arguments

|                                 |                                                                                                                                                        |
|---------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|
| panel                           | A data.table containing one row per (pair, year, arm) observation. Must contain pair_id, arm, year, treatment_year, and the outcome / cluster columns. |
| outcome                         | Character. Outcome column name (default "woody_cover").                                                                                                |
| treated_label, comparison_label | Character scalars. Levels of the arm column denoting treated vs comparison units (defaults "credited" / "comparison").                                 |

`cluster_col` Character. Cluster identifier (default "proponent\_id"); must be present in the panel.

### Details

Standard errors are cluster-robust at the proponent level (default) or at the pair level if `cluster_col = "pair_id"`. The estimator is the canonical CR1 sandwich form used in econometric panel work (Cameron & Miller, 2015), and is computed without external dependencies.

Operationally, this is the *naïve* paired-DiD comparator the translation memo (RDT-8) uses as the foil for the synthetic-control estimator: when elevation / rainfall climatology covary with treatment timing, paired-DiD has residual bias that the SC estimator nets out.

### Value

List with components:

`att_hat` Numeric. Estimated ATT (paired-DiD coefficient).

`se_cluster` Numeric. Cluster-robust (CR1) standard error.

`t_stat` Numeric. Robust t-statistic.

`ci_lo, ci_hi` Numeric. 95% normal-approximation CI.

`n_obs, n_clusters, n_pairs, n_years` Integers. Sample counts.

`coefficients` Numeric named vector of the estimated coefficients on the *demeaned* design: the ATT interaction (`did_i_dm`) plus one coefficient per non-base year dummy (`yr_<year>_dm`). The pair-and-arm intercept and the treated-arm and post-treatment main effects are absorbed by the within-pair-and-arm demeaning at fit time and are not part of the returned vector.

`call` Captured match.`call()`.

### Examples

```
sim <- simulate_staggered_panel(seed = 20260520L, n_pairs = 30L,
                               n_proponents = 8L)
did <- baci_paired_did(sim$panel)
did$att_hat
did$se_cluster
```

---

`bacipair_targets_plan` *Build a targets-skeleton plan for a bacipair audit*

---

### Description

Returns a list of targets: : `tar_target` calls covering the canonical pipeline:

**Usage**

```

bacipair_targets_plan(
  simulate_args = list(seed = 20260520L, n_pairs = 200L),
  synth_args = list(),
  did_args = list(),
  boot_args = list(n_boot = 200L, seed = 20260521L),
  log_path = "logs/oracle-recovery.json"
)

```

**Arguments**

|                            |                                                                                                                                  |
|----------------------------|----------------------------------------------------------------------------------------------------------------------------------|
| <code>simulate_args</code> | A named list passed to <code>simulate_staggered_panel()</code> (default <code>list(seed = 20260520L, n_pairs = 200L)</code> ).   |
| <code>synth_args</code>    | A named list passed to <code>synth_control_baci()</code> (default empty).                                                        |
| <code>did_args</code>      | A named list passed to <code>baci_paired_did()</code> (default empty).                                                           |
| <code>boot_args</code>     | A named list passed to <code>proponent_cluster_bootstrap()</code> (default <code>list(n_boot = 200L, seed = 20260521L)</code> ). |
| <code>log_path</code>      | Character. Path for the JSON oracle-recovery log (default <code>"logs/oracle-recovery.json"</code> ).                            |

**Details**

1. simulate the staggered-treatment panel (or load a real BACI panel),
2. fit the synthetic-control estimator,
3. fit the paired-DiD estimator,
4. proponent-cluster bootstrap of the SC point estimate,
5. Goodman-Bacon decomposition diagnostic,
6. write a JSON oracle-recovery log.

Skeleton means: this is a list ready to drop into `targets::tar_plan()` (or `targets::tar_pipeline()`). The function does *not* execute the pipeline. To run it, paste the returned list into a `_targets.R` script and call `targets::tar_make()`.

The targets package is in Suggests, so this function raises an informative error if targets is not installed.

**Value**

A named list of `targets::tar_target()` calls, one per pipeline node. Names are `panel`, `synth_control_fit`, `paired_did_fit`, `cluster_boot`, `goodman_bacon`, `oracle_recovery_log`.

**Examples**

```

if (requireNamespace("targets", quietly = TRUE)) {
  plan <- bacipair_targets_plan()
  names(plan)
}

```

---

goodman\_bacon\_decomp    *Goodman-Bacon (2021) staggered-DiD decomposition*


---

## Description

Decomposes a two-way fixed-effects DiD estimator into a weighted average of all 2x2 DiD estimators implied by the treatment timing, as in Goodman-Bacon (2021). Wraps `bacondecomp::bacon()`; that package is declared in `Suggests`, so the wrapper raises an informative error if it is not installed.

## Usage

```
goodman_bacon_decomp(
  panel,
  outcome = "woody_cover",
  id = "pair_id",
  time = "year",
  treated_label = "credited",
  treatment_indicator = "post_treatment"
)
```

## Arguments

|                                  |                                                                                                                                   |
|----------------------------------|-----------------------------------------------------------------------------------------------------------------------------------|
| <code>panel</code>               | A <code>data.table</code> containing the BACI panel (typically from <code>simulate_staggered_panel()</code> or a real-data port). |
| <code>outcome</code>             | Character. Outcome column name (default "woody_cover").                                                                           |
| <code>id</code>                  | Character. Unit identifier column (default "pair_id").                                                                            |
| <code>time</code>                | Character. Time column (default "year").                                                                                          |
| <code>treated_label</code>       | Character. Level of arm denoting treated units (default "credited").                                                              |
| <code>treatment_indicator</code> | Character. Name of the binary treatment indicator column (default "post_treatment"); must be 0/1 valued.                          |

## Details

This decomposition is the diagnostic the translation memo (RDT-8) relies on to flag "forbidden comparisons" - 2x2 cells in which an already-treated cohort acts as the comparator for a not-yet-treated cohort and so contaminates the TWFE coefficient with negative weights. A clean decomposition (large weight on Treated vs Untreated; small / non-negative weights on Earlier vs Later Treated) is a precondition for trusting the single-coefficient TWFE.

Operates on the *long-format* treated-arm panel: only rows whose arm column matches `treated_label` are passed to `bacondecomp::bacon()`, so each observation is a (pair\_id, year) pair with a single 0/1 treatment indicator.

**Value**

A list with components:

`decomposition` A data.frame returned by `bacondecomp::bacon()`; one row per 2x2 cell, with `treated`, `untreated`, `estimate`, `weight`, and `type`.

`twfe_estimate` Numeric. The implied weighted-average TWFE estimate.

`weight_summary` A data.frame summarising weights and weighted estimates by type.

**Examples**

```
if (requireNamespace("bacondecomp", quietly = TRUE)) {
  sim <- simulate_staggered_panel(seed = 20260520L, n_pairs = 30L,
                                n_proponents = 8L)
  gb <- goodman_bacon_decomp(sim$panel)
  gb$weight_summary
}
```

---

is\_bacipair\_abstention

*Is x a bacipair abstention?*

---

**Description**

Producer-side predicate; complements the orchestra-wide, producer-agnostic `is_orchestra_decline()` in `ORCHESTRA_dev/integration/refusal_contract.R`, which recognises this class (and every other member's) by the `_refusal/_abstention` naming convention without depending on `bacipair`'s namespace.

**Usage**

```
is_bacipair_abstention(x)
```

**Arguments**

`x` Any object, typically a caught condition.

**Value**

A length-1 logical.

**Examples**

```
sim <- simulate_staggered_panel(seed = 20260520L, n_pairs = 4L,
                                n_proponents = 2L)
cond <- tryCatch(
  synth_control_baci(sim$panel),
  bacipair_abstention = function(e) e
)
is_bacipair_abstention(cond)
```

---

```
proponent_cluster_bootstrap
```

*Proponent-level cluster bootstrap*

---

## Description

Resamples whole proponent clusters (with replacement) from a BACI panel and refits a chosen estimator on each draw. Returns the bootstrap distribution of the panel-average ATT, along with a 95% percentile interval and the bootstrap standard error.

## Usage

```
proponent_cluster_bootstrap(
  panel,
  estimator = c("synth_control", "paired_did"),
  n_boot = 200L,
  seed = 20260521L,
  cluster_col = "proponent_id",
  ...
)
```

## Arguments

|                          |                                                                                                                                                                                                                                                                                                                                                                                           |
|--------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>panel</code>       | A <code>data.table</code> with the columns required by the chosen estimator, plus the cluster column (default <code>proponent_id</code> ).                                                                                                                                                                                                                                                |
| <code>estimator</code>   | Character, one of <code>"synth_control"</code> (default) or <code>"paired_did"</code> . Selects which estimator to refit on each bootstrap draw.                                                                                                                                                                                                                                          |
| <code>n_boot</code>      | Integer ( $\geq 50$ ). Number of bootstrap draws (default 200).                                                                                                                                                                                                                                                                                                                           |
| <code>seed</code>        | Integer. RNG seed (default 20260521L).                                                                                                                                                                                                                                                                                                                                                    |
| <code>cluster_col</code> | Character. Cluster column name (default <code>"proponent_id"</code> ).                                                                                                                                                                                                                                                                                                                    |
| <code>...</code>         | Extra arguments forwarded to the chosen estimator. For <code>"synth_control"</code> , see <a href="#">synth_control_baci()</a> ( <code>predictor_cols</code> , <code>predictor_weight</code> , <code>min_pre_years</code> ). For <code>"paired_did"</code> , see <a href="#">baci_paired_did()</a> ( <code>outcome</code> , <code>treated_label</code> , <code>comparison_label</code> ). |

## Details

Cluster-level resampling is the right inferential unit when one proponent runs many adjacent projects (Macintosh et al., 2024) - resampling pairs would understate dependence.

## Value

List with components:

`draws` Numeric vector of length `n_boot` of point estimates per draw.  
`se` Numeric. Bootstrap standard deviation across draws.  
`ci_lo, ci_hi` Numeric. 2.5th / 97.5th percentile.  
`n_boot, estimator, seed` Bookkeeping.

## Examples

```
sim <- simulate_staggered_panel(seed = 20260520L, n_pairs = 30L,
                               n_proponents = 8L)
boot <- proponent_cluster_bootstrap(sim$panel, estimator = "paired_did",
                                   n_boot = 50L)

boot$ci_lo
boot$ci_hi
```

---

```
simulate_staggered_panel
```

*Simulate a staggered-treatment HIR-style BACI panel*

---

## Description

Generates a synthetic Before-After-Control-Impact (BACI) panel with staggered treatment timing, two arms ("credited" and "comparison"), per-pair static covariates (baseline cover, elevation, rainfall climatology), per-year shared rainfall anomaly, and proponent assignments drawn from a Zipf distribution. The data-generating process is structured so that the panel-average ATT is recoverable from the comparison arm; paired-DiD with the matched 1:1 comparator carries residual bias when elevation / rainfall climatology covary with treatment timing; and the donor-pool synthetic-control estimator should recover the oracle within the kill-gate tolerance documented in [synth\\_control\\_baci\(\)](#).

## Usage

```
simulate_staggered_panel(
  seed = 20260520L,
  n_pairs = 200L,
  t_min = 2010L,
  t_max = 2025L,
  treatment_year_min = 2013L,
  treatment_year_max = 2022L,
  additionality_share = 0.18,
  additionality_mean_pp = 4.5,
  additionality_sd_pp = 1.5,
  additionality_dynamics = c(0.2, 0.55, 0.85, 1, 1, 1, 1, 1, 1, 1),
  rainfall_sd_pp = 1.4,
  site_baseline_mean_pp = 22,
  site_baseline_sd_pp = 4,
  site_pair_sd_pp = 1,
  noise_sd_pp = 0.8,
  secular_trend_pp_per_year = 0.4,
  elevation_mean_m = 350,
  elevation_sd_m = 200,
  elevation_trend_coupling = 0.0015,
  rainfall_climatology_mean_pp = 0,
  rainfall_climatology_sd_pp = 0.6,
```

```

    elevation_treatment_year_coupling = -2.5,
    n_proponents = 50L,
    proponent_zipf_s = 1.2
)

```

## Arguments

**seed** Integer scalar. The simulation RNG seed (default 20260520L).

**n\_pairs** Integer scalar ( $\geq 2$ ). Number of treated/comparison pairs (default 200).

**t\_min, t\_max** Integer scalars. Inclusive panel year range (defaults 2010 / 2025).

**treatment\_year\_min, treatment\_year\_max** Integer scalars. Range of staggered treatment years (defaults 2013 / 2022). Must lie inside  $[t\_min, t\_max]$ .

**additionality\_share** Numeric in  $[0, 1]$ . Fraction of pairs with non-zero true additionality (default 0.18).

**additionality\_mean\_pp, additionality\_sd\_pp** Numeric. Mean / SD of the (untruncated) Normal asymptotic-additionality distribution among pairs with `has_real_additionality == TRUE`, in percentage points of woody cover (defaults 4.5 / 1.5). The draw is signed, not non-negative: a pair flagged as genuinely additional can realise a negative `true_additionality_max` when `additionality_mean_pp` is small relative to `additionality_sd_pp`, and that signed value flows unaltered into `oracle_panel_att`.

**additionality\_dynamics** Numeric vector of non-negative weights indexed by years-since-treatment (1, 2, ...). The realised individual treatment effect in year  $y$  is `true_additionality_max * additionality_dynamics[y - treatment_year + 1]`. Saturates at the last element.

**rainfall\_sd\_pp, noise\_sd\_pp** Numeric. Rainfall-anomaly and idiosyncratic noise SDs in percentage points of woody cover.

**site\_baseline\_mean\_pp, site\_baseline\_sd\_pp** Numeric. Mean and SD of the site-baseline draw in percentage points of woody cover.

**site\_pair\_sd\_pp** Numeric. Within-pair shift SD in percentage points of woody cover.

**secular\_trend\_pp\_per\_year** Numeric. Common secular trend in percentage points of woody cover per year (default 0.4).

**elevation\_mean\_m, elevation\_sd\_m** Numeric. Per-pair elevation distribution (defaults 350 / 200 m).

**elevation\_trend\_coupling** Numeric. Coupling between elevation and the secular trend (default 0.0015 pp/year per metre).

**rainfall\_climatology\_mean\_pp, rainfall\_climatology\_sd\_pp** Numeric. Per-pair rainfall-climatology offset distribution.

`elevation_treatment_year_coupling` Numeric. Strength of the selection-on-observables coupling between elevation and treatment year (default -2.5; higher elevation means earlier registration).

`n_proponents` Integer ( $1 \leq n\_proponents \leq n\_pairs$ ). Number of distinct proponents to assign (default 50).

`proponent_zipf_s` Numeric  $> 0$ . Zipf shape parameter for proponent cluster sizes (default 1.2).

### Details

All values, distributions, and seeds are explicit. There is no hidden RNG. Inputs are validated by `stopifnot()` guards.

### Value

A list with elements:

`panel` A data.table with columns including `pair_id`, `proponent_id`, `arm`, `year`, `treatment_year`, `post_treatment`, `treated_arm`, `post_idx`, `baseline_cover`, `elevation_m`, `rainfall_climatology`, `rainfall_sensitivity`, `site_pair_shift`, `rainfall_anomaly`, `true_additionality_max`, `has_real_additionality`, `ite`, `noise`, `woody_cover`.

`pair_table` A data.table of per-pair covariates.

`rainfall` A data.table of per-year shared rainfall anomaly.

`oracle_panel_att` Numeric scalar. The oracle panel-average ATT (mean ITE over treated post-treatment observations).

`oracle_unit_att` A data.table of per-unit oracle ATT (`pair_id`, `unit_att`).

`parameters` Named list of every parameter used to generate the panel, plus `generated_at` ISO-8601 timestamp.

### Examples

```
sim <- simulate_staggered_panel(seed = 20260520L, n_pairs = 50L)
sim$oracle_panel_att
head(sim$panel)
```

---

|                                 |                                                                        |
|---------------------------------|------------------------------------------------------------------------|
| <code>synth_control_baci</code> | <i>Fit a time-varying synthetic-control comparator on a BACI panel</i> |
|---------------------------------|------------------------------------------------------------------------|

---

### Description

For each treated unit, constructs a per-unit synthetic-control comparator by minimising the squared distance between the treated pre-treatment trajectory + static predictors and a simplex-constrained weighted average of every other pair's comparison-arm trajectory + predictors. Returns per-(pair, year) ATT estimates and the panel-average ATT.

**Usage**

```
synth_control_baci(
  panel,
  predictor_cols = c("baseline_cover", "elevation_m", "rainfall_climatology"),
  predictor_weight = 20,
  min_pre_years = 3L
)
```

**Arguments**

**panel** A `data.table` with columns `pair_id`, `arm`, `year`, `treatment_year`, `post_treatment`, `woody_cover`, plus all entries of `predictor_cols`. An `ite` column (the simulator's oracle individual treatment effect) is read and reported back when present, but is never required – the estimator runs on a real register that has no such column.

**predictor\_cols** Character vector of predictor column names (default `c("baseline_cover", "elevation_m", "rainfall_climatology")`).

**predictor\_weight** Numeric  $\geq 0$ . Relative weight on the predictor block (default 20).

**min\_pre\_years** Integer  $\geq 2$ . Minimum pre-treatment years required per treated unit (default 3).

**Details**

Pure base R; deterministic; no extra dependencies. The treated unit's own matched 1:1 comparator is excluded from its donor pool to avoid leakage; the pool is every *other* pair's comparison arm.

**Value**

List with components:

**fits** List of per-unit fits.

**per\_unit\_att** `data.table` of per-unit estimated ATT, plus the oracle ATT (`oracle_unit`) when the input panel carried an `ite` column.

**panel\_att\_hat** Numeric. Panel-average estimated ATT.

**pre\_fit\_rmse** Numeric. Root-mean-square pre-fit residual across treated units.

**Examples**

```
sim <- simulate_staggered_panel(seed = 20260520L, n_pairs = 30L,
                               n_proponents = 8L)
fit <- synth_control_baci(sim$panel)
fit$panel_att_hat
```

---

```
write_oracle_recovery_log
```

*Write the oracle-recovery JSON log*

---

## Description

Captures the comparison between the simulated oracle ATT and the point estimates from the synthetic-control and paired-DiD estimators (plus the cluster-bootstrap CI on the SC estimator), in the schema used by deliverables/RDT-20/logs/oracle-recovery.json.

## Usage

```
write_oracle_recovery_log(
  panel,
  synth_control_fit,
  paired_did_fit = NULL,
  cluster_boot = NULL,
  path
)
```

## Arguments

|                                |                                                                                    |
|--------------------------------|------------------------------------------------------------------------------------|
| <code>panel</code>             | A simulator return list (with <code>oracle_panel_att</code> ).                     |
| <code>synth_control_fit</code> | A return list from <code>synth_control_baci()</code> .                             |
| <code>paired_did_fit</code>    | A return list from <code>baci_paired_did()</code> (or NULL to skip the DiD block). |
| <code>cluster_boot</code>      | A return list from <code>proponent_cluster_bootstrap()</code> (or NULL).           |
| <code>path</code>              | Character. Output JSON path.                                                       |

## Details

Requires `jsonlite`; declared in `Suggests`.

## Value

Invisibly returns `path`. Side effect: writes the log.

## Examples

```
if (requireNamespace("jsonlite", quietly = TRUE)) {
  sim <- simulate_staggered_panel(seed = 20260520L, n_pairs = 30L,
                                n_proponents = 8L)

  sc <- synth_control_baci(sim$panel)
  did <- baci_paired_did(sim$panel)
  log_path <- tempfile(fileext = ".json")
  write_oracle_recovery_log(sim, sc, did, path = log_path)
  jsonlite::fromJSON(log_path)$sc_minus_oracle
}
```

# Index

baci\_paired\_did, [2](#)  
baci\_paired\_did(), [4](#), [7](#), [12](#)  
bacipair\_targets\_plan, [3](#)  
  
goodman\_bacon\_decomp, [5](#)  
  
is\_bacipair\_abstention, [6](#)  
  
proponent\_cluster\_bootstrap, [7](#)  
proponent\_cluster\_bootstrap(), [4](#), [12](#)  
  
simulate\_staggered\_panel, [8](#)  
simulate\_staggered\_panel(), [4](#), [5](#)  
synth\_control\_baci, [10](#)  
synth\_control\_baci(), [4](#), [7](#), [8](#), [12](#)  
  
write\_oracle\_recovery\_log, [12](#)