Package 'badcontrols'

September 12, 2026

Title Difference-in-Differences with Bad Controls

Version 1.0.1

Description Implements methods for difference-in-differences with bad controls, i.e., time-varying covariates that are affected by the treatment. Provides imputation, doubly robust, and machine learning estimators that are based on Caetano, Callaway, Payne, and Sant'Anna (2026) <doi:10.48550/arXiv.2608.03881>.

URL https://github.com/hugosantanna/badcontrols

BugReports https://github.com/hugosantanna/badcontrols/issues

License GPL (>= 3)

Encoding UTF-8

Depends R (>= 4.1.0)

Imports ptetools (>= 1.0.1), stats

Suggests grf (>= 2.0.0), testthat (>= 3.0.0), knitr, quarto

VignetteBuilder quarto

Config/testthat/edition 3

Config/roxygen2/version 8.1.0

NeedsCompilation no

Author Carolina Caetano [aut],
Brantly Callaway [aut],
Stroud Payne [aut],
Hugo Sant'Anna [aut, cre]

Maintainer Hugo Sant'Anna <hsantanna@uab.edu>

Repository CRAN

Date/Publication 2026-09-12 14:40:30 UTC

Contents

`didbc` *Estimate ATT(g,t) with Bad Controls*

Description

Main function for difference-in-differences with bad controls. Wraps the `ptetools` infrastructure to handle time-varying covariates affected by treatment. Supports both imputation and doubly robust ML estimation methods.

Usage

```
didbc(
  yname,
  gname,
  tname,
  idname = NULL,
  data,
  bad_control_formula = NULL,
  xformula = ~1,
  d_covs_formula = NULL,
  bad_control_cov_formula = NULL,
  bad_control_d_cov_formula = NULL,
  est_method = c("imputation", "dr_ml"),
  bad_control_identification_strategy = c("unconfoundedness", "did"),
  nuisance_method = c("ml", "parametric"),
  control_group = "notyettreated",
  anticipation = 0,
  base_period = "varying",
  weightsname = NULL,
  cband = TRUE,
  alp = 0.05,
  boot_type = "multiplier",
  bstrap = TRUE,
  cl = 1,
  biters = 100,
  nfolds = 5,
  overlap_threshold = 0.99,
  min_group_size = 5,
  num_threads = NULL,
  ...
)
```

Arguments

`yname`	Name of the outcome variable (string)

`gname` Name of the group variable (string). Should be 0 for never-treated units and the period of first treatment for treated units.

`tname` Name of the time period variable (string)

`idname` Name of the unit identifier variable (string); default `NULL`, matching `ptetools::pte`

`data` A balanced panel data.frame

`bad_control_formula`
One-sided formula naming exactly one bad control variable (a time-varying covariate affected by treatment), e.g. `~X`. `NULL` (the default) means no bad control at all, in which case `didbc` reduces to plain DiD with covariates. Multiple bad controls are not yet supported.

`xformula` One-sided formula for general exogenous covariates, entered as their pre-treatment-period level (default `~1`)

`d_covs_formula` One-sided formula for general exogenous covariates, entered as their change (post minus pre) rather than a level. Default `NULL` (unused).

`bad_control_cov_formula`
One-sided formula for auxiliary covariates (W in the paper) used to model the bad control's counterfactual evolution, entered as their pre-treatment-period level. Any variable name can be used here, including the outcome itself (e.g. `~Y` reproduces the old "lagged outcome as W" behavior). Default `NULL` (unused, and ignored if `bad_control_formula` is `NULL`).

`bad_control_d_cov_formula`
Like `bad_control_cov_formula`, but entered as a change (post minus pre) rather than a level. Default `NULL` (unused).

`est_method` Estimation method: `"imputation"` (default) for the two-step imputation approach, or `"dr_ml"` for the doubly robust estimator.

`bad_control_identification_strategy`
Which assumption identifies the bad control's untreated potential evolution: `"unconfoundedness"` (default), i.e. Covariate Unconfoundedness for the bad control given its own pre-period level and `(W, Z)`, or `"did"`, i.e. parallel trends for the bad control itself given `(W, Z)` (`app:bad-control-parallel-trends` in the supplementary appendix). `"did"` is only supported for `est_method = "imputation"`, requires `bad_control_formula` to be non-`NULL`, and does not distinguish a binary bad control.

`nuisance_method`
For `est_method = "dr_ml"` only: `"ml"` (default) estimates the doubly robust estimator's nuisance functions via cross-fitted random forests; `"parametric"` estimates them via OLS/logistic regression, assuming correct specification. Ignored for `est_method = "imputation"`.

`control_group` Which units serve as controls: `"notyettreated"` (default) or `"nevertreated"`. Passed to `ptetools::two_by_two_subset`.

`anticipation` Number of periods before treatment where it can already affect the outcome (default 0). Passed to `ptetools::two_by_two_subset`.

`base_period` Which pre-period to compare each post-period to: `"varying"` (default) or `"universal"`. Passed to `ptetools::two_by_two_subset`.

`weightsname` Name of a sampling-weights variable in `data` (default `NULL`)

cband	Logical; compute a uniform confidence band instead of pointwise confidence intervals (default TRUE)
alp	Significance level (default 0.05)
boot_type	"multiplier" (default) or "empirical"
bstrap	Logical; whether to use the multiplier bootstrap (TRUE, the default) or purely analytical standard errors (FALSE) when the estimator returns an influence function. bstrap = FALSE only supports pointwise confidence intervals.
cl	Number of clusters for parallel computing in the bootstrap (default 1)
biters	Number of bootstrap iterations (default 100)
nfolds	Number of cross-fitting folds for DR/ML (default 5)
overlap_threshold	For est_method = "dr_ml" only: if any unit's fitted propensity (from a preliminary, non-cross-fit fit) exceeds this value in a (g,t) cell, estimation falls back to the imputation estimator for that cell, with a warning naming the group, time period, and offending unit IDs. Default 0.99.
min_group_size	For est_method = "dr_ml" only: if a (g,t) cell has fewer treated units than the number of propensity-score covariates plus min_group_size, estimation falls back to the imputation estimator for that cell, with a warning naming the group, time period, and treated count – a separate trigger from overlap_threshold, since fitting the propensity score needs a nontrivial treated sample to identify at all, relative to how many covariates it has to fit, which can fail even under good covariate overlap. Default 5, matching did's own convention for an analogous check.
num_threads	For est_method = "dr_ml", nuisance_method = "ml" only: number of threads for grf's forests. Default NULL uses grf's own auto-detection (all available cores). Set to 1 to pin a single thread per forest – e.g. for Monte Carlo work, where many reps are run in outer parallelism (one core each) rather than letting each individual forest fit claim every core on the machine.
...	Additional arguments passed to ptetools::pte

Details

This function implements the methods from Caetano, Callaway, Payne, and Sant'Anna (2026). Two estimation approaches are available:

Imputation (est_method = "imputation"):

1. Among controls, learn X_t ~ f(X(t-1), W, Z)
2. For treated, predict counterfactual X_t(0)
3. Run DiD using imputed X_t(0) instead of observed X_t

Doubly Robust ML (est_method = "dr_ml"):

1. Estimate four nuisance functions via random forests
2. Combine into a doubly robust score with cross-fitting
3. Returns influence function for fast inference

Requires the grf package. Consistent if either the outcome regression or the propensity score is correctly specified.

Value

A `pte_results` object containing:

overall_att Overall ATT estimate and SE

att_gt Group-time specific ATT estimates

event_study Dynamic (event-time) ATT estimates and SEs

ptep Internal `ptetools` parameters object, used by `summary()`/`plot()`/`autoplot()` methods

References

Caetano, C., Callaway, B., Payne, S., and Sant'Anna, H. (2026). "Difference-in-Differences with Bad Controls." arXiv preprint arXiv:2608.03881. doi:10.48550/arXiv.2608.03881

Examples

```
# Simulate data
sim <- simulate_bad_controls(n = 500, T_max = 4)
head(sim$data)

# Imputation approach

res_imp <- didbc(
  yname = "Y", gname = "G", tname = "period", idname = "id",
  data = sim$data,
  bad_control_formula = ~X,
  xformula = ~Z,
  est_method = "imputation"
)
```

---

dr_ml_attgt *Doubly Robust Estimator for Bad Controls*

Description

Implements the semiparametric doubly robust estimator from Caetano, Callaway, Payne, and Sant'Anna, with nuisance functions estimated either via machine learning or via parametric (linear/logit) working models, and K-fold cross-fitting. Estimates ATT(g,t) in the presence of post-treatment covariates (bad controls).

Usage

```
dr_ml_attgt(
  gt_data,
  xformula = ~1,
  bad_control_formula = NULL,
```

```
  d_covs_formula = NULL,
  bad_control_cov_formula = NULL,
  bad_control_d_cov_formula = NULL,
  nuisance_method = c("ml", "parametric"),
  bad_control_binary = FALSE,
  overlap_threshold = 0.99,
  min_group_size = 5,
  n_folds = 5,
  num_threads = NULL,
  ...
)
```

Arguments

gt_data	data.frame from `ptetools::two_by_two_subset` with columns id, D, period, name (pre/post), Y, plus covariate columns
xformula	One-sided formula for general exogenous covariates, entered as their pre-treatment-period level (default ~1)
bad_control_formula	One-sided formula naming exactly one bad control variable (a time-varying covariate affected by treatment). `NULL` means no bad control at all; see Details. Multiple bad controls are not yet supported.
d_covs_formula	One-sided formula for general exogenous covariates, entered as their change (post minus pre) rather than a level. Used only in the outcome regression. Default `NULL` (unused).
bad_control_cov_formula	One-sided formula for auxiliary covariates (W in the paper) used to model the bad control's counterfactual evolution and the propensity score, entered as their pre-treatment level. Any variable name can be used, including the outcome itself (e.g. `~Y` reproduces the old "lagged outcome as W" behavior). Default `NULL` (unused, ignored if `bad_control_formula` is `NULL`).
bad_control_d_cov_formula	Like `bad_control_cov_formula`, but entered as a change; used only in the outcome regression, not the propensity score. Default `NULL` (unused).
nuisance_method	`"ml"` (default) estimates all four nuisance functions via cross-fitted random forests; `"parametric"` estimates them via OLS/logistic regression, assuming correct specification. See Details.
bad_control_binary	Logical; whether the bad control is binary (detected automatically by `didbc()`). Unused by `dr_ml_attgt()` itself (the bad control never appears as a modeled response here), but passed through to `imputation_attgt()` in case of a fallback on overlap; see Details.
overlap_threshold	If any unit's fitted propensity (from a preliminary, non-cross-fit fit) exceeds this value, estimation falls back to `imputation_attgt()` for that (g,t) cell. Default 0.99.

`min_group_size` If a (g,t) cell has fewer treated units than the number of p_2 covariates plus `min_group_size`, estimation falls back to `imputation_attgt()` for that cell (a separate trigger from `overlap_threshold`; see Details). Default 5, matching `did`'s own convention for an analogous check.

`n_folds` number of cross-fitting folds (default 5)

`num_threads` Number of threads for `grf`'s forests under `nuisance_method = "ml"`. Default `NULL` uses `grf`'s own auto-detection (all available cores). Set to 1 to pin a single thread per forest – e.g. for Monte Carlo work, where many reps are run in outer parallelism (one core each) rather than letting each individual forest fit claim every core on the machine. Unused under `nuisance_method = "parametric"` (`lm`/`glm` have no thread concept).

`...` additional arguments (unused)

Details

Implements Algorithm 1 from the paper: four nuisance functions, with K-fold cross-fitting:

- m_0(X_t*, X_t*-1, Z): outcome regression `E[DeltaY | X_t*, X_t*-1, Z, D=0]`
- nu_0(X_t*-1, W, Z): `E[m_0(X_t*, X_t*-1, Z) | X_t*-1, W, Z, D=0]`, a second-stage regression of m_0's fitted values on (X_t*-1, W, Z) among untreated units
- p_2(X_t*-1, W, Z): the propensity score P(D=1 | X_t*-1, W, Z)
- omega_0(X_t*, X_t*-1, Z): `E[p_2/(1-p_2) | X_t*, X_t*-1, Z, D=0]`, a regression of the fitted propensity odds ratio on (X_t*, X_t*-1, Z) among untreated units

`nuisance_method = "ml"` (the default) estimates all four via cross-fitted random forests (`grf` package). `nuisance_method = "parametric"` estimates m_0/nu_0/omega_0 via OLS and p_2 via logistic regression; this assumes all four working models are correctly specified, in which case the doubly robust score's Neyman orthogonality means no further correction to the influence function is needed beyond what is already here for either choice of `nuisance_method` – parametric estimators converge faster than the ML rate conditions this orthogonality argument requires. Cross-fitting is retained under `"parametric"` for a single, shared code path, even though it is not required for validity there.

Unlike the imputation estimator, no counterfactual imputation of the bad control is needed: X_t* is observed directly for untreated units (X_t* = X_t*(0)), and nu_0/omega_0 target m_0's/p_2's fitted values through a second regression rather than plugging in a predicted X_t*(0).

nu_0 and omega_0 are themselves nested conditional expectations of m_0/p_2 (e.g. nu_0 = E[m_0 | X_t*-1, W, Z, D=0]), so regressing m_0's/ p_2's fitted values on the coarser feature set using the same training-fold observations that fit m_0/p_2 would be a generated- regressor problem: the pseudo-outcome's estimation error is correlated with the very sample the nested regression is fit on, not just small in L^2. Under `nuisance_method = "ml"`, this is avoided using each forest's out-of-bag (OOB) predictions – `grf`'s `predict(fit)` with no `newdata`, which averages only over trees that did not include a given row in their subsample – as the pseudo-outcome for nu_0/omega_0's training data, instead of the in-sample `predict(fit, newdata = <training data>)`. This costs nothing extra to fit (no additional forests, unlike an explicit training-fold split) and keeps the full training fold available to nu_0/omega_0, at the cost of a less clean-cut independence argument than literal disjoint subsamples would give. The main m_0/p_2 fits used directly in the doubly robust score are unaffected – they are already evaluated out-of-sample via the outer K-fold. Under

nuisance_method = "parametric", no OOB equivalent applies (lm/glm have no such notion); that path is unchanged, per the paragraph above.

The doubly robust score is consistent if either (m_0, nu_0) or (p_2, omega_0) are correctly specified.

With no bad control at all (bad_control_formula = NULL), nu_0 and omega_0 are not estimated at all: nu_0 = m_0 and omega_0 = p_2/(1-p_2) exactly (m_0 no longer depends on X_t*, so there is nothing left for nu_0 to marginalize over; p_2 is already a function of Z alone, so omega_0's further conditioning on Z is a no-op). The doubly robust score then reduces exactly to the classical Sant'Anna and Zhao (2020) AIPW-DiD estimator.

There are two triggers that fall back to imputation_attgt() for a whole (g,t) cell, rather than dropping p_2/omega_0 alone – doing that would leave a moment that is not Neyman orthogonal, which is exactly why imputation_attgt() needs its own first-stage correction terms that this function's nuisances don't have:

1. **Small treated group.** If the cell has fewer treated units than the number of p_2 covariates plus min_group_size, a warning names the group, time period, and treated count. Fitting p_2 needs a nontrivial treated sample to identify at all, relative to how many covariates it has to fit (a logit can hit perfect separation, a probability_forest is essentially memorizing a handful of points), which can happen even when the population-level covariate distributions overlap fine – a distinct problem from overlap below.
2. **Overlap.** Before cross-fitting, a preliminary propensity model (not used in the final estimation) is fit once on the whole cell: if any unit's fitted propensity exceeds overlap_threshold, a warning names the group, time period, and offending unit IDs.

Value

attgt_if object with ATT estimate and influence function

References

Caetano, C., Callaway, B., Payne, S., and Sant'Anna, H. (2026). "Difference-in-Differences with Bad Controls." arXiv preprint arXiv:2608.03881. doi:10.48550/arXiv.2608.03881

Sant'Anna, P.H.C. and Zhao, J. (2020). "Doubly Robust Difference-in-Differences Estimators." *Journal of Econometrics*.

Wager, S. and Athey, S. (2018). "Estimation and Inference of Heterogeneous Treatment Effects using Random Forests." *Journal of the American Statistical Association*.

Examples

```
## Not run:
# dr_ml_attgt() is normally called internally as didbc()'s attgt_fun
# (via ptetools::pte()), not invoked directly. This is the standard way
# to reach it end-to-end; slow due to cross-fitted random forests plus
# the multiplier bootstrap, hence \dontrun.
sim <- simulate_bad_controls(n = 500)
res <- didbc(
  yname = "Y", gname = "G", tname = "period", idname = "id",
  data = sim$data,
  bad_control_formula = ~X,
```

```
  xformula = ~Z,
  est_method = "dr_ml",
  nuisance_method = "parametric"
)
summary(res)

## End(Not run)
```

`nlsy_job_displacement` *NLSY79 Job Displacement Application Data*

Description

A balanced panel of 3,231 NLSY79 respondents observed biennially from 1992 through 2002. The sample applies the paper's positive-earnings restriction and excludes respondents first treated in 1992, who have no pre-treatment period within the application window. The outcome is log earnings, the bad control is an occupation-based wage score, and treatment is job displacement.

Usage

```
data(nlsy_job_displacement)
```

Format

A data.frame with 19,386 rows and 8 columns: `id`, `year`, `log_earnings`, `occ_score`, `group`, `race`, `female`, and `educ_max_grade`.

Details

Job displacement is defined as involuntarily leaving a job because of a layoff/job elimination or plant/company/workplace closure. The occupation score is time-varying at the individual level because respondents can change occupations, even though the score is fixed for a given occupation. Respondents first displaced in 1992 are excluded because the application requires a pre-treatment period within the 1992–2002 window.

This is a researcher-created subset of public-use NLSY79 data and an occupation-score merge based on the public-use IPUMS USA 1990 5\ It is not an official NLSY79 data release. See the source files in `data-raw/` for the construction script and provenance.

Source

National Longitudinal Survey of Youth 1979 (NLSY79), U.S. Bureau of Labor Statistics, https://www.nlsinfo.org/investigator
IPUMS USA, https://usa.ipums.org/usa/.

References

Bureau of Labor Statistics, U.S. Department of Labor. National Longitudinal Survey of Youth 1979 cohort, 1979-2022 (rounds 1-30). Produced and distributed by the Center for Human Resource Research (CHRR), The Ohio State University. Columbus, OH.

Steven Ruggles, Sarah Flood, Matthew Sobek, Daniel Backman, Grace Cooper, Julia A. Rivera Drew, Stephanie Richards, Renae Rogers, Jonathan Schroeder, and Kari C.W. Williams. IPUMS USA: Version 16.0 [dataset]. Minneapolis, MN: IPUMS, 2025. https://doi.org/10.18128/D010.V16.0

Examples

```
data(nlsy_job_displacement)
head(nlsy_job_displacement)
table(nlsy_job_displacement$group[!duplicated(nlsy_job_displacement$id)])
```

---

simulate_bad_controls *Simulate Panel Data with Bad Controls*

Description

Generates a staggered difference-in-differences panel dataset matching the Monte Carlo designs in Caetano, Callaway, Payne, and Sant'Anna (2026), where a time-varying covariate X is affected by treatment (a bad control). Returns the known group-time, event-study, and overall ATT alongside the data for testing estimators against.

Usage

```
simulate_bad_controls(
  n = 2000,
  T_max = 4,
  groups = 2:T_max,
  dgp = c("dgp1", "dgp2", "dgp3", "dgp4", "dgp5"),
  lambda = 0.5,
  delta = 0.5,
  kappa = 0.5,
  beta_drift = 0.2,
  binary_bad_control = FALSE
)
```

Arguments

n	Number of units (default 2000)
T_max	Number of time periods (default 4)
groups	Integer vector of possible treatment-adoption periods, besides never-treated (default 2:T_max). Values must be between 2 and T_max, since every unit needs at least one pre-period.

`dgp` Which counterfactual evolution equation for X_t(0) to use: `"dgp1"` (linear W), `"dgp2"` (nonlinear W), `"dgp3"` (nonlinear in (X(t-1), Z), no W – Simple Covariate Unconfoundedness holds), `"dgp4"` (linear W, coefficient 1 on the lag – parallel trends for the bad control given (W,Z) holds exactly, see Details), or `"dgp5"` (nonlinear W plus an $X_{t-1} \times W$ interaction – something a linear model cannot represent at all, unlike a curved but additive term; see Details)

`lambda` How much treatment shifts X at event time 0 (default 0.5)

`delta` Direct effect of treatment on Y at event time 0, net of the effect transmitted through X (default 0.5)

`kappa` Growth rate of the treatment effect with event time `e = t - g`; effects at event time e are `(1 + kappa * e)` times their event-time-0 value (default 0.5)

`beta_drift` Drift rate of the loading on X(0) in the outcome equation across calendar time (default 0.2); see Details. Set to 0 to hold the loading fixed at 1 in every period.

`binary_bad_control`
Logical; if `TRUE`, the bad control X is binary instead of continuous, generated by treating the usual X(0) equations (baseline and dgp-specific evolution) as a logit index for a Bernoulli draw, and treatment as a shift in that index rather than a level shift in X. See Details.

Details

Common structure (shared across all dgp choices): $Z_i, \eta_i \sim N(0,1)$ and $W_i = 0.8\eta_i + 0.3Z_i + 0.2\varepsilon_i^W$ are time-invariant unit characteristics; η_i is unobserved. Treatment group is assigned by splitting units into equal-sized bins of the latent index $0.2Z_i + 0.4W_i + 0.3\eta_i + \varepsilon_i^D$, with the lowest bin never-treated and successive bins assigned to `groups` in increasing order.

$X_{i1} = 0.5\eta_i + 0.4Z_i + 0.3\varepsilon_i^{X_1}$, and for $t \geq 2$, $X_{it}(0)$ evolves according to the dgp-specific equation (redrawing fresh noise every period):

- dgp1: $0.7X_{i,t-1}(0) + 0.3Z_i + 0.2W_i + 0.15$. Matches the paper's actual Monte Carlo design, including the "Exclude BC" finding that dropping the bad control entirely stays nearly unbiased there specifically.
- dgp2: $0.7X_{i,t-1}(0) + 0.3Z_i + 0.2W_i + 0.03W_i^2 + 0.15$. The "nonlinear W" design, matching the paper's own DGP2 exactly. An earlier version of the paper's DGP2 used a larger quadratic coefficient ($0.15W_i^2$ added to dgp1's equation), which put the quadratic term's vertex at $W = -0.667$, well inside W's support ($SD(W) \approx 0.88$), making the map from W to $X_{it}(0)$ two-to-one for nearly the whole population – a pathological non-monotonicity, not just "nonlinearity," that specifically broke `dr_ml_attgt()`'s ω_0 (which conditions on $(X_{it}, X_{i,t-1}, Z)$, not W, and so must implicitly marginalize over a non-invertible W). This package's dgp2 keeps the same linear coefficient on W as dgp1 (0.2, so it still reads as "dgp1 plus one added wrinkle"), but with a small enough quadratic coefficient (0.03) that the vertex sits at $W = -3.33$ (about 3.8 SDs out) – safely beyond what any realistic sample reaches, while still contributing a real, detectable nonlinearity (about 13\ W). The paper's own DGP2 was subsequently redesigned to use this same $0.03W_i^2$ coefficient, so the two now match exactly.
- dgp3: $0.7X_{i,t-1}(0) + 0.3Z_i + 0.4X_{i,t-1}(0)Z_i + 0.2X_{i,t-1}(0)^2 + 0.15$ (no W). Matches the paper's actual Monte Carlo design.

- dgp4: $X_{i,t-1}(0) + 0.3Z_i + 0.2W_i + 0.15$. The coefficient of 1 on the lag (a random walk with drift in (Z, W)), unlike dgp1's 0.7, is deliberate: it makes $\Delta X_{it}(0) = X_{it}(0) - X_{i,t-1}(0)$ itself a function of (Z, W) alone (plus independent noise), so parallel trends for the bad control given (W, Z) (`app:bad-control-parallel-trends` in the supplementary appendix) holds exactly under dgp4, alongside Covariate Unconfoundedness given (X_{t-1}, W, Z). Does not attempt to match any Monte Carlo design in the paper, and exists only to test the parallel-trends identification strategy.
- dgp5: $0.7X_{i,t-1}(0) + 0.3Z_i + 0.2W_i + 0.03W_i^2 + 0.05X_{i,t-1}(0)W_i + 0.15$. dgp2's equation plus an $X_{t-1} \times W$ interaction. Unlike a curved-but-additive term, an interaction is something a linear model (as in `imputation_attgt()`'s Step 1) cannot represent at all, no matter how strong – a categorical, not just approximate, misspecification. Meant to show a bigger separation between Imputation and `dr_ml_attgt()` than dgp2 does, since forests capture interactions natively without needing to be told where to look. The interaction coefficient (0.05) was chosen empirically: it's the smallest of several tried that still gives Imputation clearly larger (roughly double or more) bias than dgp2 does, while keeping `dr_ml_attgt()`'s own analytical-SE calibration reasonable (larger coefficients tried, e.g. 0.10, gave a starker bias gap but degraded `nuisance_method = "ml"`'s SE calibration, apparently worsening rather than improving with `n` – a real, persistent problem rather than finite-sample noise, unlike dgp5's own milder behavior, which does improve with `n`).

each plus $0.3\varepsilon_i^{X_t}$.

The untreated outcome is $Y_{it}(0) = 0.3t + 0.5\eta_i + 0.3Z_i + \beta_t X_{it}(0) + 0.3\varepsilon_{it}^{Y}$, with a time-varying loading $\beta_t = 1 + \texttt{beta\_drift}(t-2)$ (equal to 1 at t=2 regardless of `beta_drift`). Once treated (period $t \geq G_i$, event time $e = t - G_i$), the observed covariate and outcome are $X_{it} = X_{it}(0) + \lambda_e$ and $Y_{it} = Y_{it}(0) + \beta_t\lambda_e + \delta_e$, where $\lambda_e = \lambda(1 + \kappa e)$ and $\delta_e = \delta(1 + \kappa e)$. The $\beta_t\lambda_e$ term routes the treatment's effect on X through the (time-varying) X-Y relationship, giving true $\mathrm{ATT}(g,t) = \beta_t\lambda_e + \delta_e$.

The true $\mathrm{ATT}(g,t)$ only ever depends on β_t at the post-treatment period t, which equals 1 whenever $t = 2$ regardless of `beta_drift` – so with the defaults, `T_max = 2, groups = 2` already gives true $\mathrm{ATT} = \delta + \lambda = 1.00$. But `beta_drift != 0` still makes $\beta_1 \neq \beta_2$, so an estimator that assumes a constant X-Y loading across periods (as linear Imputation implicitly does) will show bias in that case even though the true ATT value matches the paper. To reproduce the paper's two-period designs exactly, including the constant-loading assumption, set `beta_drift = 0` as well.

When `binary_bad_control = TRUE`, the same equations for X_i1 and X_it(0) (dropping their additive noise term) are used as a logit index instead of a direct value: $X_{i1} \sim \text{Bernoulli}(\Lambda(0.5\eta_i + 0.4Z_i))$ and $X_{it}(0) \sim \text{Bernoulli}(\Lambda(\text{evolve}_x))$, where Λ is the logistic CDF. Treatment shifts the index rather than the level: $X_{it}(1) \sim \text{Bernoulli}(\Lambda(\text{evolve}_x + \lambda_e))$. This reuses each `dgp`'s existing evolution equation unchanged, so `dgp1 + binary_bad_control = TRUE` is the case where the imputation estimator's binary Step 1 (logistic regression) is correctly specified, the same role `dgp1` plays for the continuous, OLS Step 1.

Value

A list with:

data Panel data.frame with columns id, period, G, D, Y, X, Z, W

true_att_gt data.frame(g, t, att): the true ATT(g,t) for every valid group-time cell. Computed exactly from the DGP parameters when `binary_bad_control = FALSE`; when `TRUE`, there is

no closed form (the shift in E(X) varies by unit), so this is instead the realized average over this sample's group-g cohort, same as `true_att_by_e`/`true_att_overall` below

true_att_by_e data.frame(e, att): the true event-study ATT at each realized event time, averaged over the (g,t) cells that share that event time in this sample

true_att_overall Realized sample average of the true individual effect across every treated (i,t) observation in this sample

Examples

```
sim <- simulate_bad_controls(n = 500, dgp = "dgp1")
head(sim$data)
sim$true_att_gt
sim$true_att_overall

# Collapses exactly to the paper's two-period design
sim2 <- simulate_bad_controls(n = 500, T_max = 2, groups = 2, dgp = "dgp1",
                              beta_drift = 0)
sim2$true_att_overall
```

Index