Package: cfcompare (via r-universe)

June 28, 2026

Title Compare DID, SDID, Matrix Completion and the Triply Robust Panel Estimator

Version 0.1.0

Description A comparison toolkit for binary-treatment panel causal inference. Runs difference-in-differences (two-way fixed effects), synthetic difference-in-differences, synthetic control, matrix completion, and the Triply RObust Panel (TROP) estimator of Athey, Imbens, Qu and Viviano (2026) <doi:10.1002/jae.70061> on the same data, and returns their average treatment effects on a single tidy schema with shared plots. TROP, DID and matrix completion are implemented natively; synthetic difference-in-differences and synthetic control are obtained through the 'synthdid' package, and an alternative matrix-completion / interactive fixed-effects estimator through 'gsynth'. This is an unofficial, independent implementation and is not affiliated with or endorsed by the authors of the TROP estimator.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

Depends R (>= 4.1)

Imports ggplot2, stats, utils

Suggests RSpectra, future, future.apply, patchwork, synthdid, gsynth, augsynth, did, fixest, testthat (>= 3.0.0), knitr, rmarkdown

URL https://github.com/takuma1102/cfcompare

BugReports https://github.com/takuma1102/cfcompare/issues

VignetteBuilder knitr

Remotes synth-inference/synthdid, ebenmichael/augsynth

Config/testthat/edition 3

Config/testthat/parallel true

Config/roxygen2/version 8.0.0

Repository https://takuma1102.r-universe.dev

Date/Publication 2026-06-27 23:25:37 UTC

RemoteUrl https://github.com/takuma1102/cfcompare

RemoteRef HEAD

RemoteSha 3b1568bbe38f2fccc0f03f0032dda618a6b69a2a

Contents

`as_att` *Coerce estimator output to the cfcompare ATT schema*

Description

Brings the output of `trop()`, `panel_compare()`, a **synthdid** estimate, or any object with an `estimate` into the common `cf_att_tbl` schema, so results computed elsewhere can be slotted into the same comparison and plots.

Usage

```
as_att(x, ...)
```

Arguments

`x`	An object to tidy.
`...`	Passed to methods (e.g. `method`, `outcome`, `conf_level`).

Value

A `cf_att_tbl` (a `data.frame`).

`autoplot.cf_comparison` *Forest plot of ATT estimates across methods*

Description

Point estimate and (where available) confidence interval for each method, on a single axis – the quick visual comparison this package is built for.

Usage

```
## S3 method for class 'cf_comparison'
autoplot(object, ...)

## S3 method for class 'cf_att_tbl'
autoplot(object, ...)
```

Arguments

`object`	A `cf_comparison` (from `panel_compare()`) or a `cf_att_tbl`.
`...`	Unused.

Value

A **ggplot2** object.

Examples

```
df <- sim_panel(seed = 1)
cmp <- panel_compare(df, "y", "w", "id", "t",
                     methods = c("DID", "MC", "TROP"))
autoplot(cmp)
```

`autoplot.cf_rmse_curve` *Line plot of estimation RMSE versus a design dimension*

Description

One line per estimator; the y axis is on a log scale (as in the paper). Lower is better.

Usage

```
## S3 method for class 'cf_rmse_curve'
autoplot(object, log_y = TRUE, ...)
```

Arguments

`object`	A `cf_rmse_curve` from `rmse_curve()`.
`log_y`	Logical; log10 y axis (default `TRUE`).
`...`	Unused.

Value

A `ggplot` object.

`autoplot.cf_rmse_curves` *Autoplot for paired RMSE curves*

Description

Autoplot for paired RMSE curves

Usage

```
## S3 method for class 'cf_rmse_curves'
autoplot(object, combined = FALSE, log_y = TRUE, ...)
```

Arguments

`object`	A `cf_rmse_curves`.
`combined`	`FALSE` (default) returns a named list of two `ggplots`; `TRUE` returns a single side-by-side `ggplot` (needs the `patchwork` package).
`log_y`	Logical; log10 y axis.
`...`	Unused.

Value

A list of two `ggplots`, or one combined `ggplot`.

autoplot.cf_rmse_tbl *Bar chart of out-of-sample RMSE across methods*

Description

Visualises a `panel_rmse()` result as a ranked bar chart (lowest RMSE first), with +/- 1 standard-error whiskers across placebo runs. This is the cross-model RMSE comparison from the paper.

Usage

```
## S3 method for class 'cf_rmse_tbl'
autoplot(object, ...)
```

Arguments

`object`	A `cf_rmse_tbl` from `panel_rmse()`.
`...`	Unused.

Value

A `ggplot` object.

autoplot.cf_trop_grid *Heatmap of the TROP penalty-sensitivity grid*

Description

Cells are coloured by cross-validation loss (darker = better out-of-sample fit) and annotated with the ATT estimate at that penalty pair; the CV-selected cell is outlined.

Usage

```
## S3 method for class 'cf_trop_grid'
autoplot(object, ...)
```

Arguments

`object`	A `cf_trop_grid` from `trop_sensitivity()`.
`...`	Unused.

Value

A `ggplot` object.

autoplot.trop *Synthetic-control-style trajectory plot for a single TROP fit*

Description

Draws the treated-unit average observed path against the estimated untreated (counterfactual) path, in the style of the **synthdid** plot: a dotted line marks the first treated period, the post-treatment gap between the two lines is the estimated effect, and – since TROP carries explicit time weights – the time weights θ_s are drawn as a ribbon along the bottom to show which periods the counterfactual leans on.

Usage

```
## S3 method for class 'trop'
autoplot(object, show_weights = TRUE, ...)
```

Arguments

`object`	A `trop` fit from `trop()`.
`show_weights`	Logical; draw the time-weight ribbon along the bottom.
`...`	Unused.

Value

A `ggplot` object.

Examples

```
df <- sim_panel(N = 20, T = 12, n_treated = 4, t0 = 9, att = 3, seed = 1)
autoplot(trop(df, "y", "w", "id", "t",
              control = trop_control(n_cv_cells = 8L, cv_cycles = 1L)))
```

panel_compare *Compare DID, SDID, MC and TROP on the same panel*

Description

Runs a set of panel estimators on a single long panel and returns their average-treatment-effect-on-the-treated (ATT) estimates on a common tidy schema, so applied researchers can compare them at a glance. DID, MC and TROP are computed natively (no external dependencies); SDID and SC are routed through the **synthdid** package when available, and an alternative MC/IFE can be routed through **gsynth**. Methods whose optional package is missing, or that do not apply to the design, are skipped with a message.

Usage

```
panel_compare(
  data,
  outcome,
  treatment,
  unit,
  time,
  methods = c("DID", "SDID", "MC", "TROP", "DIFP"),
  exclude = NULL,
  anchor = "auto",
  se = c("auto", "jackknife", "bootstrap", "placebo", "none"),
  control = trop_control(),
  verbose = FALSE
)
```

Arguments

`data`	A long `data.frame` with one row per unit-time.
`outcome`, `treatment`, `unit`, `time`	Column names (strings). `treatment` must be a 0/1 indicator of *active* treatment in that cell (works for block, staggered, and non-absorbing designs).
`methods`	Character vector of methods to run. Any of `"DID"`, `"SDID"`, `"SC"`, `"MC"`, `"DIFP"`, `"TROP"`, `"gsynth"`, `"augsynth"`, `"CS"`. Defaults to the native engines plus SDID: `c("DID", "SDID", "MC", "TROP", "DIFP")`.
`exclude`	Optional character vector of methods to drop from `methods` (after defaults are applied). Convenient for running "everything except one", e.g. `exclude = "DIFP"`. Unknown names are ignored with a warning.
`anchor`	Weight anchoring for TROP; see `trop()`.
`se`	Standard-error method for the native engines; see `trop()`.
`control`	A list of solver/CV settings from `trop_control()`.
`verbose`	Logical; print CV progress.

Value

An object of class `cf_comparison`: a list with the tidy table `att` (class `cf_att_tbl`), per-method counterfactual matrices `counterfactual`, the native fit objects `fits`, and the reshaped `panel`.

See Also

`trop()`, `autoplot.cf_comparison()`, `plot_counterfactual()`

Examples

```
df <- sim_panel(N = 25, T = 14, n_treated = 4, t0 = 10, seed = 3)
cmp <- panel_compare(df, "y", "w", "id", "t",
                     methods = c("DID", "MC", "TROP"), se = "none",
                     control = trop_control(n_cv_cells = 8L, cv_cycles = 1L))
cmp$att
```

panel_rmse *Out-of-sample RMSE across panel estimators*

Description

Compares estimators by how well each does on held-out control data, following the "random blocks" placebo idea of the doubly/triply robust panel estimator paper. Two scoring rules are available:

Usage

```
panel_rmse(
  data,
  outcome,
  treatment,
  unit,
  time,
  methods = c("DID", "SC", "SDID", "MC", "TROP", "DIFP"),
  exclude = NULL,
  metric = c("placebo", "prediction"),
  horizon = 10L,
  n_pseudo = 10L,
  n_runs = 10L,
  control = trop_control(),
  seed = NULL,
  verbose = FALSE
)
```

Arguments

data	A long data.frame, one row per unit-time.
outcome, treatment, unit, time	Column names (strings).
methods	Methods to compare; subset of "DID", "MC", "TROP", "DIFP", "SDID", "SC", "gsynth", "augsynth", "CS". Defaults to c("DID", "SC", "SDID", "MC", "TROP", "DIFP").
exclude	Optional character vector of methods to drop from methods (e.g. exclude = "DIFP"). Unknown names are ignored with a warning.
metric	"placebo" (placebo-ATT RMSE, all methods) or "prediction" (per-cell held-out RMSE, native methods "DID"/"MC"/"TROP" only; other methods, including "DIFP", are ignored for this metric).
horizon	Number of final periods held out per placebo cohort.
n_pseudo	Number of placebo (pseudo-treated) control units per run.
n_runs	Number of placebo runs to average over.
control	A list of solver/CV controls from trop_control().

seed	Optional integer seed for reproducible placebo draws.
verbose	Logical; print progress.

Details

- metric = "placebo" (default): in each run a random set of control units is given a *placebo* block treatment in the final horizon periods (true effect zero), every method is estimated on that control-only panel, and the score is sqrt(mean(ATT^2)) across runs. This works for **every** method – native (DID, MC, TROP) and wrapped (SDID/SC via **synthdid**, gsynth, augsynth, CS = Callaway & Sant'Anna via **did**).
- metric = "prediction": per-cell one-step-ahead held-out RMSE (the paper's Table-31 style). Implemented for the native methods (DID, MC, TROP) only.

Lower is better. Native methods are tuned once on the real data (DID is parameter-free; MC selects lambda_nn; TROP selects the full triplet by cross-validation) and reused across runs. Wrapped methods are skipped with a note when their package is missing or the design does not apply.

Value

A cf_rmse_tbl (a data.frame) with one row per method and columns method, rmse, rmse_se, n_runs, engine, note.

References

Athey, S., Imbens, G. W., Qu, Z., & Viviano, D. (2025/2026). Triply/Doubly Robust Panel Estimators.

See Also

panel_compare(), autoplot.cf_rmse_tbl()

Examples

```
df <- sim_panel(N = 20, T = 12, n_treated = 4, t0 = 9, att = 2, seed = 1)
r <- panel_rmse(df, "y", "w", "id", "t",
                methods = c("DID", "TROP"),
                horizon = 2, n_pseudo = 3, n_runs = 2,
                control = trop_control(n_cv_cells = 8L, cv_cycles = 1L),
                seed = 1)
r
autoplot(r)
```

`plot.cf_rmse_curve` *Plot one or both estimation-RMSE curves*

Description

Plot one or both estimation-RMSE curves

Usage

```
## S3 method for class 'cf_rmse_curve'
plot(x, log_y = TRUE, file = NULL, width = 8, height = 5, ...)

## S3 method for class 'cf_rmse_curves'
plot(
  x,
  combined = FALSE,
  log_y = TRUE,
  file = NULL,
  width = NULL,
  height = NULL,
  ...
)
```

Arguments

`x`	A `cf_rmse_curve` or `cf_rmse_curves`.
`log_y`	Logical; log10 y axis (default `TRUE`).
`file`	Optional path. If given, the figure is written to a PNG at `width` x `height` inches; for separate panels two files are written with `-control`/`-pre` suffixes. If `NULL`, draws to the current device.
`width, height`	Figure size in inches. Defaults are deliberately generous (single 8x5; combined 12x5).
`...`	Unused.
`combined`	For `cf_rmse_curves`: `FALSE` (default) draws the two panels separately; `TRUE` draws them side by side (paper-style).

Value

The input, invisibly.

plot_counterfactual *Plot observed vs predicted counterfactual trajectories*

Description

For each native engine in a comparison (DID, MC, TROP), draws the average outcome over the treated units against the predicted control counterfactual, to show how the methods extrapolate through the post-treatment period.

Usage

```
plot_counterfactual(x, methods = NULL)
```

Arguments

x	A cf_comparison from panel_compare().
methods	Optional subset of methods to draw.

Value

A **ggplot2** object.

rmse_curve *Estimation-RMSE curve over one design dimension*

Description

Sweeps one design dimension – the number of control units ("n_control") or the number of pre-treatment periods ("n_pre") – and, at each value, runs a semi-synthetic Monte Carlo: panels are drawn from a latent factor model in which treatment is selected on the factor loadings (so plain DID/TWFE is biased), a known constant effect att is imposed, every estimator is run, and the estimation RMSE against the known truth is recorded. The result is the data behind the paper-style "RMSE vs. N_control / T_pre" line plot, one line per estimator. With a dense values grid and enough n_runs the curves are smooth, as in the paper.

Usage

```
rmse_curve(
  vary = c("n_control", "n_pre"),
  values = NULL,
  n_runs = 500L,
  methods = c("DID", "SDID", "SC", "MC", "DIFP", "TROP"),
  exclude = NULL,
  n_control = 60L,
  n_treated = 8L,
```

```
  n_pre = 16L,
  n_post = 6L,
  rank = 4L,
  att = 2,
  noise = 1,
  ar = 0.4,
  trend_sd = 0.05,
  anchor = "pooled",
  control = trop_control(),
  seed = 1L,
  parallel = FALSE,
  verbose = FALSE
)
```

Arguments

vary	Which dimension to sweep: "n_control" or "n_pre".
values	Integer vector of values for the swept dimension. Defaults to seq(20, 45, by = 5) for control units and seq(5, 38, by = 2) for pre-periods. Each value is an actual measured point (its own Monte Carlo); pass a denser/wider grid for smoother or longer curves.
n_runs	Monte Carlo replications per value (higher = smoother; the paper uses ~1000). Default 500. **Note:** with the dense default grid and six estimators this is a large simulation (thousands of fits per panel); reduce n_runs and/or values for a quick look, or parallelise the outer loop.
methods	Estimators to include; any subset of the panel_compare() methods. Default c("DID", "SDID", "SC", "MC", "DIFP", "TROP"), the six estimators compared in the paper. gsynth, augsynth and CS can be added if those packages are installed.
exclude	Optional character vector of methods to drop from methods (e.g. exclude = "DIFP"). Unknown names are ignored with a warning.
n_control, n_treated, n_pre, n_post	Base design; the dimension named in vary is overridden by values, the rest are held fixed. Defaults give a sizeable panel (60 controls, 8 treated, 16 pre- and 6 post-periods).
rank, att, noise	Number of latent factors, the imposed (true) ATT, and the idiosyncratic-noise scale.
ar	AR(1) coefficient of the idiosyncratic errors (serial correlation).
trend_sd	SD of the heterogeneous unit-specific linear trend slopes (non-parallel trends); treatment is also selected on this slope.
anchor	Estimation anchor for the native ATT ("pooled" by default for speed across the many fits).
control	Solver/CV controls from trop_control().
seed	Base integer seed (each replication uses a distinct offset).

parallel — Logical; if TRUE and the future.apply package is installed, the Monte Carlo tasks are run in parallel honouring the active future plan (e.g. future::plan(future::multisession, workers = 6)). Results are identical to the sequential default because each task is self-seeded. Default FALSE.

verbose — Logical; print progress per swept value.

Details

This is a simulation diagnostic on synthetic data; it does not take a user panel. To run a curve on your own data, draw panels with sim_semisynthetic() and call panel_compare() in a loop.

Value

A cf_rmse_curve (a data.frame) with columns method, x (the swept value), rmse, bias, n_runs. The swept-dimension label is in attr(., "vary").

See Also

rmse_curves(), panel_rmse(), autoplot.cf_rmse_curve()

Examples

```
# quick look (small grid + few reps); raise n_runs/values for paper quality
cc <- rmse_curve("n_control", values = c(30, 50), n_runs = 2,
                 methods = c("DID", "TROP"),
                 control = trop_control(n_cv_cells = 8L, cv_cycles = 1L))
autoplot(cc)
```

---

rmse_curves *Estimation-RMSE curves over both design dimensions*

Description

Convenience wrapper that runs rmse_curve() for both the number of control units and the number of pre-treatment periods and bundles the two curves, so they can be plotted individually (default) or side by side (combined = TRUE), as in the paper's two-panel figure.

Usage

```
rmse_curves(values_control = NULL, values_pre = NULL, ...)
```

Arguments

values_control, values_pre — Optional grids for each sweep (see rmse_curve() defaults).

... — Passed to rmse_curve() (n_runs, methods, base design, att, noise, control, seed, parallel, verbose, ...). Set parallel = TRUE (plus a future::plan) to run both sweeps in parallel.

Value

A `cf_rmse_curves` object: a list with `$n_control` and `$n_pre`, each a `cf_rmse_curve`.

See Also

`rmse_curve()`, `autoplot.cf_rmse_curves()`

Examples

```
# defaults are a large simulation; use a small grid + few reps for a quick look
g <- rmse_curves(values_control = c(30, 50), values_pre = c(8, 14),
                 n_runs = 2, methods = c("DID", "TROP"),
                 control = trop_control(n_cv_cells = 8L, cv_cycles = 1L))
plot(g)                  # two separate figures (default)
# combined = TRUE needs the optional 'patchwork' package:
if (requireNamespace("patchwork", quietly = TRUE))
  plot(g, combined = TRUE) # side-by-side, paper-style
```

---

sim_panel *Simulate a panel from a low-rank factor model*

Description

Generates a long panel whose untreated potential outcomes follow an interactive-fixed-effects (factor) model on top of two-way fixed effects, in the spirit of the data-generating processes in Athey, Imbens, Qu & Viviano (2025). A block treatment is applied to `n_treated` units from period `t0` onward, with a constant additive effect `att`.

Usage

```
sim_panel(
  N = 30,
  T = 20,
  n_treated = 5,
  t0 = NULL,
  rank = 3L,
  att = 1,
  noise = 0.5,
  seed = NULL
)
```

Arguments

`N`	Number of units.
`T`	Number of periods.
`n_treated`	Number of treated units.

t0	First treated period (block design).
rank	Number of latent factors.
att	True treatment effect added to treated cells.
noise	Standard deviation of idiosyncratic noise.
seed	Optional RNG seed.

Value

A long data.frame with columns id, t, y, w, and the noiseless counterfactual y0 (useful for evaluating estimators).

Examples

```
df <- sim_panel(N = 30, T = 15, n_treated = 5, t0 = 11, seed = 42)
head(df)
```

---

sim_semisynthetic *Build a semi-synthetic panel from real data*

Description

Takes a real long panel, uses its outcomes (optionally smoothed through a low-rank-plus-two-way-fixed-effects fit) as the untreated potential outcomes Y(0), and imposes a *known* treatment effect on a chosen block of units and periods. Because the baseline is real data but the effect is known, the result is a ground-truth benchmark that "closely matches" the real setting – the style of semi-synthetic experiment used to evaluate panel estimators in Athey, Imbens, Qu & Viviano (2026). Pair it with panel_compare() or panel_rmse() to score estimators against the truth.

Usage

```
sim_semisynthetic(
  data,
  outcome,
  unit,
  time,
  n_treated,
  t0 = NULL,
  att = 1,
  effect = NULL,
  baseline = c("observed", "lowrank"),
  lambda_nn = NULL,
  noise = 0,
  seed = NULL
)
```

Arguments

data	A real long data.frame, one row per unit-time.
outcome, unit, time	Column names (strings).
n_treated	Number of units to assign to the placebo treated group (sampled at random from all units).
t0	First treated period (block design). Defaults to about three quarters of the way through the panel.
att	Constant additive treatment effect imposed on treated cells. Ignored if effect is supplied.
effect	Optional per-treated-period effect: a single number, or a numeric vector of length T - t0 + 1 giving a dynamic effect path.
baseline	"observed" uses the real outcomes directly as Y(0); "lowrank" replaces them with a low-rank + two-way-FE fit (optionally plus resampled residuals scaled by noise) for a smoother synthetic baseline.
lambda_nn	Nuclear-norm penalty for the "lowrank" baseline fit.
noise	For "lowrank", standard-deviation multiplier on resampled residuals added back to the fit (0 = noiseless baseline).
seed	Optional RNG seed.

Value

A long data.frame with columns id, t, y, w, y0 (the imposed untreated potential outcome) and tau (the true effect, 0 off treatment).

See Also

sim_panel(), panel_compare(), panel_rmse()

Examples

```
real <- sim_panel(N = 40, T = 18, n_treated = 0L, att = 0, seed = 1)
ss <- sim_semisynthetic(real, "y", "id", "t",
                        n_treated = 6, t0 = 14, att = 3, seed = 2)
mean(ss$tau[ss$w == 1])   # true ATT = 3
```

---

trop *Triply RObust Panel (TROP) estimator*

Description

Fits the TROP estimator of Athey, Imbens, Qu & Viviano (2025) on a long panel. TROP combines a low-rank-plus-two-way-fixed-effects outcome model with exponential-decay unit weights (upweighting controls similar to the treated) and time weights (upweighting periods near the treated periods). Penalties are chosen by leave-one-out cross-validation on the control cells. The estimator nests DID/TWFE, matrix completion and synthetic-control-type weighting as special cases.

Usage

```
trop(
  data,
  outcome,
  treatment,
  unit,
  time,
  lambda = NULL,
  anchor = c("auto", "per_cell", "pooled"),
  se = c("auto", "jackknife", "bootstrap", "placebo", "none"),
  grids = NULL,
  control = trop_control(),
  verbose = FALSE
)
```

Arguments

`data`	A long `data.frame` with one row per unit-time.
`outcome`, `treatment`, `unit`, `time`	Column names (strings). `treatment` must be a 0/1 indicator of *active* treatment in that cell (works for block, staggered, and non-absorbing designs).
`lambda`	Optional named list `list(time=, unit=, nn=)` to fix the penalties and skip cross-validation.
`anchor`	How weights are anchored to treated cells: `"per_cell"` re-solves eq. (2) with cell-specific weights for every treated cell (faithful to the paper); `"pooled"` solves once with weights anchored to the treated set (fast); `"auto"` (default) uses `per_cell` when there are at most `max_cells` treated cells and `pooled` otherwise.
`se`	Standard-error method: `"auto"`, `"jackknife"` (leave-one-treated- unit-out; needs >= 2 treated units), `"placebo"` (assign the treated pattern to controls; for a single treated unit), or `"none"`.
`grids`	Optional list of penalty grids; see `trop_control()`.
`control`	A list of solver/CV settings from `trop_control()`.
`verbose`	Logical; print CV progress.

Value

An object of class `trop`: a list with the ATT `estimate`, `std.error`, `conf.low`/`conf.high`, the selected penalties `lambda`, per-cell effects, the estimated counterfactual matrix, weights, and the reshaped panel.

References

Athey, S., Imbens, G. W., Qu, Z., & Viviano, D. (2025). Triply Robust Panel Estimators. arXiv:2508.21536.

Examples

```
df <- sim_panel(N = 20, T = 12, n_treated = 4, t0 = 9, seed = 1)
fit <- trop(df, "y", "w", "id", "t", se = "none",
            control = trop_control(n_cv_cells = 8L, cv_cycles = 1L))
fit
```

`trop_control` *Control settings for* `trop()`

Description

Control settings for `trop()`

Usage

```
trop_control(
  max_iter = 200L,
  tol = 1e-05,
  n_cv_cells = 120L,
  cv_cycles = 2L,
  max_cells = 60L,
  conf_level = 0.95,
  n_boot = 200L,
  boot_ci = c("percentile", "normal"),
  svd = c("truncated", "full"),
  workers = 1L,
  seed = NULL
)
```

Arguments

`max_iter`	Maximum solver iterations.
`tol`	Solver convergence tolerance.
`n_cv_cells`	Number of control cells sampled for the CV criterion.
`cv_cycles`	Number of coordinate-descent cycles in penalty selection.
`max_cells`	Threshold for `anchor = "auto"` to switch to pooled weights.
`conf_level`	Confidence level for intervals.
`n_boot`	Number of replications for the bootstrap standard error (`se = "bootstrap"`).
`boot_ci`	Bootstrap confidence-interval type: `"percentile"` or `"normal"`.
`svd`	Singular-value decomposition used by the soft-impute solver: `"truncated"` (default) computes only the leading singular triplets with `RSpectra` when it is installed and the matrix is large enough to benefit, falling back to the full SVD otherwise; `"full"` always uses the exact base R `svd()`. The two agree to numerical tolerance; `"truncated"` is faster on large panels, `"full"` is used for exact numerical-agreement checks.

workers	Number of parallel workers for the embarrassingly parallel loops (cross-validation cells, and the bootstrap / jackknife / placebo replicates). 1 (default) runs serially. Values > 1 use future.apply/future when installed (a transient multisession plan is set up and restored automatically); if those packages are missing it falls back to serial with no error. Results are reproducible given seed.
seed	Optional seed for CV cell sampling (reproducibility).

Value

A list of control parameters.

`trop_matrix` *TROP estimate on matrix input*

Description

A thin, matrix-in wrapper around the TROP working model, written independently from the paper. The matrix-in form is convenient for numerically comparing `trop()` against other matrix-based implementations on identical inputs. For data-frame input, cross-validated penalty selection, inference and the multi-estimator comparison, use `trop()` and `panel_compare()` instead.

Usage

```
trop_matrix(
  Y,
  W,
  treated_units,
  lambda_unit,
  lambda_time,
  lambda_nn,
  treated_periods,
  control = trop_control()
)
```

Arguments

Y	N x T outcome matrix.
W	N x T 0/1 treatment matrix (1 = actively treated cell).
treated_units	Integer row indices of the treated units, used to anchor the unit weights.
lambda_unit, lambda_time	Non-negative decay parameters for the unit and time weights.
lambda_nn	Nuclear-norm penalty; use Inf to drop the low-rank term.
treated_periods	Number of final columns treated as the post block, used to build the pre-period mask and the time-distance centre.
control	A list of solver controls from `trop_control()`.

Details

The untreated outcome model is fitted on the control cells with the supplied weights, and the returned effect is the average over treated cells of `Y - alpha - beta - L`, exactly as in the paper's eq. (2). With `lambda_nn = Inf` the low-rank term is dropped and the fit is weighted two-way fixed effects, matching the reference to numerical tolerance. With a finite `lambda_nn` the nuclear-norm term is solved by this package's proximal-gradient routine; because that uses a different parameterisation from the reference's convex solver, finite-penalty results agree in behaviour but not to the last digit.

Value

A single numeric ATT estimate.

References

Athey, S., Imbens, G. W., Qu, Z., & Viviano, D. (2025). *Triply Robust Panel Estimators.* arXiv:2508.21536.

See Also

`trop()`, `panel_compare()`

Examples

```
df <- sim_panel(N = 15, T = 12, n_treated = 3, t0 = 9, att = 2, seed = 1)
Y <- matrix(0, max(df$id), max(df$t)); W <- Y
for (k in seq_len(nrow(df))) { Y[df$id[k], df$t[k]] <- df$y[k]
  W[df$id[k], df$t[k]] <- df$w[k] }
tu <- which(rowSums(W) > 0)
trop_matrix(Y, W, tu, 0.1, 0.1, Inf, treated_periods = 4)
```

`trop_sensitivity` *TROP penalty-sensitivity grid (heatmap data)*

Description

Sweeps the time penalty λ_{time} against the nuclear-norm penalty λ_{nn} (holding the unit penalty fixed) and records, at each grid point, both the resulting ATT estimate and the leave-one-out cross-validation loss. This is the data behind the diagnostic heatmap: colour the cells by CV loss to see where the data-driven choice lands, and read the ATT off each cell to judge how sensitive the estimate is to the penalties.

Usage

```
trop_sensitivity(
  data,
  outcome,
  treatment,
  unit,
```

```
  time,
  lambda_time = NULL,
  lambda_nn = NULL,
  lambda_unit = NULL,
  anchor = "pooled",
  control = trop_control(),
  seed = NULL,
  verbose = FALSE
)
```

Arguments

`data`	A long `data.frame`, one row per unit-time.
`outcome`, `treatment`, `unit`, `time`	Column names (strings).
`lambda_time`, `lambda_nn`	Numeric grids for the two swept penalties. Defaults are derived from the data scale; `lambda_nn` defaults to the finite part of the default grid (the heatmap axis needs finite values).
`lambda_unit`	The fixed unit penalty. If `NULL`, it is chosen once by cross-validation and held fixed across the sweep.
`anchor`	Estimation anchor passed to the ATT computation (`"pooled"` by default for speed).
`control`	A list of solver/CV controls from `trop_control()`.
`seed`	Optional integer seed for CV-cell sampling.
`verbose`	Logical; print progress.

Value

A `cf_trop_grid` (a `data.frame`) with columns `lambda_time`, `lambda_nn`, `lambda_unit`, `att`, `cv_loss`. The grid point minimising `cv_loss` (the data-driven choice) is stored in `attr(., "selected")`.

See Also

`trop()`, `autoplot.cf_trop_grid()`

Examples

```
df <- sim_panel(N = 20, T = 12, n_treated = 4, t0 = 9, att = 2, seed = 1)
g <- trop_sensitivity(df, "y", "w", "id", "t",
                      lambda_time = c(0, 0.1, 0.5), lambda_nn = c(2, 5),
                      control = trop_control(n_cv_cells = 8L, cv_cycles = 1L))
autoplot(g)
```

Index