# Package: ppsynth (via r-universe)

November 9, 2026

**Type** Package

**Title** Partial-Pooling Synthetic Control with Multiple Outcomes

**Version** 0.2.1

**Date** 2026-10-24

**Description** Synthetic control for studies with several outcomes. Each outcome receives its own donor weights, their dispersion around a common centroid is penalized, no single penalty is ever selected (the estimator aggregates over a penalty grid by leave-one-out exponential weighting), and the resulting mass profile reports how much pooling the data support. One call returns the per-outcome effects, the weights, the realized pooling, and whole-pipeline placebo p-values. Implements the estimator of Grier (2026) ``Correlated Weights: Partial Pooling for Synthetic Control with Multiple Outcomes'' <https://papers.ssrn.com/sol3/papers.cfm?abstract_id=7149398>, the common-weights corner of which is the estimator of Sun, Ben-Michael and Feller (2025) <doi:10.1162/rest_a_01592>. The compiled engine and a pure-R reference engine are certified against the Python engine behind the 'Stata' command 'ppsynth'.

**License** MIT + file LICENSE

**URL** https://github.com/angusgrier/PPSYNTH

**BugReports** https://github.com/angusgrier/PPSYNTH/issues

**Encoding** UTF-8

**LazyData** true

**Depends** R (>= 4.1)

**Imports** stats, graphics, parallel, Rcpp (>= 1.0.0)

**LinkingTo** Rcpp

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

**VignetteBuilder** knitr

**Config/testthat/edition** 3

**RoxygenNote** 7.3.1

**NeedsCompilation** yes

**Author** Kevin B. Grier [aut, cre]

**Maintainer** Kevin B. Grier <kevin.grier@ttu.edu>

**Repository** https://angusgrier.r-universe.dev

**Date/Publication** 2026-11-08 19:52:11 UTC

**RemoteUrl** https://github.com/cran/ppsynth

**RemoteRef** HEAD

**RemoteSha** e60c21bd277e2842b5173f75a1aba80a6d1e88ee

# Contents

fit_ppsynth . . . . . 2
FITTABILITY_THRESHOLD . . . . . 4
germany_panel . . . . . 4
LAM_GRID_DEFAULT . . . . . 5
placebo_inference . . . . . 5
plot.ppsynth . . . . . 7
ppsynth . . . . . 7
ppsynth_selftest . . . . . 11
ppsynth_test_panel . . . . . 11
print.ppsynth . . . . . 12

**Index** **13**

---

`fit_ppsynth` *Fit the partial-pooling synthetic control (matrix interface)*

---

## Description

Direct port of `fit_ppsynth()` in `ppsynth_core.py`. Most users will call `ppsynth` on a long panel instead; this function is the engine underneath it and takes the data as matrices.

## Usage

```
fit_ppsynth(
  Y_treated,
  Y_donors,
  T0,
  method = "aggregate",
  lam = NULL,
  lam_grid = NULL,
  holdout = NULL,
  outcome_names = NULL,
  engine = c("cpp", "r"),
```

```
  cores = 1L
)
```

## Arguments

| | |
|---|---|
| `Y_treated` | list of K numeric vectors of length T (the treated unit, one vector per outcome). |
| `Y_donors` | list of K numeric matrices, each T x N0 (the donors, one matrix per outcome, columns in the same donor order for every outcome). |
| `T0` | number of pre-treatment periods (at least 4). |
| `method` | `"aggregate"` (default; leave-one-out exponentially weighted aggregation over the lambda grid) or `"cv"` (leave-future-out selection of a single lambda). |
| `lam` | optional fixed lambda; overrides `method`. Pass `Inf` for the common-weights corner. Snapped to the nearest grid point. |
| `lam_grid` | the lambda grid; default `LAM_GRID_DEFAULT`. |
| `holdout` | number of holdout periods for `method = "cv"`; default `max(2, T0 %/% 4)`. |
| `outcome_names` | optional outcome labels, used only in the caution messages. |
| `engine` | `"cpp"` (default; compiled inner solver) or `"r"` (pure-R reference solver). Same algorithm and settings; see the certification report shipped in `inst/certification`. |
| `cores` | number of cores for the leave-one-out folds of the aggregation step (default 1). Results do not depend on `cores`. |

## Details

Fittability guardrail: any outcome whose pre-RMSPE exceeds `FITTABILITY_THRESHOLD` (0.9) times its own pre-period sd is listed in `poor_fit_outcomes`. Aggregation is not corrected for this (its LOO scoring already down-weights a bad expert), but under `method = "cv"` the `cv_warning` field is populated, since CV's summed leave-future-out loss has no such damping.

Support check: for each outcome, each pre-treatment period where the treated value falls outside the donor min-max band is a period no convex combination of donors can match in levels. The per-outcome share of such periods is returned in `support_outside_share` (period indices in `support_outside_periods`, 1-based), and any violation at all populates `support_warning`. Report-only; the estimator is never modified.

## Value

A list with elements `gamma` (N0 x K weights), `gamma_common` (N0; cross-outcome mean weights), `tau` (K mean post-treatment effects, original scale), `att_path` (list of K gap paths of length T), `method`, `lam` (effective lambda), `lam_grid`, `agg_mass` (aggregation mass over the grid, or `NULL`), `eta`, `cv_curve` (or `NULL`), `gamma_path` (M x K x N0 array of every expert's weights), `pre_rmspe`, `poor_fit_outcomes`, `cv_warning`, `support_outside_share`, `support_outside_periods`, `support_warning`, `realized_pooling`, `pooling_grid`, `pooling_q25`, `pooling_q75`, `pooling_degenerate`.

## See Also

`ppsynth`, `placebo_inference`

## Examples

```
# the shipped test panel as matrices: unit 1 treated from 2010 (T0 = 10)
data(ppsynth_test_panel)
outs <- c("gdp", "health", "invest")
Y_tr <- lapply(outs, function(v) ppsynth_test_panel[[v]][ppsynth_test_panel$id == 1])
Y_dn <- lapply(outs, function(v)
  sapply(2:9, function(u) ppsynth_test_panel[[v]][ppsynth_test_panel$id == u]))
fit <- fit_ppsynth(Y_tr, Y_dn, T0 = 10, outcome_names = outs)
fit$agg_mass
fit$realized_pooling
```

---

# FITTABILITY_THRESHOLD *Fittability threshold*

---

## Description

An outcome whose pre-treatment RMSPE exceeds this multiple of its own pre-period standard deviation is flagged as poorly fit (see Section 5.3 of the working paper).

## Usage

```
FITTABILITY_THRESHOLD
```

## Format

An object of class `numeric` of length 1.

---

# germany_panel *German reunification panel*

---

## Description

The worked example of Section 5A of the working paper: five outcomes for 21 OECD countries, 1970-2004. Germany (`id = 8`) is treated at reunification (first treated year 1990), giving 20 pre-treatment years and 15 post years. The outcomes are four subcomponents of the historical index of economic liberty (property rights, sound money, freedom to trade, regulation) and the government consumption share. Identical to `germany_panel.dta` shipped with the Stata package.

## Usage

```
germany_panel
```

## Format

A data frame with 735 rows and 8 columns: `id`, `country`, `year`, `proprights`, `money`, `inttrade`, `regulation`, `csh_g`.

### Source

https://github.com/angusgrier/PPSYNTH

---

## LAM_GRID_DEFAULT *Default lambda grid*

---

### Description

The ten-point pooling grid used by the aggregation estimator: exact separate fits at 0, exact common weights at `Inf`, and eight interior penalties warm-started along the grid.

### Usage

```
LAM_GRID_DEFAULT
```

### Format

An object of class `numeric` of length 10.

---

## placebo_inference *Whole-pipeline placebo inference (matrix interface)*

---

### Description

Reruns the ENTIRE chosen pipeline (leave-one-out aggregation or CV included) for each donor treated in turn as a placebo, with the actual treated unit returned to the donor pool. Direct port of `placebo_inference()` in `ppsynth_core.py`.

### Usage

```
placebo_inference(
  Y_treated,
  Y_donors,
  T0,
  method = "aggregate",
  lam = NULL,
  lam_grid = NULL,
  holdout = NULL,
  convention = "phipson_smyth",
  outcome_names = NULL,
  verbose = FALSE,
  engine = c("cpp", "r"),
  cores = 1L
)
```

## Arguments

| | |
|---|---|
| Y_treated | list of K numeric vectors of length T (the treated unit, one vector per outcome). |
| Y_donors | list of K numeric matrices, each T x N0 (the donors, one matrix per outcome, columns in the same donor order for every outcome). |
| T0 | number of pre-treatment periods (at least 4). |
| method | "aggregate" (default; leave-one-out exponentially weighted aggregation over the lambda grid) or "cv" (leave-future-out selection of a single lambda). |
| lam | optional fixed lambda; overrides method. Pass Inf for the common-weights corner. Snapped to the nearest grid point. |
| lam_grid | the lambda grid; default LAM_GRID_DEFAULT. |
| holdout | number of holdout periods for method = "cv"; default max(2, T0 %/% 4). |
| convention | "phipson_smyth" (default; p = (1 + #{>=})/(N0+1), always valid, minimum 1/(N0+1)) or "strict" (p = #{>}/(N0+1); synth-style strict exceedance, can be zero, anti-conservative). |
| outcome_names | optional outcome labels, used only in the caution messages. |
| verbose | print a progress line after each placebo fit (ignored when cores > 1). |
| engine | "cpp" (default; compiled inner solver) or "r" (pure-R reference solver). Same algorithm and settings; see the certification report shipped in inst/certification. |
| cores | number of cores: the placebo units are fitted in parallel (each with a single-core leave-one-out step). Results do not depend on cores. |

## Details

Per-outcome p-values rank the treated unit's post/pre RMSPE ratio against the placebo ratios. Joint p-values use the max and mean of the per-outcome ratios studentized over ALL N0+1 units (treated and donors), which preserves exchangeability under the sharp null.

## Value

A list: p_outcome (K), p_joint_max, p_joint_mean, stats_treated (K), stats_placebo (N0 x K), treated_fit (the fit_ppsynth result for the treated unit), placebo_gaps (list over donors of K-lists of gap paths), convention.

## See Also

ppsynth, fit_ppsynth

## Examples

```
data(ppsynth_test_panel)
outs <- c("gdp", "health", "invest")
Y_tr <- lapply(outs, function(v) ppsynth_test_panel[[v]][ppsynth_test_panel$id == 1])
Y_dn <- lapply(outs, function(v)
  sapply(2:9, function(u) ppsynth_test_panel[[v]][ppsynth_test_panel$id == u]))
inf <- placebo_inference(Y_tr, Y_dn, T0 = 10, outcome_names = outs)
inf$p_outcome
inf$p_joint_max
```

`plot.ppsynth` *Plot a ppsynth fit*

## Description

Base-graphics versions of the Stata command's `fig` graphs. Each selected graph is drawn on its own page; wrap the call in `pdf()` or `png()` to save.

## Usage

```
## S3 method for class 'ppsynth'
plot(x, which = c("mass", "cv", "paths", "gaps"), likesynth = x$likesynth, ...)
```

## Arguments

| | |
|---|---|
| `x` | a `"ppsynth"` object. |
| `which` | which graphs to draw: any of `"paths"` (treated vs. synthetic, one panel per outcome), `"gaps"` (treated gap against the placebo donors' gaps; needs `inference = TRUE`), `"mass"` (aggregation mass over the lambda grid; aggregate method) and `"cv"` (leave-future-out loss over the grid; cv method). The default draws every graph that applies, in the Stata order. |
| `likesynth` | logical; draw the treatment line at the first treated period (synth presentation) instead of the last pre-treatment period. Defaults to the value used in the fit. |
| `...` | ignored. |

## Value

`x`, invisibly.

`ppsynth` *Partial-pooling synthetic control with multiple outcomes*

## Description

R port of the Stata command `ppsynth` (and of the Python engine it calls). Estimates synthetic-control weights for two or more outcomes with partial pooling: each outcome gets its own simplex weight vector, shrunk toward a common centroid (the cross-outcome mean weight vector) by a fusion penalty of strength lambda. lambda = 0 reproduces separate per-outcome SCM; lambda = Inf reproduces common (concatenated) weights as in Sun, Ben-Michael and Feller (REStat 2025). Both corners are computed exactly; interior grid values are computed by block-coordinate descent, warm-started along the grid.

## Usage

```
ppsynth(
  data,
  outcomes,
  id,
  time,
  trunit,
  trperiod,
  method = c("aggregate", "cv"),
  lambda = NULL,
  holdout = NULL,
  inference = FALSE,
  convention = c("phipson_smyth", "strict"),
  likesynth = FALSE,
  lam_grid = NULL,
  engine = c("cpp", "r"),
  cores = 1L,
  verbose = TRUE
)
```

## Arguments

| | |
|---|---|
| `data` | a data frame holding a balanced long panel. |
| `outcomes` | character vector of outcome column names (K >= 1; the method is designed for K >= 2). |
| `id` | name of the numeric panel-unit identifier column. |
| `time` | name of the numeric time column. |
| `trunit` | value of `id` identifying the treated unit. |
| `trperiod` | first post-treatment value of `time`. |
| `method` | `"aggregate"` (default) or `"cv"`; see Details. |
| `lambda` | fix the pooling strength instead of tuning it; pass `Inf` for the common-weights corner. Snapped to the nearest grid point. Overrides `method`. |
| `holdout` | holdout periods for `method = "cv"`; default `max(2, T0 %/% 4)`. |
| `inference` | logical; run whole-pipeline placebo inference (reruns the entire pipeline, tuning included, for every donor treated as a placebo). Slow in this pure-R engine: see the README for timings. |
| `convention` | p-value convention, `"phipson_smyth"` (default) or `"strict"`; see Details. |
| `likesynth` | logical; synth/synth_runner presentation: implies `convention = "strict"` and draws the treatment line in plots at the first treated period rather than the last pre-treatment period. |
| `lam_grid` | the lambda grid; default `LAM_GRID_DEFAULT`. |
| `engine` | `"cpp"` (default; compiled inner solver) or `"r"` (pure-R reference solver). Same algorithm and settings; both are certified against the Python engine behind the Stata command. |

| | |
|---|---|
| cores | number of cores to use (default 1): the leave-one-out folds of the aggregation step and, with inference = TRUE, the placebo units are independent and are spread across cores. Results do not depend on cores. |
| verbose | logical; print the inference progress notes. |

## Details

Two tuning methods, selected by method:

"aggregate" **(default)** exponentially weighted average over the lambda-grid "experts", each scored by leave-one-out prediction on the pre-period. The reported weights are the mass-weighted average of expert weights, a convex combination and hence itself on the simplex. This is the paper's primary estimator; it carries a pathwise regret guarantee relative to every expert on the grid, including both corners. The default display reports the final estimator's realized weight pooling and the middle 50% of predictive mass on the realized-pooling scale; print(x, massprofile = TRUE) adds the mass over the grid.

"cv" leave-future-out cross-validation selects a single lambda from the grid. A companion/diagnostic, not the primary estimator: its loss is summed across outcomes with no down-weighting, so a single poorly-fitting outcome can dominate the selection (see the fittability guardrail).

lambda overrides both and fixes lambda directly.

Donor-support check: for each outcome, the number of pre-treatment periods in which the treated unit's value falls outside the range spanned by the donors. Nonnegative weights summing to one cannot reach outside that range. Runs on every fit, fires on any violation, report-only; the caution is printed and returned in support_warning.

Fittability guardrail: if any outcome's pre-RMSPE exceeds 0.9 times its own pre-period standard deviation, that outcome is essentially unfittable from this donor pool. Under method = "cv" a caution naming the outcome(s) is printed and returned in cv_warning.

inference = TRUE reruns the entire chosen pipeline for each donor treated in turn as a placebo, the actual treated unit returned to the donor pool. Per-outcome p-values compare the treated unit's post/pre RMSPE ratio against the placebo distribution; joint p-values use the mean and max of the ratios studentized over all N0+1 units. convention:

"phipson_smyth" **(default)** p = (1 + count of placebo ratios at or above the treated ratio) / (N0+1). Always valid (Phipson & Smyth 2010); minimum attainable value 1/(N0+1).

"strict" p = (count strictly above) / (N0+1), the synth-style strict-exceedance count. Can be exactly zero and is anti-conservative; provided for comparability with legacy synth output.

Per-period placebo p-values (pp_pvals) rank, at each post period, the treated unit's absolute gap standardized by its own pre-RMSPE among the placebos' (descriptive; shown by print(x, effects = TRUE)).

## Value

An object of class "ppsynth": a list with the quantities the Stata command returns in r(), under the same names where possible: gamma (N0 x K donor weights, rows named by donor id), gamma_common, tau, pre_rmspe, effects (T1 x K post-period effects), pp_pvals (with inference), lam_grid, agg_mass, cv_curve, p_outcome, p_joint_max, p_joint_mean, y_treated, y_synth, att_path (all T x K), times, plgaps (list of K T x N0 placebo gap matrices, with inference),

`lambda`, `T0`, `T1`, `N0`, `method`, `outcomes`, `cv_warning`, `support_warning`, `realized_pooling`, `pooling_q25`, `pooling_q75`, `pooling_grid`, `pooling_degenerate`, `convention`, plus `fit` (the raw `fit_ppsynth` result) and `inference` (the raw `placebo_inference` result or `NULL`).

## References

Grier, K. B. (2026). "Correlated Weights: Partial Pooling for Synthetic Control with Multiple Outcomes." Working paper, https://papers.ssrn.com/sol3/papers.cfm?abstract_id=7149398.

Sun, L., Ben-Michael, E. and Feller, A. (2025). "Using Multiple Outcomes to Improve the Synthetic Control Method." Review of Economics and Statistics.

Phipson, B. and Smyth, G. K. (2010). "Permutation p-values should never be zero." Statistical Applications in Genetics and Molecular Biology 9(1).

## See Also

`print.ppsynth`, `plot.ppsynth`, `fit_ppsynth`, `placebo_inference`

## Examples

```
# a small simulated panel shipped with the package (unit 1 treated from 2010)
data(ppsynth_test_panel)
fit <- ppsynth(ppsynth_test_panel, outcomes = c("gdp", "health", "invest"),
               id = "id", time = "year", trunit = 1, trperiod = 2010)
print(fit, massprofile = TRUE)
fit$gamma

# the Stata worked example (Section 5A of the paper): German reunification,
# five outcomes, 20 donors. The aggregate fit takes a few seconds.

data(germany_panel)
outs <- c("proprights", "money", "inttrade", "regulation", "csh_g")
fit <- ppsynth(germany_panel, outcomes = outs, id = "id", time = "year",
               trunit = 8, trperiod = 1990)
print(fit, massprofile = TRUE)
plot(fit, which = "paths")

# Placebo inference reruns the whole pipeline for every donor (a minute or
# two on this panel; `cores` spreads the donors across cores):

inf <- ppsynth(germany_panel, outcomes = outs, id = "id", time = "year",
               trunit = 8, trperiod = 1990, inference = TRUE, cores = 2)
print(inf, effects = TRUE)
plot(inf, which = "gaps")
```

## ppsynth_selftest *Run the ppsynth self-tests*

### Description

Deterministic toy fits that check the estimator's invariants (simplex weights, sane aggregation mass, finite effects, in-sample loss monotone in lambda), the fittability guardrail (aggregation is not flagged; CV carries the caution) and the donor-support check (fires on any violation, counts periods correctly, boundary counts as inside).

### Usage

```
ppsynth_selftest(engine = c("cpp", "r"))
```

### Arguments

| | |
|---|---|
| engine | "cpp" (default) or "r"; the engine to test. |

### Details

The toy designs are built from a fixed internal number sequence, so the function neither sets nor disturbs R's random-number generator.

### Value

TRUE invisibly; stops with a message on the first failure.

### Examples

```
ppsynth_selftest()
```

## ppsynth_test_panel *Small simulated test panel*

### Description

A 9-unit, 16-period simulated panel with three outcomes (gdp, health, invest), shipped with the Stata package as ppsynth_test_panel.dta. Used by the certification tests with unit 1 treated from 2010.

### Usage

```
ppsynth_test_panel
```

### Format

A data frame with 144 rows and 5 columns: id, year, gdp, health, invest.

## Source

https://github.com/angusgrier/PPSYNTH

---

`print.ppsynth` *Print a ppsynth fit*

---

## Description

Reproduces the Stata command's results display.

## Usage

```
## S3 method for class 'ppsynth'
print(x, massprofile = FALSE, effects = FALSE, ...)
```

## Arguments

| | |
|---|---|
| `x` | a `"ppsynth"` object. |
| `massprofile` | logical; also print the aggregation mass profile over the lambda grid (the Stata `massprofile` option). |
| `effects` | logical; also print the per-period effects and, with inference, the per-period placebo p-values (the Stata `effects` option). |
| `...` | ignored. |

## Value

`x`, invisibly.

# Index

∗ **datasets**
FITTABILITY_THRESHOLD, 4
germany_panel, 4
LAM_GRID_DEFAULT, 5
ppsynth_test_panel, 11

fit_ppsynth, 2, *6*, *10*
FITTABILITY_THRESHOLD, 4

germany_panel, 4

LAM_GRID_DEFAULT, *3*, 5, *6*, *8*

placebo_inference, *3*, 5, *10*
plot.ppsynth, 7, *10*
ppsynth, *2*, *3*, *6*, 7
ppsynth_selftest, 11
ppsynth_test_panel, 11
print.ppsynth, *10*, 12