

Package ‘pensynth’

March 28, 2024

Type Package

Title Penalized Synthetic Control Estimation

Version 0.5.1

Description Estimate penalized synthetic control models and perform hold-out validation to determine their penalty parameter. This method is based on the work by Abadie & L'Hour (2021) <[doi:10.1080/01621459.2021.1971535](https://doi.org/10.1080/01621459.2021.1971535)>. Penalized synthetic controls smoothly interpolate between one-to-one matching and the synthetic control method.

License MIT + file LICENSE

URL <https://github.com/vankesteren/pensynth>

Encoding UTF-8

RoxygenNote 7.3.1

Imports clarabel, Matrix

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation no

Author Erik-Jan van Kesteren [cre, aut]
(<<https://orcid.org/0000-0003-1548-1663>>),
Isaac Slaughter [ctb] (<<https://orcid.org/0000-0002-1911-2374>>)

Maintainer Erik-Jan van Kesteren <e.vankesteren1@uu.nl>

Repository CRAN

Date/Publication 2024-03-28 15:10:02 UTC

R topics documented:

| | |
|--------------------------|---|
| cv_pensynth | 2 |
| in_convex_hull | 3 |
| pensynth | 4 |
| placebo_test | 6 |

| | |
|------------------------------|----|
| plot.cvpensynth | 7 |
| plot.pensynthtest | 8 |
| predict.cvpensynth | 8 |
| predict.pensynth | 9 |
| print.cvpensynth | 9 |
| print.pensynth | 10 |
| simulate_data | 10 |

| | |
|--------------|-----------|
| Index | 13 |
|--------------|-----------|

| | |
|-------------|---|
| cv_pensynth | <i>Hold-out validated penalized synthetic control estimator</i> |
|-------------|---|

Description

Compute a penalized synthetic control estimator with hold-out validation for the lambda penalty parameter. Lambda will be determined by minimizing the mean squared error on a hold-out set of pre-intervention outcome time-series.

Usage

```
cv_pensynth(
  X1,
  X0,
  Z1,
  Z0,
  v = 1,
  nlambda = 100,
  opt_pars = clarabel::clarabel_control(),
  standardize = TRUE,
  return_solver_info = FALSE
)
```

Arguments

| | |
|--------------------|--|
| X1 | N_covars by 1 matrix of treated unit covariates |
| X0 | N_covars by N_donors matrix of donor unit covariates |
| Z1 | N_targets by 1 matrix of treated unit hold-out outcome |
| Z0 | N_targets by N_donors matrix of donor unit hold-out outcome |
| v | N_covars vector of variable weights, default 1 |
| nlambda | integer length of lambda sequence (see details) |
| opt_pars | clarabel settings using clarabel::clarabel_control() |
| standardize | boolean whether to standardize the input matrices (default TRUE) |
| return_solver_info | boolean whether to return diagnostic information concerning solver (default FALSE) |

Details

The lambda sequence is an exponentially increasing sequence where The minimum lambda is always $1e-11$, the max lambda is determined by the data.

Value

A list of the lambda sequence, the associated weights, and the mses. If return_solver_info is TRUE, the list will also contain diagnostic information about the solvers.

See Also

[pensynth\(\)](#), [plot.cvpensynth\(\)](#), [placebo_test\(\)](#), [simulate_data\(\)](#)

Examples

```
set.seed(45)
dat <- simulate_data()
res <- with(dat, cv_pensynth(X1, X0, Z1, Z0))
plot(res)
```

in_convex_hull

Check whether treated unit is in the convex hull of donors

Description

This function finds out if the treated unit is in the convex hull of the donor units.

Usage

```
in_convex_hull(X1, X0, ...)
```

Arguments

| | |
|-----|---|
| X1 | N_covars by 1 matrix of treated unit covariates |
| X0 | N_covars by N_donors matrix of donor unit covariates |
| ... | additional arguments passed to clarabel::clarabel() |

Details

This function does not actually construct the convex hull (which is infeasible in higher dimensions), but rather it checks whether the following linear program has a feasible solution:

$$\min q'w \text{ s.t. } Aw = b$$

with w constrained to be above 0 and sum to 1, the feasibility of this linear program directly corresponds to whether the treated is in the convex hull

When the treated unit very close to the boundary of the convex hull this method usually cannot determine this exactly and this function may return NA with the warning "Solver terminated due to lack of progress"

Value

bool whether the treated unit is in the convex hull of the donor units. NA if this cannot be determined.
Vector if X1 has multiple columns.

Examples

```
# create some data
set.seed(45)
X0 <- matrix(runif(20), nrow = 2)
X1 <- matrix(c(.5, .5))

# test if X1 is in the convex hull:
in_convex_hull(X1, X0)

# also works with multiple units in X1:
X1 <- cbind(X1, c(1.3, -3))
in_convex_hull(X1, X0)
```

pensynth

Penalized synthetic control estimator

Description

For a given set of variable weights (v) this function estimates the unit weights for a synthetic control with penalization according to Abadie & L'Hour (2021). This function deals with only a single treated unit.

Usage

```
pensynth(
  X1,
  X0,
  v = 1,
  lambda = 0,
  opt_pars = clarabel::clarabel_control(),
  standardize = TRUE
)
```

Arguments

| | |
|-------------|---|
| X1 | N_covars by 1 matrix of treated unit covariates |
| X0 | N_covars by N_donors matrix of donor unit covariates |
| v | N_covars vector of variable weights (default 1) |
| lambda | numeric penalization parameter |
| opt_pars | clarabel settings using <code>clarabel::clarabel_control()</code> |
| standardize | boolean whether to standardize the input matrices (default TRUE) |

Details

This routine uses the same notation of the original `Synth::synth()` implementation but uses a different, faster quadratic program solver (namely, `clarabel::clarabel()`). Additionally, it implements the penalization procedure described in Abadie & L'Hour (2021), such that the loss function is as in equation 5 of that paper (but for a single treated unit).

Variable weights are not optimized by this function, meaning they need to be pre-specified. This is by design.

The original synthetic control method can be recovered by setting `lambda = 0`. For determining `lambda` based on data, see `cv_pensynth()`.

Value

A list with two values: `w`, the estimated weights; and `solution`, the result of the optimization.

References

Abadie, A., & L'Hour, J. (2021). A penalized synthetic control estimator for disaggregated data. *Journal of the American Statistical Association*, 116(536), 1817-1834.

See Also

`cv_pensynth()`, `placebo_test()`, `simulate_data()`, `Synth::synth()`

Examples

```
# generate some data
X0 <- matrix(
  c(1, 1.3,
    0.5, 1.8,
    1.1, 2.4,
    1.8, 1.8,
    1.3, 1.8), 2)
X1 <- matrix(c(0.8, 1.65), 2)

# run classic synthetic control (no penalization)
res <- pensynth(X1, X0)

# plot donor units in covariate space
plot(t(X0), asp = 1, xlab = "X1", ylab = "X2",
     main = "Covariate space plot")
# add the treated unit
points(t(X1), pch = 2)
# add the synthetic control
points(t(X0%%res$w), pch = 3)

# run synthetic control with penalty
res <- pensynth(X1, X0, lambda = 0.5)
# the resulting synthetic control is
# biased towards its closest neighbours
points(t(X0 %% res$w), pch = 4)
```

placebo_test *Placebo permutation test for pensynth*

Description

Perform a permutation test on a pensynth object, in the sense of Abadie, Diamond, and Hainmueller (2010). The pensynth method is performed multiple times, treating each donor as the treated unit and the treated unit with the remaining donors as the donor units.

Usage

```
placebo_test(object, Y1, Y0)

## S3 method for class 'pensynth'
placebo_test(object, Y1, Y0)

## S3 method for class 'cvpensynth'
placebo_test(object, Y1, Y0)
```

Arguments

| | |
|--------|--|
| object | a fitted pensynth or cvpensynth object |
| Y1 | the post-intervention outcome of the treated unit |
| Y0 | the post-intervention outcome of the donor units (with N_donors columns) |

Details

Note that this function updates the original call in order to re-estimate the synthetic control on the permuted data. Ensure that the data is available to the placebo test function (i.e., avoid complex environment functions such as `with()`), and ensure that the data does not change between estimating the original object and calling this function.

Value

A list with two elements

- E1, the treated unit effect, computed as $Y1 - Y0 \%*\% w$
- E0, the donor unit effects, computed in the same way but using the permutation test's weights.
- ATE1, the estimated ATE of the treated unit
- ATE0, the estimated ATE of the donor units

References

Abadie, A., Diamond, A., & Hainmueller, J. (2010). Synthetic control methods for comparative case studies: Estimating the effect of California's tobacco control program. *Journal of the American statistical Association*, 105(490), 493-505.

See Also

[pensynth\(\)](#), [cv_pensynth\(\)](#), [plot.pensynthtest\(\)](#), [stats::update\(\)](#)

Examples

```
set.seed(45)

# simulate data with an effect of 0.8 SD
dat <- simulate_data(treatment_effect = .8)

# fit a model
fit <- pensynth(dat$X1, dat$X0, lambda = 1e-5)

# Perform placebo test
test <- placebo_test(fit, dat$Y1, dat$Y0)
plot(test)
abline(h = .8, lty = 2)
legend("bottomright", lty = 2, legend = "true effect")

# compute a pseudo p-value based on ATE in
# the post-intervention time period
ref_dist <- stats::ecdf(test$ATE0)
1 - ref_dist(test$ATE1)
```

plot.cvpensynth

Plotting for hold-out validated penalized synthetic control objects

Description

Displays a mean squared error curve and weights curve as a function of lambda, the penalization parameter.

Usage

```
## S3 method for class 'cvpensynth'
plot(x, ...)
```

Arguments

```
x          a cvpensynth output object
...        additional arguments passed to plot()
```

Value

No return value, called for side effects

See Also

[cv_pensynth\(\)](#) [pensynth\(\)](#)

plot.pensynthtest *Plotting a pensynth permutation object*

Description

Plotting the reference distribution and the estimated treatment effect for the treated unit for the pensynth permutation test.

Usage

```
## S3 method for class 'pensynthtest'
plot(x, ...)
```

Arguments

x a pensynthtest object
 ... additional parameters passed to plot

Value

No return value, called for side effects

See Also

[base::plot\(\)](#)

predict.cvpensynth *Create prediction from cvpensynth model*

Description

Matrix multiplies the values in newdata by the unit weights extracted from the cvpensynth object to produce predicted values.

Usage

```
## S3 method for class 'cvpensynth'
predict(object, newdata, lambda, ...)
```

Arguments

object a fitted cvpensynth model
 newdata N_values * N_donors matrix of values for the donor units.
 lambda desired lambda value (defaults to optimal lambda)
 ... ignored

Details

For a chosen lambda that is not in the list of tested lambdas in the cvpensynth object, the closest lambda (on the log scale) will be chosen.

Value

a matrix (column vector) of predicted values

predict.pensynth *Create prediction from pensynth model*

Description

Matrix multiplies the values in newdata by the unit weights extracted from the pensynth object to produce predicted values.

Usage

```
## S3 method for class 'pensynth'
predict(object, newdata, ...)
```

Arguments

| | |
|---------|---|
| object | a fitted cvpensynth model |
| newdata | N_values * N_donors matrix of values for the donor units. |
| ... | ignored |

Value

a matrix (column vector) of predicted values

print.cvpensynth *Print cvpensynth model*

Description

Print cvpensynth model

Usage

```
## S3 method for class 'cvpensynth'
print(x, ...)
```

Arguments

x a cvpensynth object
... ignored

Value

the cvpensynth object, invisibly

| | |
|----------------|-----------------------------|
| print.pensynth | <i>Print pensynth model</i> |
|----------------|-----------------------------|

Description

Print pensynth model

Usage

```
## S3 method for class 'pensynth'  
print(x, ...)
```

Arguments

x a pensynth object
... ignored

Value

the pensynth object, invisibly

| | |
|---------------|--|
| simulate_data | <i>Simulate synthetic control data</i> |
|---------------|--|

Description

This function simulates a basic form of synthetic control data, mainly for testing purposes. This

Usage

```
simulate_data(  
  N_donor = 50,  
  N_covar = 5,  
  N_pre = 12,  
  N_post = 6,  
  N_nonzero = 4,  
  treatment_effect = 1,  
  sd_resid_X1 = 0.1,  
  sd_resid_Z1 = 0.1,  
  sd_resid_Y1 = 0.1  
)
```

Arguments

| | |
|------------------|--|
| N_donor | number of donors |
| N_covar | number of covariates |
| N_pre | number of pre-intervention timepoints |
| N_post | number of post-intervention timepoints |
| N_nonzero | number of true nonzero weights |
| treatment_effect | the size of the true treatment effect |
| sd_resid_X1 | the residual standard deviation of X1 |
| sd_resid_Z1 | the residual standard deviation of Z1 |
| sd_resid_Y1 | the residual standard deviation of Y1 |

Details

Note that treatment effect can be a single number, but it may also be a vector of length N_post, indicating the effect size at each post-intervention measurement occasion.

Value

A list with the following elements

- w the true unit weights
- X0 the donor unit covariates
- X1 the treated unit covariates
- Z0 the donor unit pre-intervention outcomes
- Z1 the treated unit pre-intervention outcomes
- Y0 the donor unit post-intervention outcomes
- Y1 the treated unit post-intervention outcomes

See Also

[pensynth\(\)](#), [cv_pensynth\(\)](#), [placebo_test\(\)](#)

Examples

```
# simulate data with an effect of 0.8 SD
dat <- simulate_data(treatment_effect = 0.8)

plot(
  NA,
  ylim = c(-3, 3),
  xlim = c(1, 18),
  main = "Simulated data",
  ylab = "Outcome value"
)
for (n in 1:ncol(dat$Z0))
  lines(1:18, c(dat$Z0[, n], dat$Y0[, n]), col = "grey")
lines(1:18, c(dat$Z1, dat$Y1))
lines(1:18, rbind(dat$Z0, dat$Y0) %*% dat$w, lty = 2)
abline(v = length(dat$Z1) + 0.5, lty = 3)
legend(
  x = "bottomleft",
  legend = c(
    "Donor units",
    "Treated unit",
    "True synth. control",
    "Intervention time"
  ),
  lty = c(1, 1, 2, 3),
  col = c("grey", "black", "black", "black")
)
```

Index

`base::plot()`, 8

`clarabel::clarabel()`, 3, 5
`clarabel::clarabel_control()`, 2, 4
`cv_pensynth`, 2
`cv_pensynth()`, 5, 7, 11

`in_convex_hull`, 3

`pensynth`, 4
`pensynth()`, 3, 7, 11
`placebo_test`, 6
`placebo_test()`, 3, 5, 11
`plot.cvpensynth`, 7
`plot.cvpensynth()`, 3
`plot.pensynthtest`, 8
`plot.pensynthtest()`, 7
`predict.cvpensynth`, 8
`predict.pensynth`, 9
`print.cvpensynth`, 9
`print.pensynth`, 10

`simulate_data`, 10
`simulate_data()`, 3, 5
`stats::update()`, 7
`Synth::synth()`, 5