

Package ‘Synth’

February 14, 2012

Version 1.1-2

Date 2011-12-26

Title Synthetic Control Group Method for Comparative Case Studies

Author Jens Hainmueller <jhainm@mit.edu> and Alexis Diamond <adiamond@fas.harvard.edu>

Maintainer Jens Hainmueller <jhainm@mit.edu>

Description Implements the synthetic control group method for comparative case studies developed in Abadie and Gardeazabal (2003) and Abadie, Diamond, and Hainmueller (2010). The synthetic control method allows for effect estimation in settings where a single unit (a state, country, firm, etc.) is exposed to an event or intervention. It provides a data-driven procedure to construct synthetic control units based on a weighted combination of comparison units that approximates the characteristics of the unit that is exposed to the intervention. A combination of comparison units often provides a better comparison for the unit exposed to the intervention than any comparison unit alone.

Imports kernlab, optimx

Suggests rgenoud, LowRankQP

License GPL (>= 3)

URL <http://www.mit.edu/~jhainm/software.htm>

Repository CRAN

Date/Publication 2011-12-27 10:06:00

R topics documented:

basque	2
collect.optimx	4
dataprep	5

fn.V	10
gaps.plot	11
path.plot	13
spec.pred.func	14
synth	15
synth.data	20
synth.tab	21

Index	23
--------------	-----------

basque	<i>Panel Data from Spanish Regions to demonstrate the use of the Synthetic Control Method</i>
--------	---

Description

The dataset contains information from 1955–1997 on 17 Spanish regions. It was used by Abadie and Gardeazabal (2003), which studied the economic effects of conflict, using the terrorist conflict in the Basque Country as a case study. This paper used a combination of other Spanish regions to construct a synthetic control region resembling many relevant economic characteristics of the Basque Country before the onset of political terrorism in the 1970s. The data contains per-capita GDP (the outcome variable), as well as population density, sectoral production, investment, and human capital (the predictor variables) for the relevant years, and is used here to demonstrate the implementation of the synthetic control method with the synth library.

Usage

basque

Format

A panel dataframe made up of 18 units: 1 treated (no 17; the Basque country) and 16 control regions (no. 2-16,18). Region no. 1 is the average for the whole country of Spain. 1 outcome variable (gdpcap). 13 predictor variables (6 sectoral production shares, 6 highest educational attainment categories, population density, and the investment rate). Region names and numbers are stored in regionno and regionname. 42 time periods (1955 - 1997). All columns have self-explanatory column names. For reference the variables are:

- regionno
: Region Number.
- regionname
: Region Name.
- year
: Year.
- gdpcap
: real GDP per capita (in 1986 USD, thousands).

- `sec.agriculture`
: production in agriculture, forestry, and fishing sector as a percentage of total production.
- `sec.energy`
: production in energy and water sector as a percentage of total production.
- `sec.industry`
: production in industrial sector as a percentage of total production.
- `sec.construction`
: production in construction and engineering sector as a percentage of total production.
- `sec.energy`
: production in marketable services sector as a percentage of total production.
- `sec.energy`
: production in Nonmarketable services sector as a percentage of total production.
- `school.illit`
: number of illiterate persons.
- `school.prim`
: number of persons with primary education or without studies.
- `school.med`
: number of persons with some high school education.
- `school.high`
: number of persons with high school degree.
- `school.post.high`
: number of persons with tertiary education.
- `popdens`
: population density (persons per square kilometer).
- `invest`
: gross total investment as a share of GDP.

Source

Abadie, A. and Gardeazabal, J. (2003) Economic Costs of Conflict: A Case Study of the Basque Country *American Economic Review* 93 (1) 113–132.

Abadie, A., Diamond, A., Hainmueller, J. (2011). Synth: An R Package for Synthetic Control Methods in Comparative Case Studies. *Journal of Statistical Software* 42 (13) 1–17.

collect.optimx	<i>Collect results from optimx optimization methods</i>
----------------	---

Description

An internal function that collects the results from the different optimization methods run by `optimx`. It stores the parameter and function values and extracts the results for the best performing method (minimum or maximum).

Usage

```
collect.optimx(res, opt = "min")
```

Arguments

<code>res</code>	Output from a call to <code>optimx()</code> .
<code>opt</code>	Either "min" or "max" to extract results for the methods that obtained the minimum or maximum function value across the methods.

Value

<code>out.list</code>	Dataframe with parameter and function values from the different methods.
<code>par</code>	Parameter values from method that attained minimum/maximum across the methods.
<code>value</code>	Function value from method that attained minimum/maximum across the methods.

Author(s)

Jens Hainmueller

See Also

Also see [optimx](#).

dataprep	<i>Constructs a list of matrices from panel dataset to be loaded into synth()</i>
----------	---

Description

The `synth` function takes a standard panel dataset and produces a list of data objects necessary for running `synth` and other Synth package functions to construct synthetic control groups according to the methods outlined in Abadie and Gardeazabal (2003) and Abadie, Diamond, Hainmueller (2010) (see references and example).

User supplies a dataframe ("foo"), chooses predictors, special predictors (explained below), the operators that act upon these predictors, the dependent variable, identifies the columns associated with unit numbers, time periods (and unit names, when available), as well as the treated unit, the control units, the time-period over which to select the predictors, the time-period over which to optimize, and the time-period over which outcome data should be plotted.

The output of `dataprep` contains a list of matrices. This list object can be directly loaded into `synth`.

Usage

```
dataprep(foo = NULL, predictors = NULL,
         predictors.op = "mean", special.predictors = NULL,
         dependent = NULL, unit.variable = NULL,
         time.variable = NULL, treatment.identifier = NULL,
         controls.identifier = NULL, time.predictors.prior = NULL,
         time.optimize.ssr = NULL, time.plot = time.optimize.ssr,
         unit.names.variable = NA)
```

Arguments

<code>foo</code>	The dataframe with the panel data.
<code>predictors</code>	A vector of column numbers or column-name character strings that identifies the predictors' columns. All predictors have to be numeric.
<code>predictors.op</code>	A character string identifying the method (operator) to be used on the predictors. Default is "mean". <code>rm.na = T</code> is hardwired into the code. See <i>*Details*</i> .
<code>special.predictors</code>	A list object identifying additional numeric predictors and their associated pre-treatment years and operators (analogous to "predictors.op" above). See <i>*Details*</i> .
<code>dependent</code>	A scalar identifying the column number or column-name character string that corresponds to the numeric dependent (outcome) variable.
<code>unit.variable</code>	A scalar identifying the column number or column-name character string associated unit numbers. The unit.variable has to be numeric.
<code>time.variable</code>	A scalar identifying column number or column-name character string associated with period (time) data. The time variable has to be numeric.

<code>treatment.identifier</code>	A scalar identifying the “unit.variable” number or a character string giving the “unit.name ”of the treated unit. If a character is supplied, a <code>unit.names.variable</code> also has to be supplied to identify the treated unit.
<code>controls.identifier</code>	A scalar identifying the “unit.variable” numbers or a vector of character strings giving the “unit.name”s of control units. If a character is supplied, a <code>unit.names.variable</code> also has to be supplied to identify the control units unit.
<code>time.predictors.prior</code>	A numeric vector identifying the pretreatment periods over which the values for the outcome predictors should be averaged.
<code>time.optimize.ssr</code>	A numeric vector identifying the periods of the dependent variable over which the loss function should be minimized (i.e. the periods over which mean squared prediction error (MSPE) , that is the sum of squared residuals between treated and the synthetic control unit, are minimized.
<code>time.plot</code>	A vector identifying the periods over which results are to be plotted with <code>gaps.plot</code> and <code>path.plot</code> .
<code>unit.names.variable</code>	A scalar or column-name character string identifying the column with the names of the units. This variable has to be of mode character.

Details

The `predictors.op` argument is a character string that provides a function (eg., "mean", "median", etc.) identifying the name of the operator to be applied to the predictors over the given time period.

The `special.predictors` argument is a list object that contains one or more lists of length = 3. The required components of each of these lists are:

(a) scalar column number associated with that predictor (b) vector of time-period number(s) desired (eg., 1998:2003) (c) character-string identifying the name of the operation to be applied (ie., "mean", "median", etc.)

eg., `special.predictors <- list(listc(x1, 1990:2000, "mean"), listc(x2, 1980:1983, "median"), listc(x3, 1980, "mean"))`

indicates that predictor x1, should be used with its values averaged over periods 1990:2000; predictor x2 should be used with its median values over periods 1980:1983; x3 should be used with the values from period 1980 only.

Value

X1	matrix of treated predictor data. nrows = number of predictors and (possibly) special predictors. ncols = one.
X0	matrix of controls' predictor data. nrows = number of predictors and (possibly) special predictors. ncols = number of control units.
Z1	matrix of treated outcome data for the pre-treatment periods over which MSPE is to be minimized. nrows = number of pre-treatment periods. ncols = one.

Z0	matrix of controls' outcome data for the pre-treatment periods over which MSPE is to be minimized. nrows = number of pre-treatment periods. ncols = number of control units.
Y1plot	matrix of outcome data for treated unit to be used for results plotting. nrows = number of periods. ncols = one.
Y0plot	matrix of outcome data for control units to be used for results plotting. nrows = number of periods. ncols = number of control units.
names.and.numbers	dataframe with two columns showing all unit numbers and corresponding unit names.
tag	a list of all arguments in initial function call.

Author(s)

Jens Hainmueller and Alexis Diamond

References

Abadie, A. and Gardeazabal, J. (2003) Economic Costs of Conflict: A Case Study of the Basque Country *American Economic Review* 93 (1) 113–132.

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.

Abadie, A., Diamond, A., Hainmueller, J. (2011). Synth: An R Package for Synthetic Control Methods in Comparative Case Studies. *Journal of Statistical Software* 42 (13) 1–17.

See Also

[synth](#), [gaps.plot](#), [path.plot](#), [synth.tab](#)

Examples

```
data(synth.data)
```

```
dataprep.out <-
  dataprep(
    foo = synth.data,
    predictors = c("X1", "X2", "X3"),
    predictors.op = "mean",
    dependent = "Y",
    unit.variable = "unit.num",
    time.variable = "year",
    special.predictors = list(
      list("Y", 1991, "mean"),
      list("Y", 1985, "mean"),
      list("Y", 1980, "mean")
    ),
    treatment.identifier = 7,
    controls.identifier = c(29, 2, 13, 17, 32, 38),
```

```

        time.predictors.prior = c(1984:1989),
        time.optimize.ssr = c(1984:1990),
        unit.names.variable = "name",
        time.plot = 1984:1996
    )

## then we run the synth command to identify optimal
## weights-- the optimal way to aggregate information about all the
## control units in such a way as to create the best possible synthetic
## control unit for the treated.
synth.out <- synth(dataprep.out)

## then we want to summarize the results, both numerically and graphically
## there are two ways to summarize the results
## we can either access the output from synth.out directly
## eg.
round(synth.out$solution.w,2)
# contains the unit weights or
synth.out$solution.v
## contains the predictor weights.

## the output from synth opt can be flexibly combined with
## the output from dataprep to compute other quantities of interest
## for example, the period by period discrepancies between the
## treated unit and its synthetic control unit
## can be computed by typing
gaps<- dataprep.out$Y1plot-(
        dataprep.out$Y0plot*%synth.out$solution.w
    ) ; gaps

## the Synth package also offers three convenience functions to summarize results.
## to get summary tables for all information
## (V and W weights plus balance btw. treated and synthetic control) use the
## synth.tab() command
synth.tables <- synth.tab(dataprep.res = dataprep.out,synth.res = synth.out)
print(synth.tables)

## to get summary plots for outcome trajectories
## of the treated and the synthetic control unit use the
## path.plot() and the gaps.plot() commands

## plot in levels (treated and synthetic)
path.plot(dataprep.res = dataprep.out,synth.res = synth.out)

## plot the gaps (treated - synthetic)
gaps.plot(dataprep.res = dataprep.out,synth.res = synth.out)

## Second example: The economic impact of terrorism in the
## Basque country using data from Abadie and Gardeazabal (2003)
## see JSS paper in the references details
data(basque)

```

```

# dataprep: prepare data for synth
dataprep.out <-
  dataprep(
    foo = basque
    ,predictors = c(
      "school.illit", "school.prim", "school.med", "school.high", "school.post.high"
      , "invest"
    )
    ,predictors.op = c("mean")
    ,dependent = c("gdpcap")
    ,unit.variable = c("regionno")
    ,time.variable = c("year")
    ,special.predictors = list(
      list("gdpcap", 1960:1969, c("mean")),
      list("sec.agriculture", seq(1961,1969,2), c("mean")),
      list("sec.energy", seq(1961,1969,2), c("mean")),
      list("sec.industry", seq(1961,1969,2), c("mean")),
      list("sec.construction", seq(1961,1969,2), c("mean")),
      list("sec.services.venta", seq(1961,1969,2), c("mean")),
      list("sec.services.nonventa", seq(1961,1969,2), c("mean")),
      list("popdens", 1969, c("mean"))
    )
    ,treatment.identifier = 17
    ,controls.identifier = c(2:16,18)
    ,time.predictors.prior = c(1964:1969)
    ,time.optimize.ssr = c(1960:1969)
    ,unit.names.variable = c("regionname")
    ,time.plot = c(1955:1997)
  )

# 1. combine highest and second highest schooling category and eliminate highest category
dataprep.out$X1["school.high",] <- dataprep.out$X1["school.high",] + dataprep.out$X1["school.post.high",]
dataprep.out$X1
  <- as.matrix(dataprep.out$X1[-which(rownames(dataprep.out$X1)=="school.post.high"),])
dataprep.out$X0["school.high",] <- dataprep.out$X0["school.high",] + dataprep.out$X0["school.post.high",]
dataprep.out$X0
  <- dataprep.out$X0[-which(rownames(dataprep.out$X0)=="school.post.high"),]

# 2. make total and compute shares for the schooling categories
lowest <- which(rownames(dataprep.out$X0)=="school.illit")
highest <- which(rownames(dataprep.out$X0)=="school.high")

dataprep.out$X1[lowest:highest,] <- (100 * dataprep.out$X1[lowest:highest,]) / sum(dataprep.out$X1[lowest:highest,])
dataprep.out$X0[lowest:highest,] <- 100 * scale(
  dataprep.out$X0[lowest:highest,],
  center=FALSE,
  scale=colSums(dataprep.out$X0[lowest:highest,])
)

# run synth
synth.out <- synth(data.prep.obj = dataprep.out)

# Get result tables
synth.tables <- synth.tab(

```

```

        dataprep.res = dataprep.out,
        synth.res = synth.out
    )

# results tables:
print(synth.tables)

# plot results:
# path
path.plot(synth.res = synth.out,
          dataprep.res = dataprep.out,
          Ylab = c("real per-capita GDP (1986 USD, thousand)"),
          Xlab = c("year"),
          Ylim = c(0,13),
          Legend = c("Basque country","synthetic Basque country"),
          )

## gaps
gaps.plot(synth.res = synth.out,
          dataprep.res = dataprep.out,
          Ylab = c("gap in real per-capita GDP (1986 USD, thousand)"),
          Xlab = c("year"),
          Ylim = c(-1.5,1.5),
          )

```

fn.V

Loss Function for nested optimization of W and V weights

Description

Loss function for the nested optimization of W and V weights used for constructing synthetic control groups according to the methods outlined in Abadie and Gardeazabal (2003) and Abadie, Diamond, Hainmueller (2010) (see references). This function is called by `synth` internally, and should not be called manually by a normal user.

Usage

```
fn.V(variables.v = stop("variables.v missing"), X0.scaled = stop("X0.scaled missing"), X1.scaled = stop("X1.scaled missing"))
```

Arguments

<code>variables.v</code>	1 by k a vector of v weights.
<code>X0.scaled</code>	matrix of controls' predictor data. <code>nrows</code> = number of predictors and (possibly) special predictors. <code>ncols</code> = number of control units.
<code>X1.scaled</code>	matrix of treated predictor data. <code>nrows</code> = number of predictors and (possibly) special predictors. <code>ncols</code> = one.

Z0	matrix of controls' outcome data for the pre-treatment periods over which MSPE is to be minimized. nrows = number of pre-treatment periods. ncols = number of control units.
Z1	matrix of treated outcome data for the pre-treatment periods over which MSPE is to be minimized. nrows = number of pre-treatment periods. ncols = one.
margin.ipop	setting for ipop optimization routine: how close we get to the constrains (see ipop for details)
sigf.ipop	setting for ipop optimization routine: Precision (default: 7 significant figures (see ipop for details)
bound.ipop	setting for ipop optimization routine: Clipping bound for the variables (see ipop for details)
quadopt	string vector that specifies the routine for quadratic optimization over w weights. possible values are "ipop" and "LowRankQP" (see ipop and LowRankQP LowRankQP for details)

Value

A scalar that contains the function value.

Author(s)

Jens Hainmueller and Alexis Diamond

References

- Abadie, A. and Gardeazabal, J. (2003) Economic Costs of Conflict: A Case Study of the Basque Country *American Economic Review* 93 (1) 113–132.
- 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.
- Abadie, A., Diamond, A., Hainmueller, J. (2011). Synth: An R Package for Synthetic Control Methods in Comparative Case Studies. *Journal of Statistical Software* 42 (13) 1–17.

See Also

[synth](#), [dataprep](#), [gaps.plot](#), [path.plot](#), [synth.tab](#)

gaps.plot	<i>Plots Gap in Outcome Trajectories between the Treated its Synthetic Control Unit</i>
-----------	---

Description

This function plots the gaps in the trajectories of the outcome variable for the treated unit and the synthetic control group constructed by [synth](#) and [dataprep](#). The user can specify whether the whole time period or only the pre-treatment period should be plotted.

Usage

```
gaps.plot(synth.res = NA,
          dataprep.res = NA,
          Ylab = c("Title"),
          Xlab = c("Time"),
          Main = c("Gaps: Treated - Synthetic"),
          tr.intake = NA,
          Ylim = NA,
          Z.plot = FALSE)
```

Arguments

synth.res	Output list created by synth .
dataprep.res	Output list created by dataprep .
tr.intake	Optional scalar to indicate the time of treatment intake with a vertical line.
Ylab	Optional label for Y axis.
Xlab	Optional label for X axis.
Ylim	Optional Ylim.
Main	Optional main title.
Z.plot	Flag. If true, only pretreatment period is plotted.

Details

The trajectory of the outcome for the synthetic control group is calculated as: `dataprep.res$Y0plot` `%%` `synth.res$solution.w`. You can use this calculation to construct custom made plots.

Value

The plot of trajectories.

Author(s)

Jens Hainmueller and Alexis Diamond

References

Abadie, A. and Gardeazabal, J. (2003) Economic Costs of Conflict: A Case Study of the Basque Country *American Economic Review* 93 (1) 113–132

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.

Abadie, A., Diamond, A., Hainmueller, J. (2011). Synth: An R Package for Synthetic Control Methods in Comparative Case Studies. *Journal of Statistical Software* 42 (13) 1–17.

See Also

[dataprep](#), [synth](#), [path.plot](#), [synth.tab](#)

path.plot	<i>Plots Outcome Trajectories for Treated Unit and its Synthetic Control Unit</i>
-----------	---

Description

This function plots the trajectories of the outcome variable for the treated unit and the synthetic control group constructed by [synth](#) and [dataprep](#). The user can specify whether the whole time period or only the pretreatment period should be plotted.

Usage

```
path.plot(synth.res = NA, dataprep.res = NA, tr.intake = NA, Ylab = c("Y Axis"), Xlab = c("Time"), Ylim = NA,
```

Arguments

synth.res	Output list created by synth .
dataprep.res	Output list created by dataprep .
tr.intake	Optional scalar to indicate the time of treatment intake with a vertical line.
Ylab	Optional label for Y axis.
Xlab	Optional label for X axis.
Ylim	Optional Ylim.
Main	Optional main title.
Legend	Optional legend text (e.g. c("Treated", "Synthetic")); see <code>?legend</code> for details.
Legend.position	Optional legend position (e.g. "bottomright"); see <code>?legend</code> for details.
Z.plot	Flag. If true, only pretreatment period is plotted.

Details

The trajectory of the outcome for the synthetic control group is calculated as: `dataprep.res$Y0plot%*%synth.res$solution.w`. You can use this calculation to construct custom made plots.

Value

The plot of trajectories.

Author(s)

Jens Hainmueller and Alexis Diamond

References

Abadie, A. and Gardeazabal, J. (2003) Economic Costs of Conflict: A Case Study of the Basque Country *American Economic Review* 93 (1) 113–132.

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.

Abadie, A., Diamond, A., Hainmueller, J. (2011). Synth: An R Package for Synthetic Control Methods in Comparative Case Studies. *Journal of Statistical Software* 42 (13) 1–17.

See Also

[dataprep](#), [gaps.plot](#), [synth](#), [synth.tab](#)

spec.pred.func

Special Predictor Function for Dataprep

Description

This function is called by [dataprep](#) to handle special predictors in the process of setting up the dataset to be loaded into [synth](#). It should not be called manually by the normal user.

Usage

```
spec.pred.func(list.object = NULL,  
              tr.numb = NULL,  
              co.numb = NULL,  
              unit.var = NULL,  
              time.var = NULL,  
              foo.object = NULL,  
              X0.inner = NULL,  
              X1.inner = NULL)
```

Arguments

list.object	NA
tr.numb	NA
co.numb	NA
unit.var	NA
time.var	NA
foo.object	NA
X0.inner	NA
X1.inner	NA

Details

NA

Value

NA

Author(s)

Jens Hainmueller and Alexis Diamond

References

Abadie, A. and Gardeazabal, J. (2003) Economic Costs of Conflict: A Case Study of the Basque Country *American Economic Review* 93 (1) 113–132.

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.

Abadie, A., Diamond, A., Hainmueller, J. (2011). Synth: An R Package for Synthetic Control Methods in Comparative Case Studies. *Journal of Statistical Software* 42 (13) 1–17.

See Also

[synth](#), [dataprep](#), [gaps.plot](#), [path.plot](#), [synth.tab](#)

synth

Constructs synthetic control units for comparative case studies

Description

Implements the synthetic control method for causal inference in comparative case studies as developed in Abadie and Gardeazabal (2003) and Abadie, Diamond, Hainmueller (2010). [synth](#) estimates the effect of an intervention by comparing the evolution of an aggregate outcome for a unit affected by the intervention to the evolution of the same aggregate outcome for a synthetic control group.

[synth](#) constructs this synthetic control group by searching for a weighted combination of control units chosen to approximate the unit affected by the intervention in terms of characteristics that are predictive of the outcome. The evolution of the outcome for the resulting synthetic control group is an estimate of the counterfactual of what would have been observed for the affected unit in the absence of the intervention.

[synth](#) can also be used to conduct a variety of placebo and permutation tests that produce informative inference regardless of the number of available comparison units and the number of available time-periods. See Abadie and Gardeazabal (2003), Abadie, Diamond, and Hainmueller (2010, 2011) for details.

[synth](#) requires the user to supply four matrices as its main arguments. These matrices are named X0, X1, Z1, and Z0 accordingly. X1 and X0 contain the predictor values for the treated unit and the

control units respectively. `Z1` and `Z0` contain the outcome variable for the pre-intervention period for the treated unit and the control unit respectively. The pre-intervention period refers to the time period prior to the intervention, over which the mean squared prediction error (MSPE) should be minimized. The MSPE refers to the squared deviations between the outcome for the treated unit and the synthetic control unit summed over all pre-intervention periods specified in `Z1` and `Z0`.

Creating the matrices `X1`, `X0`, `Z1`, and `Z0` from a (panel) dataset can be tedious. Therefore the Synth library offers a preparatory function called `dataprep` that allows the user to easily create all inputs required for `synth`. By first calling `dataprep` the user creates a single list object called `data.prep.obj` that contains all essential data elements to run `synth`.

Accordingly, a usual sequence of commands to implement the synthetic control method is to first call `dataprep` to prepare the data to be loaded into `synth`. Then `synth` is called to construct the synthetic control group. Finally, results are summarized using the functions `synth.tab`, `path.plot`, or `gaps.plot`.

An example of this sequence is provided in the documentation to `dataprep`. This procedure is strongly recommended. Alternatively, the user may provide his own preprocessed data matrices and load them into `synth` via the `X0`, `X1`, `Z1`, and `Z0` arguments. In this case, no `data.prep.obj` should be specified.

The output from `synth` is a list object that contains the weights on predictors (`solution.V`) and weights on control units (`solution.W`) that define contributions to the synthetic control unit.

Usage

```
synth(data.prep.obj = NULL, X1 = NULL, X0 = NULL, Z0 = NULL, Z1 = NULL, custom.v = NULL, optimxmethod = c("Nelder-Mead", "BFGS"))
```

Arguments

<code>data.prep.obj</code>	the object that comes from running <code>dataprep</code> . This object contains all information about <code>X0</code> , <code>X1</code> , <code>Z1</code> , and <code>Z0</code> . Therefore, if <code>data.prep.obj</code> is supplied, none of <code>X0</code> , <code>X1</code> , <code>Z1</code> , and <code>Z0</code> should be manually specified!
<code>X1</code>	matrix of treated predictor data, <code>nrows</code> = number of predictors <code>ncols</code> = ones.
<code>X0</code>	matrix of controls' predictor data. <code>nrows</code> = number of predictors. <code>ncols</code> = number of control units (≥ 2).
<code>Z1</code>	matrix of treated outcome data for the pre-treatment periods over which MSPE is to be minimized. <code>nrows</code> = number of pre-treatment periods. <code>ncols</code> = 1.
<code>Z0</code>	matrix of controls' outcome data for the pre-treatment periods over which MSPE is to be minimized. <code>nrows</code> = number of pre-treatment periods. <code>ncols</code> = number of control units.
<code>custom.v</code>	vector of weights for predictors supplied by the user. uses <code>synth</code> to bypass optimization for <code>solution.V</code> . See details.
<code>optimxmethod</code>	string vector that specifies the optimization algorithms to be used. Permissible values are all optimization algorithms that are currently implemented in the <code>optimx</code> function (see this function for details). This list currently includes <code>c("Nelder-Mead", "BFGS", "CG", "L-BFGS-B", "nlm", "nlminb", "spg", and "ucminf")</code> . If multiple algorithms are specified, <code>synth</code> will run the optimization with all chosen algorithms and then return the result for the best performing method. Default is <code>c("Nelder-Mead", "BFGS")</code> . As an additional possibility, the user can

also specify 'All' which means that `synth` will run the results over all algorithms in `optimx`.

<code>genoud</code>	Logical flag. If true, <code>synth</code> embarks on a two step optimization. In the first step, <code>genoud</code> , an optimization function that combines evolutionary algorithm methods with a derivative-based (quasi-Newton) method to solve difficult optimization problems, is used to obtain a solution. See <code>genoud</code> for details. In the second step, the <code>genoud</code> results are passed to the optimization algorithm(s) chosen in <code>optimxmethod</code> for a local optimization within the neighborhood of the <code>genoud</code> solution. This two step optimization procedure will require much more computing time, but may yield lower loss in cases where the search space is highly irregular.
<code>quadopt</code>	string vector that specifies the routine for quadratic optimization over <code>w</code> weights. possible values are "ipop" and "LowRankQP" (see <code>ipop</code> and <code>LowRankQP</code> for details). default is 'ipop'
<code>Margin.ipop</code>	setting for ipop optimization routine: how close we get to the constrains (see <code>ipop</code> for details)
<code>Sigf.ipop</code>	setting for ipop optimization routine: Precision (default: 7 significant figures (see <code>ipop</code> for details)
<code>Bound.ipop</code>	setting for ipop optimization routine: Clipping bound for the variables (see <code>ipop</code> for details)
<code>verbose</code>	Logical flag. If TRUE then intermediate results will be shown.
<code>...</code>	Additional arguments to be passed to <code>optimx</code> and or <code>genoud</code> to adjust optimization.

Details

As proposed in Abadie and Gardeazabal (2003) and Abadie, Diamond, Hainmueller (2010), the `synth` function routinely searches for the set of weights that generate the best fitting convex combination of the control units. In other words, the predictor weight matrix V is chosen among all positive definite diagonal matrices such that MSPE is minimized for the pre-intervention period.

Instead of using this data-driven procedures to search for the best fitting synthetic control group, the user may supply his own vector of V weights, based on his subjective assessment of the predictive power of the variables in $X1$ and $X0$. In this case, the vector of V weights for each variable should be supplied via the `custom.V` option in `synth` and the optimization over the V matrices is bypassed.

Value

<code>solution.v</code>	vector of predictor weights.
<code>solution.w</code>	vector of weights across the controls.
<code>loss.v</code>	MSPE from optimization over <code>v</code> and <code>w</code> weights.
<code>loss.w</code>	Loss from optimization over <code>w</code> weights.
<code>custom.v</code>	if this argument was specified in the call to <code>synth</code> , this outputs the weight vector specified.
<code>rgV.optim</code>	Results from <code>optimx()</code> minimization. Could be used for diagnostics.

Author(s)

Jens Hainmueller and Alexis Diamond

References

Abadie, A. and Gardeazabal, J. (2003) Economic Costs of Conflict: A Case Study of the Basque Country *American Economic Review* 93 (1) 113–132.

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.

Abadie, A., Diamond, A., Hainmueller, J. (2011). Synth: An R Package for Synthetic Control Methods in Comparative Case Studies. *Journal of Statistical Software* 42 (13) 1–17.

See Also

[dataprep](#), [gaps.plot](#), [path.plot](#), [synth.tab](#)

Examples

```
## While synth() can be used to construct synthetic control groups
## directly, by providing a X1,X0,Z1, and Z0 matrix. We strongly
## recommend to first run dataprep() to setup the data.
## Two extensive examples illustrating the whole sequence of
## of commands with:
## 1. dataprep() for matrix-extraction
## 2. synth() for the construction of the synthetic control group
## 3. synth.tab(), gaps.plot(), and path.plot() to summarize the results
## are provided in the examples for the dataprep() function
## please take a close look there.
```

```
## Here we provide a stand-alone example, for synth()
## that does not require the user to run dataprep first.
## This example is a generic version of the original estimation
## of economic impact of terrorism in the Basque country
## presented in Abadie and Gardeazabal (2003).
## See the paper for details and the example in dataprep() for the exact replication.
## Here we just use a subset of the predictors.
## Notice the various matrix extractions below;
## these are all taken care of by running dataprep() first.
```

```
# load data
data(basque)
```

```
# Construct matrix of predictors
predictors = c(
  "school.illit", "school.prim", "school.med", "school.high",
  "invest", "gdpcap", "popdens"
)
```

```
# for the treated unit (Basque country)
# (predictors are averaged for the 1964:1969 period)
```

```

X1 <-
  as.matrix(apply(
    basque[
      basque$regionname == "Basque Country (Pais Vasco)" &
      is.element(basque$year,1964:1969),
      predictors
    ]
    ,2,mean,na.rm=TRUE))

# and the control units (other Spanish regions)
controls <- c(2:16,18)
X0 <- basque[
  is.element(basque$regionno,controls) &
  is.element(basque$year,1964:1969),
  c(predictors,"regionno")
]
X0 <- split(X0, X0[,dim(X0)[2]])
X0 <- sapply(X0, apply, 2, mean, na.rm = TRUE)
X0 <- as.matrix(X0[-dim(X0)[1],])

# get matrix of pre-intervention values of the outcome variable (GDP)
# over which mean squared prediction error should be minimized
# treated unit
Z1 <-
  matrix(basque$gdpcap[
    is.element(basque$year,1960:1969) &
    basque$regionname == "Basque Country (Pais Vasco)"
  ])

Z0 <-
  matrix(basque$gdpcap[
    is.element(basque$year,1960:1969) &
    is.element(basque$regionno,controls)
  ],ncol=length(controls))

rownames(Z0) <- rownames(Z1) <- 1960:1969
colnames(Z0) <- colnames(X0) <- controls

# now construct a synthetic basque country
synth.out <- synth(X1=X1,X0=X0,Z1=Z1,Z0=Z0)

# examine the weights associated with each of the control units
# with region_numbers
round(synth.out$solution.w,3)
# or regionnames
data.frame(round(synth.out$solution.w,3),row.names = unique(basque$regionname)[c(-1,-17)])

# compare the predictor values for the treated unit and its synthetic counterpart
tab <- cbind(X1,X0*%synth.out$solution.w)
colnames(tab) <- c("Basque Country","Synthetic Basque country")
tab

# plot the pre-terrorism trajectory for the treated unit and its synthetic counterpart

```

```

# over with the mean squared prediction error was minimized
matplot(cbind(1960:1969,1960:1969),
        cbind(Z1,Z0*%as.matrix(synth.out$solution.w)),type="l",xlab="year",ylab="GDPcap")
legend("topleft",legend=c("Basque","Synthetic Basque"),col=c("black","red"),lty=c(1,2))

# plot the whole pre and post terrorism period
Y1 <-
  matrix(basque$gdpcap[
    is.element(basque$year,1955:1997) &
    basque$regionname == "Basque Country (Pais Vasco)"
  ])

Y0 <-
  matrix(basque$gdpcap[
    is.element(basque$year,1955:1997) &
    is.element(basque$regionno,controls)
  ],ncol=length(controls))
matplot(cbind(1955:1997,1955:1997),
        cbind(Y1,Y0*%as.matrix(synth.out$solution.w)),type="l",xlab="year",ylab="GDPcap",ylim=c(0,11.5))
legend("topleft",legend=c("Basque","Synthetic Basque"),col=c("black","red"),lty=c(1,2))

## to run placebo studies, simply re-run synth
## and change the treated unit or time of intervention (see references for details)

```

synth.data

Panel Data to demonstrate the use of the Synthetic Control Method

Description

This artificial panel data set is used to demonstrate the use of the Synthetic Control Method.

Usage

```
synth.data
```

Format

A dataframe made up of 8 units: 1 treated (no 7) and 7 control (no. 2,7,13,17,29,32,36,38) , 3 predictors (X1, X2, X3), 21 time periods (1980 - 2000), a unit.names.variable column ("names") and an outcome variable column (Y). All columns have column names.

synth.tab	<i>Creates Tables that Summarize Results of Synthetic Control Group Method</i>
-----------	--

Description

This function is called after `dataprep` and `synth` in order to create tables summarizing the results of the run of the synthetic control method. The result tables can be latexed directly.

Usage

```
synth.tab(synth.res = NA,  
          dataprep.res = NA,  
          round.digit = 3)
```

Arguments

synth.res	The list resulting from the call to <code>synth</code> .
dataprep.res	The list resulting from the call to <code>dataprep</code> .
round.digit	Integer for rounding in tables.

Details

NA

Value

tab.v	The matrix that contains the table of V-weights and respective variable names.
tab.w	The matrix that contains the table of W-weights and respective unit numbers and possibly names.
tab.loss	The matrix that contains the table of W-loss and V-loss

Author(s)

Jens Hainmueller and Alexis Dimaond

References

Abadie, A. and Gardeazabal, J. (2003) Economic Costs of Conflict: A Case Study of the Basque Country *American Economic Review* 93 (1) 113–132.

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.

Abadie, A., Diamond, A., Hainmueller, J. (2011). Synth: An R Package for Synthetic Control Methods in Comparative Case Studies. *Journal of Statistical Software* 42 (13) 1–17.

See Also

[synth](#), [dataprep](#), [gaps.plot](#), [path.plot](#)

Index

*Topic **datasets**

synth.data, [20](#)

basque, [2](#)

collect.optimx, [4](#)

dataprep, [5](#), [5](#), [11–16](#), [18](#), [21](#), [22](#)

fn.V, [10](#)

gaps.plot, [6](#), [7](#), [11](#), [11](#), [14–16](#), [18](#), [22](#)

genoud, [17](#)

ipop, [11](#), [17](#)

LowRankQP, [11](#), [17](#)

optimx, [4](#), [16](#), [17](#)

path.plot, [6](#), [7](#), [11](#), [12](#), [13](#), [15](#), [16](#), [18](#), [22](#)

spec.pred.func, [14](#)

synth, [5](#), [7](#), [10–14](#), [15](#), [15–17](#), [21](#), [22](#)

synth.data, [20](#)

synth.tab, [7](#), [11](#), [12](#), [14–16](#), [18](#), [21](#)