

Package ‘scpi’

September 7, 2022

Title Prediction Intervals for Synthetic Control Methods with Multiple Treated Units and Staggered Adoption

Version 1.0.0

Description Implementation of estimation and inference procedures for Synthetic Control methods using least square, lasso, ridge, or simplex-type constraints. Uncertainty is quantified with prediction intervals as developed in Cattaneo, Feng, and Titiunik (2021) <https://cattaneo.princeton.edu/papers/Cattaneo-Feng-Titiunik_2021_JASA.pdf>.

Depends R (>= 4.1.0)

License GPL-2

Encoding UTF-8

Imports abind (>= 1.4.5), CVXR (>= 1.0.10), doRNG (>= 1.8.2), doSNOW (>= 1.0.19), dplyr (>= 1.0.7), fastDummies (>= 1.6.3), foreach (>= 1.5.1), ggplot2 (>= 3.3.3), magrittr (>= 2.0.1), MASS (>= 7.3), Matrix (>= 1.3.3), methods (>= 4.1.0), nloptr (>= 1.2.2.2), parallel (>= 4.1.0), purrr (>= 0.3.4), Qtools (>= 1.5.4), reshape2 (>= 1.4.4), rlang (>= 0.4.11), stats (>= 4.1.0), stringr (>= 1.4.0), tibble (>= 3.1.2), tidyr (>= 1.1.3), utils (>= 4.1.1)

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.1

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation no

Author Matias Cattaneo [aut],
Yingjie Feng [aut],
Filippo Palomba [aut, cre],
Rocio Titiunik [aut]

Maintainer Filippo Palomba <fpalomba@princeton.edu>

R topics documented:

scpi-package	2
print.scest	3
print.scpi	3

scdata	4
scdataMulti	7
scest	11
scpi	15
scpi_germany	23
scplot	24
scplotMulti	26
summary.scest	28
summary.scpi	29

Index	30
--------------	-----------

scpi-package	<i>scpi: A package to compute Synthetic Control Prediction Intervals with Multiple Treated Units and Staggered Adoption</i>
--------------	---

Description

The package implements estimation, inference procedures, and produces plots for Synthetic Control (SC) methods using least squares, lasso, ridge, or simplex-type constraints. Uncertainty is quantified using prediction intervals according to Cattaneo, Feng, and Titiunik (2021).

Included functions are: `scdata` and `scdataMulti` for data preparation, `scest` for point estimation, `scpi` for inference procedures, and `scplot` and `scplotMulti` for plots.

`print()` and `summary()` methods are available for `scest` and `scpi`.

Companion *Stata* and *Python* packages are described in Cattaneo, Feng, Palomba, and Titiunik (2022).

Related *Stata*, *R*, and *Python* packages useful for inference in SC designs are described in the following website:

<https://nppackages.github.io/scpi/>

For an introduction to synthetic control methods, see Abadie (2021) and references therein.

Author(s)

Matias Cattaneo, Princeton University. <cattaneo@princeton.edu>.

Yingjie Feng, Tsinghua University. <fengyj@sem.tsinghua.edu.cn>.

Filippo Palomba, Princeton University (maintainer). <fpalomba@princeton.edu>.

Rocio Titiunik, Princeton University. <titiunik@princeton.edu>.

References

- Abadie, A. (2021). Using synthetic controls: Feasibility, data requirements, and methodological aspects. *Journal of Economic Literature*, 59(2), 391-425.
- Cattaneo, M. D., Feng, Y., and Titiunik, R. (2021). Prediction intervals for synthetic control methods. *Journal of the American Statistical Association*, 116(536), 1865-1880.
- Cattaneo, M. D., Feng, Y., Palomba F., and Titiunik, R. (2022). *scpi: Uncertainty Quantification for Synthetic Control Estimators*, *arXiv:2202.05984*.

print.scest	<i>Print Method for Synthetic Control Estimation</i>
-------------	--

Description

The print method for synthetic control estimation objects.

Usage

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

Arguments

x	Class "scpi_scest" or "scpi_scest_multi" object, obtained by calling scest .
...	Other arguments.

Value

No return value, called to print [scest](#) results.

Author(s)

Matias Cattaneo, Princeton University. <cattaneo@princeton.edu>.
Yingjie Feng, Tsinghua University. <fengyj@sem.tsinghua.edu.cn>.
Filippo Palomba, Princeton University (maintainer). <fpalomba@princeton.edu>.
Rocio Titiunik, Princeton University. <titiunik@princeton.edu>.

See Also

[scest](#) for synthetic control estimation.
Supported methods: [print.scest](#), [summary.scest](#).

print.scpi	<i>Print Method for Synthetic Control Inference</i>
------------	---

Description

The print method for synthetic control inference objects.

Usage

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

Arguments

x	Class "scpi" object, obtained by calling scpi .
...	Other arguments.

Value

No return value, called to print `scpi` results.

Author(s)

Matias Cattaneo, Princeton University. <cattaneo@princeton.edu>.

Yingjie Feng, Tsinghua University. <fengyj@sem.tsinghua.edu.cn>.

Filippo Palomba, Princeton University (maintainer). <fpalomba@princeton.edu>.

Rocio Titiunik, Princeton University. <titiunik@princeton.edu>.

See Also

`scpi` for synthetic control inference

Supported methods: `print.scpi`, `summary.scpi`.

scdata

Data preparation to use before calling `scest` or `scpi` for point estimation and inference procedures using Synthetic Control.

Description

The command prepares the data to be used by `scest` or `scpi` to implement estimation and inference procedures for Synthetic Control (SC) methods. It allows the user to specify the outcome variable, the features of the treated unit to be matched, and covariate-adjustment feature by feature. The names of the output matrices follow the terminology proposed in Cattaneo, Feng, and Titiunik (2021).

Companion `Stata` and `Python` packages are described in Cattaneo, Feng, Palomba, and Titiunik (2022).

Companion commands are: `scdataMulti` for data preparation in the multiple treated units case with staggered adoption, `scest` for point estimation, `scpi` for inference procedures, `scplot` and `scplotMulti` for plots in the single and multiple treated unit(s) cases, respectively.

Related `Stata`, `R`, and `Python` packages useful for inference in SC designs are described in the following website:

<https://nppackages.github.io/scpi/>

For an introduction to synthetic control methods, see Abadie (2021) and references therein.

Usage

```
scdata(
  df,
  id.var,
  time.var,
  outcome.var,
  period.pre,
  period.post,
  unit.tr,
  unit.co,
  features = NULL,
```

```

cov.adj = NULL,
cointegrated.data = FALSE,
anticipation = 0,
constant = FALSE,
verbose = TRUE
)

```

Arguments

<code>df</code>	a dataframe object.
<code>id.var</code>	a character or numeric scalar with the name of the variable containing units' IDs. The ID variable can be numeric or character.
<code>time.var</code>	a character with the name of the time variable. The time variable has to be numeric, integer, or Date. In case <code>time.var</code> is Date it should be the output of <code>as.Date()</code> function. An integer or numeric time variable is suggested when working with yearly data, whereas for all other formats a Date type time variable is preferred.
<code>outcome.var</code>	a character with the name of the outcome variable. The outcome variable has to be numeric.
<code>period.pre</code>	a numeric vector that identifies the pre-treatment period in <code>time.var</code> .
<code>period.post</code>	a numeric vector that identifies the post-treatment period in <code>time.var</code> .
<code>unit.tr</code>	a character or numeric scalar that identifies the treated unit in <code>id.var</code> .
<code>unit.co</code>	a character or numeric vector that identifies the donor pool in <code>id.var</code> .
<code>features</code>	a character vector containing the name of the feature variables used for estimation. If <code>features</code> is specified, then <code>outcome.var</code> must be included in it. If this option is not specified the default is <code>features = outcome.var</code> .
<code>cov.adj</code>	a list specifying the names of the covariates to be used for adjustment for each feature. See the Details section for more.
<code>cointegrated.data</code>	a logical that indicates if there is a belief that the data is cointegrated or not. The default value is FALSE. See the Details section for more.
<code>anticipation</code>	a scalar that indicates the number of periods of potential anticipation effects. Default is 0.
<code>constant</code>	a logical which controls the inclusion of a constant term across features. The default value is FALSE.
<code>verbose</code>	if TRUE prints additional information in the console.

Details

- `cov.adj` can be used in two ways. First, if only one feature is specified through the option `features`, `cov.adj` has to be a list with one (even unnamed) element (eg. `cov.adj = list(c("constant", "trend"))`). Alternatively, if multiple features are specified, then the user has two possibilities:
 - provide a list with one element, then the same covariates are used for adjustment for each feature. For example, if there are two features specified and the user inputs `cov.adj = list(c("constant", "trend"))`, then a constant term and a linear trend are for adjustment for both features.

- provide a list with as many elements as the number of features specified, then feature-specific covariate adjustment is implemented. For example, `cov.adj = list('f1' = c("constant", "trend"), 'f2' = c("trend"))`. In this case the name of each element of the list should be one (and only one) of the features specified.

This option allows the user to include feature-specific constant terms

- `cointegrated.data` allows the user to model the belief that **A** and **B** form a cointegrated system. In practice, this implies that when dealing with the pseudo-true residuals **u**, the first-difference of **B** are used rather than the levels.

Value

The command returns an object of class 'scpi_data' containing the following

A	a matrix containing pre-treatment features of the treated unit.
B	a matrix containing pre-treatment features of the control units.
C	a matrix containing covariates for adjustment.
P	a matrix whose rows are the vectors used to predict the out-of-sample series for the synthetic unit.
Y.pre	a matrix containing the pre-treatment outcome of the treated unit.
Y.post	a matrix containing the post-treatment outcome of the treated unit.
Y.donors	a matrix containing the pre-treatment outcome of the control units.
specs	a list containing some specifics of the data: <ul style="list-style-type: none"> • J, the number of control units • K, a numeric vector with the number of covariates used for adjustment for each feature • KM, the total number of covariates used for adjustment • M, number of features • period.pre, a numeric vector with the pre-treatment period • period.post, a numeric vector with the post-treatment period • T0.features, a numeric vector with the number of periods used in estimation for each feature • T1.outcome, the number of post-treatment periods • outcome.var, a character with the name of the outcome variable • features, a character vector with the name of the features • constant, for internal use only • out.in.features, for internal use only • effect, for internal use only • treated.units, list containing the IDs of all treated units

Author(s)

Matias Cattaneo, Princeton University. <cattaneo@princeton.edu>.

Yingjie Feng, Tsinghua University. <fengyj@sem.tsinghua.edu.cn>.

Filippo Palomba, Princeton University (maintainer). <fpalomba@princeton.edu>.

Rocio Titiunik, Princeton University. <titiunik@princeton.edu>.

References

- [Abadie, A. \(2021\)](#). Using synthetic controls: Feasibility, data requirements, and methodological aspects. *Journal of Economic Literature*, 59(2), 391-425.
- [Cattaneo, M. D., Feng, Y., and Titiunik, R. \(2021\)](#). Prediction intervals for synthetic control methods. *Journal of the American Statistical Association*, 116(536), 1865-1880.
- [Cattaneo, M. D., Feng, Y., Palomba F., and Titiunik, R. \(2022\)](#). scpi: Uncertainty Quantification for Synthetic Control Estimators, *arXiv:2202.05984*.

See Also

[sccdataMulti](#), [sccest](#), [scpi](#), [scplot](#), [scplotMulti](#)

Examples

```
data <- scpi_germany

df <- sccdata(df = data, id.var = "country", time.var = "year",
             outcome.var = "gdp", period.pre = (1960:1990),
             period.post = (1991:2003), unit.tr = "West Germany",
             unit.co = unique(data$country)[-7], constant = TRUE,
             cointegrated.data = TRUE)
```

sccdataMulti

Data preparation to use before calling sccest or scpi for point estimation and inference procedures using Synthetic Control with Multiple Treated Units and Staggered Adoption.

Description

The command prepares the data to be used by [sccest](#) or [scpi](#) to implement estimation and inference procedures for Synthetic Control (SC) methods in the general case of multiple treated units and staggered adoption. It is a generalization of [sccdata](#), since this latter prepares the data in the particular case of a single treated unit.

The names of the output matrices follow the terminology proposed in [Cattaneo, Feng, Palomba and Titiunik \(2022\)](#) ([UPDATE LINK](#)).

Companion [Stata](#) and [Python](#) packages are described in [Cattaneo, Feng, Palomba, and Titiunik \(2022\)](#).

Companion commands are: [sccdata](#) for data preparation in the single treated unit case, [sccest](#) for point estimation, [scpi](#) for inference procedures, [scplot](#) and [scplotMulti](#) for plots in the single and multiple treated unit(s) cases, respectively.

Related Stata, R, and Python packages useful for inference in SC designs are described in the following website:

<https://nppackages.github.io/scpi/>

For an introduction to synthetic control methods, see [Abadie \(2021\)](#) and references therein.

Usage

```
scdataMulti(
  df,
  id.var,
  time.var,
  outcome.var,
  treatment.var,
  features = NULL,
  cov.adj = NULL,
  cointegrated.data = FALSE,
  post.est = NULL,
  units.est = NULL,
  anticipation = 0,
  effect = "unit-time",
  constant = FALSE,
  verbose = TRUE,
  sparse.matrices = FALSE
)
```

Arguments

<code>df</code>	a dataframe object.
<code>id.var</code>	a character with the name of the variable containing units' IDs. The ID variable can be numeric or character.
<code>time.var</code>	a character with the name of the time variable. The time variable has to be numeric, integer, or Date. In case <code>time.var</code> is Date it should be the output of as.Date() function. An integer or numeric time variable is suggested when working with yearly data, whereas for all other formats a Date type time variable is preferred.
<code>outcome.var</code>	a character with the name of the outcome variable. The outcome variable has to be numeric.
<code>treatment.var</code>	a character with the name of the variable containing the treatment assignment of each unit. The referenced variable has to take value 1 if the unit is treated in that period and value 0 otherwise. Please notice that, as common in the SC literature, we presume that once a unit is treated it remains treated forever. If <code>treatment.var</code> does not comply with this requirement the command would not work as expected!
<code>features</code>	a list containing the names of the feature variables used for estimation. If <code>features</code> is specified, then <code>outcome.var</code> must be included in it. If this option is not specified the default is <code>features = outcome.var</code> .
<code>cov.adj</code>	a list specifying the names of the covariates to be used for adjustment for each feature.
<code>cointegrated.data</code>	a logical that indicates if there is a belief that the data is cointegrated or not. The default value is FALSE.
<code>post.est</code>	a scalar specifying the number of post-treatment periods or a list specifying the periods for which treatment effects have to be estimated for each treated unit.
<code>units.est</code>	a list specifying the treated units for which treatment effects have to be estimated.

anticipation	a scalar that indicates the number of periods of potential anticipation effects. Default is 0.
effect	a string indicating the type of treatment effect to be estimated. Options are: 'unit-time', which estimates treatment effects for each treated unit-time combination; 'unit', which estimates the treatment effect for each unit by averaging post-treatment features over time.
constant	a logical which controls the inclusion of a constant term across features. The default value is FALSE.
verbose	if TRUE prints additional information in the console.
sparse.matrices	if TRUE all block diagonal matrices (B , C , and P) are sparse matrices. This is suggested if the dimension of the dataset is large as it will likely reduce the execution time. The sparse matrices will be objects of class 'dgCMatrix' or 'lgCMatrix', thus to visualize them they need to be transformed in matrices, e.g. <code>View(as.matrix(B))</code> .

Details

- **Covariate-adjustment.** See the **Details** section in [sccdata](#) for further information on how to specify covariate-adjustment feature-by-feature.
- **Cointegration.** `cointegrated.data` allows the user to model the belief that **A** and **B** form a cointegrated system. In practice, this implies that when dealing with the pseudo-true residuals **u**, the first-difference of **B** are used rather than the levels.
- **Effect.** `effect` allows the user to select between two causal quantities. The default option, `effect = "unit-time"`, prepares the data for estimation of

$$\tau_{ik}, \quad k \geq, i = 1, \dots, N_1,$$

whereas the option `effect = "unit"` prepares the data for estimation of

$$\tau_{\cdot k} = \frac{1}{N_1} \sum_{i=1}^{N_1} \tau_{ik}$$

which is the average effect on the treated unit across multiple post-treatment periods.

Value

The command returns an object of class 'scpi_data' containing the following

A	a matrix containing pre-treatment features of the treated units.
B	a matrix containing pre-treatment features of the control units.
C	a matrix containing covariates for adjustment.
P	a matrix whose rows are the vectors used to predict the out-of-sample series for the synthetic units.
P.diff	for internal use only.
Y.df	a dataframe containing the outcome variable for all units.
Y.donors	a matrix containing the pre-treatment outcome of the control units.
specs	a list containing some specifics of the data: <ul style="list-style-type: none"> • J, a list containing the number of donors for each treated unit

- `K`, a list containing the number of covariates used for adjustment for each feature for each treated unit
- `KM`, a list containing the total number of covariates used for adjustment for each treated unit
- `M`, a list containing number of features used for each treated unit
- `I`, number of treated units
- `KMI`, overall number of covariates used for adjustment
- `period.pre`, a list containing a numeric vector with the pre-treatment period for each treated unit
- `period.post`, a list containing a numeric vector with the post-treatment period for each treated unit
- `T0.features`, a list containing a numeric vector with the number of periods used in estimation for each feature for each treated unit
- `T1.outcome`, a list containing the number of post-treatment periods for each treated unit
- `features.list`, a list containing the name of the features for each treated unit
- `outcome.var`, a character containing the name of the outcome variable
- `constant`, for internal use only
- `effect`, for internal use only
- `anticipation`, number of periods of potential anticipation effects
- `out.in.features`, for internal use only
- `treated.units`, list containing the IDs of all treated units
- `donors.list`, list containing the IDs of the donors of each treated unit

Author(s)

Matias Cattaneo, Princeton University. <cattaneo@princeton.edu>.

Yingjie Feng, Tsinghua University. <fengyj@sem.tsinghua.edu.cn>.

Filippo Palomba, Princeton University (maintainer). <fpalomba@princeton.edu>.

Rocio Titiunik, Princeton University. <titiunik@princeton.edu>.

References

- [Abadie, A. \(2021\)](#). Using synthetic controls: Feasibility, data requirements, and methodological aspects. *Journal of Economic Literature*, 59(2), 391-425.
- [Cattaneo, M. D., Feng, Y., and Titiunik, R. \(2021\)](#). Prediction intervals for synthetic control methods. *Journal of the American Statistical Association*, 116(536), 1865-1880.
- [Cattaneo, M. D., Feng, Y., Palomba F., and Titiunik, R. \(2022\)](#). scpi: Uncertainty Quantification for Synthetic Control Estimators, *arXiv:2202.05984*.

See Also

[scdata](#), [scest](#), [scpi](#), [scplot](#), [scplotMulti](#)

Examples

```
datager <- scpi_germany

datager$tr_id <- 0
datager$tr_id[(datager$country == "West Germany" & datager$year > 1990)] <- 1
datager$tr_id[(datager$country == "Italy" & datager$year > 1992)] <- 0

outcome.var <- "gdp"
id.var <- "country"
treatment.var <- "tr_id"
time.var <- "year"
df.unit <- scdataMulti(datager, id.var = id.var, outcome.var = outcome.var,
                      treatment.var = treatment.var,
                      time.var = time.var, features = list(c("gdp", "trade")),
                      cointegrated.data = TRUE, constant = TRUE)
```

scest

Estimation of Synthetic Control

Description

The command implements estimation procedures for Synthetic Control (SC) methods using least squares, lasso, ridge, or simplex-type constraints. For more information see [Cattaneo, Feng, and Titiunik \(2021\)](#).

Companion [Stata](#) and [Python](#) packages are described in [Cattaneo, Feng, Palomba, and Titiunik \(2022\)](#).

Companion commands are: [scdata](#) and [scdataMulti](#) for data preparation in the single and multiple treated unit(s) cases, respectively, [scpi](#) for inference procedures, [scplot](#) and [scplotMulti](#) for plots in the single and multiple treated unit(s) cases, respectively.

Related Stata, R, and Python packages useful for inference in SC designs are described in the following website:

<https://nppackages.github.io/scpi/>

For an introduction to synthetic control methods, see [Abadie \(2021\)](#) and references therein.

Usage

```
scest(
  data,
  w.constr = NULL,
  V = "separate",
  opt.list = NULL,
  plot = FALSE,
  plot.name = NULL,
  plot.path = NULL,
  save.data = NULL
)
```

Arguments

data	a class 'scpi_data' object, obtained by calling <code>sdata</code> , or class 'scpi_data_multi' obtained via <code>sdataMulti</code> .
w.constr	a list specifying the constraint set the estimated weights of the donors must belong to. w.constr can contain up to four objects: <ul style="list-style-type: none"> • 'p', a string indicating the norm to be constrained (p should be one of "no norm", "L1", and "L2") • 'dir', a string indicating whether the constraint on the norm is an equality ("==") or inequality ("<=") • 'Q', a scalar defining the value of the constraint on the norm • 'lb', a scalar defining the lower bound on the weights. It can be either 0 or -Inf. • 'name', a character selecting one of the default proposals. See the Details section for more.
V	specifies the weighting matrix to be used when minimizing the sum of squared residuals $(\mathbf{A} - \mathbf{B}\mathbf{w} - \mathbf{C}\mathbf{r})'\mathbf{V}(\mathbf{A} - \mathbf{B}\mathbf{w} - \mathbf{C}\mathbf{r})$ <p>The default is the identity matrix, so equal weight is given to all observations. In the case of multiple treated observations (you used <code>sdataMulti</code> to prepare the data), the user can specify V as a string equal to either "separate" or "pooled". See the Details section for more.</p>
opt.list	a list specifying the stopping criteria and the algorithm for the underlying optimizer (<code>nloptr</code> or <code>CVXR</code>) for point estimation. See the Details section for more.
plot	a logical specifying whether <code>scplot</code> should be called and a plot saved in the current working directory. For more options see <code>scplot</code> .
plot.name	a string containing the name of the plot (the format is by default .png). For more options see <code>scplot</code> .
plot.path	a string containing the path at which the plot should be saved (default is output of <code>getwd()</code> .)
save.data	a character specifying the name and the path of the saved dataframe containing the processed data used to produce the plot.

Details

Information is provided for the simple case in which $N_1 = 1$ if not specified otherwise.

- **Estimation of Weights.** w.constr specifies the constraint set on the weights. First, the element p allows the user to choose between imposing a constraint on either the L1 (p = "L1") or the L2 (p = "L2") norm of the weights and imposing no constraint on the norm (p = "no norm"). Second, Q specifies the value of the constraint on the norm of the weights. Third, lb sets the lower bound of each component of the vector of weights. Fourth, dir sets the direction of the constraint on the norm in case p = "L1" or p = "L2". If dir = "==", then

$$\|\mathbf{w}\|_p = Q, \quad w_j \geq lb, \quad j = 1, \dots, J$$

If instead dir = "<=", then

$$\|\mathbf{w}\|_p \leq Q, \quad w_j \geq lb, \quad j = 1, \dots, J$$

If instead dir = "NULL" no constraint on the norm of the weights is imposed.

An alternative to specifying an ad-hoc constraint set on the weights would be choosing among some popular types of constraints. This can be done by including the element 'name' in the list `w.constr`. The following are available options:

- If `name == "simplex"` (the default), then

$$\|\mathbf{w}\|_1 = 1, \quad w_j \geq 0, \quad j = 1, \dots, J.$$

- If `name == "lasso"`, then

$$\|\mathbf{w}\|_1 \leq Q,$$

where Q is by default equal to 1 but it can be provided as an element of the list (eg. `w.constr = list(name = "lasso", Q = 2)`).

- If `name == "ridge"`, then

$$\|\mathbf{w}\|_2 \leq Q,$$

where Q is a tuning parameter that is by default computed as

$$(J + KM)\hat{\sigma}_u^2 / \|\hat{\mathbf{w}}_{OLS}\|_2^2$$

where J is the number of donors and KM is the total number of covariates used for adjustment. The user can provide Q as an element of the list (eg. `w.constr = list(name = "ridge", Q = 1)`).

- If `name == "ols"`, then the problem is unconstrained and the vector of weights is estimated via ordinary least squares.
- If `name == "L1-L2"`, then

$$\|\mathbf{w}\|_1 = 1, \quad \|\mathbf{w}\|_2 \leq Q,$$

where Q is a tuning parameter computed as in the "ridge" case.

- **Weighting Matrix with Multiple Treated Units.**

- if `V <- "separate"`, then $\mathbf{V} = \mathbf{I}$ and the minimized objective function is

$$\sum_{i=1}^{N_1} \sum_{l=1}^M \sum_{t=1}^{T_0} (a_{t,l}^i - \mathbf{b}_{t,l}^{i'} \mathbf{w}^i - \mathbf{c}_{t,l}^{i'} \mathbf{r}_l^i)^2,$$

which optimizes the separate fit for each treated unit.

- if `V <- "pooled"`, then $\mathbf{V} = \frac{1}{J} \mathbf{1} \mathbf{1}' \otimes \mathbf{I}$ and the minimized objective function is

$$\sum_{l=1}^M \sum_{t=1}^{T_0} \left(\frac{1}{N_1} \sum_{i=1}^{N_1} (a_{t,l}^i - \mathbf{b}_{t,l}^{i'} \mathbf{w}^i - \mathbf{c}_{t,l}^{i'} \mathbf{r}_l^i) \right)^2,$$

which optimizes the pooled fit for the average of the treated units.

- **Algorithm Options.** The default is a sequential quadratic programming (SQP) algorithm for nonlinearly constrained gradient-based optimization (`algorithm = 'NLOPTRD_LD_SLSQP'`) for all cases not involving the L1 norm. For a complete list of algorithms see [the official nloptr documentation](#). The other default values are `maxeval = 5000`, `ftol_res = 1.0e-8`, `ftol_abs = 1.0e-8`, `xtol_res = 1.0e-8`, `xtol_abs = 1.0e-8`, `tol_constraints_eq = 1.0e-8`, and, finally, `tol_constraints_ineq = 1.0e-8`. More information on the stopping criteria can be obtained running `nloptr.print.options()` or `nloptr.get.default.options()`. If the optimization involves the L1 norm then CVXR is used for optimization. More information on the stopping criteria can be obtained reading [the official documentation](#).

Value

The function returns an object of class 'scpi_scest' containing two lists. The first list is labeled 'data' and contains used data as returned by [scdata](#) and some other values.

A	a matrix containing pre-treatment features of the treated unit.
B	a matrix containing pre-treatment features of the control units.
C	a matrix containing covariates for adjustment.
P	a matrix whose rows are the vectors used to predict the out-of-sample series for the synthetic unit.
P.diff	for internal use only.
Y.pre	a matrix containing the pre-treatment outcome of the treated unit.
Y.post	a matrix containing the post-treatment outcome of the treated unit.
Y.donors	a matrix containing the pre-treatment outcome of the control units.
specs	a list containing some specifics of the data: <ul style="list-style-type: none"> • J, the number of control units • K, a numeric vector with the number of covariates used for adjustment for each feature • M, number of features • KM, the total number of covariates used for adjustment • KMI, the total number of covariates used for adjustment • I, number of treated units • period.pre, a numeric vector with the pre-treatment period • period.post, a numeric vector with the post-treatment period • T0.features, a numeric vector with the number of periods used in estimation for each feature • T1.outcome, the number of post-treatment periods • constant, for internal use only • effect, for internal use only • anticipation, number of periods of potential anticipation effects • out.in.features, for internal use only • treated.units, list containing the IDs of all treated units • donors.list, list containing the IDs of the donors of each treated unit • class.type, for internal use only

The second list is labeled 'est.results' and contains estimation results.

w	a matrix containing the estimated weights of the donors.
r	a matrix containing the values of the covariates used for adjustment.
b	a matrix containing w and r .
Y.pre.fit	a matrix containing the estimated pre-treatment outcome of the SC unit.
Y.post.fit	a matrix containing the estimated post-treatment outcome of the SC unit.
A.hat	a matrix containing the predicted values of the features of the treated unit.
res	a matrix containing the residuals $\mathbf{A} - \hat{\mathbf{A}}$.
V	a matrix containing the weighting matrix used in estimation.
w.constr	a list containing the specifics of the constraint set used on the weights.
S	for internal use only.

Author(s)

Matias Cattaneo, Princeton University. <cattaneo@princeton.edu>.

Yingjie Feng, Tsinghua University. <fengyj@sem.tsinghua.edu.cn>.

Filippo Palomba, Princeton University (maintainer). <fpalomba@princeton.edu>.

Rocio Titiunik, Princeton University. <titiunik@princeton.edu>.

References

- [Abadie, A. \(2021\)](#). Using synthetic controls: Feasibility, data requirements, and methodological aspects. *Journal of Economic Literature*, 59(2), 391-425.
- [Cattaneo, M. D., Feng, Y., and Titiunik, R. \(2021\)](#). Prediction intervals for synthetic control methods. *Journal of the American Statistical Association*, 116(536), 1865-1880.
- [Cattaneo, M. D., Feng, Y., Palomba F., and Titiunik, R. \(2022\)](#). scpi: Uncertainty Quantification for Synthetic Control Estimators, *arXiv:2202.05984*.

See Also

[scdataMulti](#), [scdata](#), [scpi](#), [scplot](#), [scplotMulti](#)

Examples

```
data <- scpi_germany

df <- scdata(df = data, id.var = "country", time.var = "year",
            outcome.var = "gdp", period.pre = (1960:1990),
            period.post = (1991:2003), unit.tr = "West Germany",
            unit.co = unique(data$country)[-7], constant = TRUE,
            cointegrated.data = TRUE)

result <- scest(df, w.constr = list(name = "simplex", Q = 1))
result <- scest(df, w.constr = list(lb = 0, dir = "==", p = "L1", Q = 1))
```

Description

The command implements estimation and inference procedures for Synthetic Control (SC) methods using least squares, lasso, ridge, or simplex-type constraints. Uncertainty is quantified using prediction intervals according to [Cattaneo, Feng, and Titiunik \(2021\)](#). [scpi](#) returns the estimated post-treatment series for the synthetic unit through the command [scest](#) and quantifies in-sample and out-of-sample uncertainty to provide confidence intervals for each point estimate.

Companion [Stata](#) and [Python](#) packages are described in [Cattaneo, Feng, Palomba, and Titiunik \(2022\)](#).

Companion commands are: [scdata](#) and [scdataMulti](#) for data preparation in the single and multiple treated unit(s) cases, respectively, [scest](#) for point estimation, [scplot](#) and [scplotMulti](#) for plots in the single and multiple treated unit(s) cases, respectively.

Related Stata, R, and Python packages useful for inference in SC designs are described in the following website:

<https://nppackages.github.io/scpi/>

For an introduction to synthetic control methods, see [Abadie \(2021\)](#) and references therein.

Usage

```
scpi(
  data,
  w.constr = NULL,
  V = "separate",
  P = NULL,
  u.missp = TRUE,
  u.sigma = "HC1",
  u.order = 1,
  u.lags = 0,
  u.design = NULL,
  u.alpha = 0.05,
  e.method = "all",
  e.order = 1,
  e.lags = 0,
  e.design = NULL,
  e.alpha = 0.05,
  sims = 200,
  rho = NULL,
  rho.max = NULL,
  lgapp = "generalized",
  cores = NULL,
  plot = FALSE,
  plot.name = NULL,
  w.bounds = NULL,
  e.bounds = NULL,
  opt.list.est = NULL,
  opt.list.inf = NULL,
  save.data = NULL,
  verbose = TRUE
)
```

Arguments

- | | |
|----------|---|
| data | a class 'scpi_data' object, obtained by calling scdata . |
| w.constr | <p>a list specifying the constraint set the estimated weights of the donors must belong to. w.constr can contain up to five elements:</p> <ul style="list-style-type: none"> • 'p', a scalar indicating the norm to be used (p should be one of "no norm", "L1", and "L2") • 'dir', a string indicating whether the constraint on the norm is an equality ("==") or inequality ("<=") • 'Q', a scalar defining the value of the constraint on the norm • 'lb', a scalar defining the lower bound on the weights. It can be either 0 or -Inf. |

- 'name', a character selecting one of the default proposals See the **Details** section for more.

V a weighting matrix to be used when minimizing

$$(\mathbf{A} - \mathbf{B}\mathbf{w} - \mathbf{C}\mathbf{r})'\mathbf{V}(\mathbf{A} - \mathbf{B}\mathbf{w} - \mathbf{C}\mathbf{r})$$

P a $I \cdot T_1 \times I \cdot (J + KM)$ matrix containing the design matrix to be used to obtain the predicted, post-intervention outcome of the synthetic control unit. T_1 is the number of post-treatment periods, J is the size of the donor pool, and K_1 is the number of covariates used for adjustment in the outcome equation.

u.missp a logical indicating if misspecification should be taken into account when dealing with **u**.

u.sigma a string specifying the type of variance-covariance estimator to be used when estimating the conditional variance of **u**.

u.order a scalar that sets the order of the polynomial in **B** when predicting moments of **u**. The default is `u.order = 1`, however if there is risk of over-fitting, the command automatically sets it to `u.order = 0`. See the **Details** section for more information.

u.lags a scalar that sets the number of lags of **B** when predicting moments of **u**. The default is `u.lags = 0`, however if there is risk of over-fitting, the command automatically sets it to `u.lags = 0`. See the **Details** section for more information.

u.design a matrix with the same number of rows of **A** and **B** and whose columns specify the design matrix to be used when modeling the estimated pseudo-true residuals **u**.

u.alpha a scalar specifying the confidence level for in-sample uncertainty, i.e. `1 - u.alpha` is the confidence level.

e.method a string selecting the method to be used in quantifying out-of-sample uncertainty among: "gaussian" which uses conditional subgaussian bounds; "ls" which specifies a location-scale model for **u**; "qreg" which employs a quantile regressions to get the conditional bounds; "all" uses each one of the previous methods.

e.order a scalar that sets the order of the polynomial in **B** when predicting moments of **e**. The default is `e.order = 1`, however if there is risk of over-fitting, the command automatically sets it to `e.order = 0`. See the **Details** section for more information.

e.lags a scalar that sets the number of lags of **B** when predicting moments of **e**. The default is `e.order = 1`, however if there is risk of over-fitting, the command automatically sets it to `e.order = 0`. See the **Details** section for more information.

e.design a matrix with the same number of rows of **A** and **B** and whose columns specify the design matrix to be used when modeling the estimated out-of-sample residuals **e**.

e.alpha a scalar specifying the confidence level for out-of-sample uncertainty, i.e. `1 - e.alpha` is the confidence level.

sims a scalar providing the number of simulations to be used in quantifying in-sample uncertainty.

rho a string specifying the regularizing parameter that imposes sparsity on the estimated vector of weights. If `rho = 'type-1'` (the default), then the tuning parameter is computed based on optimization inequalities. Users can provide a scalar with their own value for `rho`. Other options are described in the **Details** section.

rho.max	a scalar indicating the maximum value attainable by the tuning parameter rho.
lgapp	selects the way local geometry is approximated in simulation. The options are "generalized" and "linear". The first one accommodates for possibly non-linear constraints, whilst the second one is valid with linear constraints only.
cores	number of cores to be used by the command. The default is the number of cores available minus one.
plot	a logical specifying whether <code>scplot</code> should be called and a plot saved in the current working directory. For more options see <code>scplot</code> .
plot.name	a string containing the name of the plot (the format is by default .png). For more options see <code>scplot</code> .
w.bounds	a $N_1 \cdot T_1 \times 2$ matrix with the user-provided bounds on β . If w.bounds is provided, then the quantification of in-sample uncertainty is skipped. It is possible to provide only the lower bound or the upper bound by filling the other column with NAs.
e.bounds	a $N_1 \cdot T_1 \times 2$ matrix with the user-provided bounds on $(\hat{\mathbf{w}}, \hat{\mathbf{r}})'$. If e.bounds is provided, then the quantification of out-of-sample uncertainty is skipped. It is possible to provide only the lower bound or the upper bound by filling the other column with NAs.
opt.list.est	a list specifying the stopping criteria and the algorithm for the underlying optimizer <code>nloptr</code> or <code>CVXR</code> for point estimation. See the Details section for more.
opt.list.inf	similar to the previous one but for the optimizer used for inference purposes. See the Details section for more.
save.data	a character specifying the name and the path of the saved dataframe containing the processed data used to produce the plot.
verbose	if TRUE prints additional information in the console.

Details

Information is provided for the simple case in which $N_1 = 1$ if not specified otherwise.

- **Estimation of Weights.** w.constr specifies the constraint set on the weights. First, the element p allows the user to choose between imposing a constraint on either the L1 (p = "L1") or the L2 (p = "L2") norm of the weights and imposing no constraint on the norm (p = "no norm"). Second, Q specifies the value of the constraint on the norm of the weights. Third, lb sets the lower bound of each component of the vector of weights. Fourth, dir sets the direction of the constraint on the norm in case p = "L1" or p = "L2". If dir = "=", then

$$\|\mathbf{w}\|_p = Q, \quad w_j \geq lb, \quad j = 1, \dots, J$$

If instead dir = "<=", then

$$\|\mathbf{w}\|_p \leq Q, \quad w_j \geq lb, \quad j = 1, \dots, J$$

If instead dir = "NULL" no constraint on the norm of the weights is imposed.

An alternative to specifying an ad-hoc constraint set on the weights would be choosing among some popular types of constraints. This can be done by including the element 'name' in the list w.constr. The following are available options:

- If name == "simplex" (the default), then

$$\|\mathbf{w}\|_1 = 1, \quad w_j \geq 0, \quad j = 1, \dots, J.$$

- If name == "lasso", then

$$\|\mathbf{w}\|_1 \leq Q,$$

where Q is by default equal to 1 but it can be provided as an element of the list (eg. `w.constr = list(name = "lasso", Q = 2)`).

- If name == "ridge", then

$$\|\mathbf{w}\|_2 \leq Q,$$

where Q is a tuning parameter that is by default computed as

$$(J + KM)\hat{\sigma}_u^2 / \|\hat{\mathbf{w}}_{OLS}\|_2^2$$

where J is the number of donors and KM is the total number of covariates used for adjustment. The user can provide Q as an element of the list (eg. `w.constr = list(name = "ridge", Q = 1)`).

- If name == "ols", then the problem is unconstrained and the vector of weights is estimated via ordinary least squares.
- If name == "L1-L2", then

$$\|\mathbf{w}\|_1 = 1, \quad \|\mathbf{w}\|_2 \leq Q,$$

where Q is a tuning parameter computed as in the "ridge" case.

- **Weighting Matrix with Multiple Treated Units.**

- if `V <- "separate"`, then $\mathbf{V} = \mathbf{I}$ and the minimized objective function is

$$\sum_{i=1}^{N_1} \sum_{l=1}^M \sum_{t=1}^{T_0} (a_{t,l}^i - \mathbf{b}_{t,l}' \mathbf{w}^i - \mathbf{c}_{t,l}' \mathbf{r}_l^i)^2,$$

which optimizes the separate fit for each treated unit.

- if `V <- "pooled"`, then $\mathbf{V} = \mathbf{1}\mathbf{1}' \otimes \mathbf{I}$ and the minimized objective function is

$$\sum_{l=1}^M \sum_{t=1}^{T_0} \left(\frac{1}{N_1} \sum_{i=1}^{N_1} (a_{t,l}^i - \mathbf{b}_{t,l}' \mathbf{w}^i - \mathbf{c}_{t,l}' \mathbf{r}_l^i) \right)^2,$$

which optimizes the pooled fit for the average of the treated units.

- **Regularization.** rho is estimated through the formula

$$\varrho = \mathcal{C} \frac{\log(T_0)^c}{T_0^{1/2}}$$

where $\mathcal{C} = \hat{\sigma}_u / \min_j \hat{\sigma}_{b_j}$ if `rho = 'type-1'`, $\mathcal{C} = \max_j \hat{\sigma}_{b_j} \hat{\sigma}_u / \min_j \hat{\sigma}_{b_j}^2$ if `rho = 'type-2'`, and $\mathcal{C} = \max_j \hat{\sigma}_{b_j u} / \min_j \hat{\sigma}_{b_j}^2$ if `rho = 'type-3'`,
rho defines a new sparse weight vector as

$$\hat{w}_j^* = \mathbf{1}(\hat{w}_j \geq \varrho)$$

- **In-sample uncertainty.** To quantify in-sample uncertainty it is necessary to model the pseudo-residuals \mathbf{u} . First of all, estimation of the first moment of \mathbf{u} can be controlled through the option `u.missp`. When `u.missp = FALSE`, then $\mathbf{E}[\mathbf{u} \mid \mathbf{D}_u] = 0$. If instead `u.missp = TRUE`, then $\mathbf{E}[\mathbf{u} \mid \mathbf{D}_u]$ is estimated using a linear regression of $\hat{\mathbf{u}}$ on \mathbf{D}_u . The default set of variables in \mathbf{D}_u is composed of \mathbf{B} , \mathbf{C} and, if required, it is augmented with lags (`u.lags`) and polynomials (`u.order`) of \mathbf{B} . The option `u.design` allows the user to provide an ad-hoc set of variables to form \mathbf{D}_u . Regarding the second moment of \mathbf{u} , different estimators can be chosen: HC0, HC1, HC2, HC3, and HC4 using the option `u.sigma`.

- **Out-of-sample uncertainty.** To quantify out-of-sample uncertainty it is necessary to model the out-of-sample residuals \mathbf{e} and estimate relevant moments. By default, the design matrix used during estimation \mathbf{D}_e is composed of the blocks in \mathbf{B} and \mathbf{C} corresponding to the outcome variable. Moreover, if required by the user, \mathbf{D}_e is augmented with lags (`e.lags`) and polynomials (`e.order`) of \mathbf{B} . The option `e.design` allows the user to provide an ad-hoc set of variables to form \mathbf{D}_e . Finally, the option `e.method` allows the user to select one of three estimation methods: "gaussian" relies on conditional sub-Gaussian bounds; "ls" estimates conditional bounds using a location-scale model; "qreg" uses conditional quantile regression of the residuals \mathbf{e} on \mathbf{D}_e .
- **Residual Estimation Over-fitting.** To estimate conditional moments of \mathbf{u} and e_t we rely on two design matrices, \mathbf{D}_u and \mathbf{D}_e (see above). Let d_u and d_e be the number of columns in \mathbf{D}_u and \mathbf{D}_e , respectively. Assuming no missing values and balanced features, the number of observation used to estimate moments of \mathbf{u} is $N_1 \cdot T_0 \cdot M$, whilst for moments of e_t is T_0 . Our rule of thumb to avoid over-fitting is to check if $N_1 \cdot T_0 \cdot M \geq d_u + 10$ or $T_0 \geq d_e + 10$. If the former condition is not satisfied we automatically set `u.order = u.lags = 0`, if instead the latter is not met we automatically set `e.order = e.lags = 0`.
- **Algorithm Options.** The default is a sequential quadratic programming (SQP) algorithm for nonlinearly constrained gradient-based optimization (`algorithm = 'NLOPTRD_LD_SLSQP'`) for all cases not involving the L1 norm. For a complete list of algorithms see [the official nloptr documentation](#). The other default values are `maxeval = 5000`, `ftol_res = 1.0e-8`, `ftol_abs = 1.0e-8`, `xtol_res = 1.0e-8`, `xtol_abs = 1.0e-8`, `tol_constraints_eq = 1.0e-8`, and, finally, `tol_constraints_ineq = 1.0e-8`. More information on the stopping criteria can be obtained running `nloptr.print.options()` or `nloptr.get.default.options()`. If the optimization involves the L1 norm then CVXR is used for optimization. More information on the stopping criteria can be obtained reading [the official documentation](#).

Value

The function returns an object of class 'scpi_scpi' containing three lists. The first list is labeled 'data' and contains used data as returned by [scdata](#) and some other values.

A	a matrix containing pre-treatment features of the treated unit.
B	a matrix containing pre-treatment features of the control units.
C	a matrix containing covariates for adjustment.
P	a matrix whose rows are the vectors used to predict the out-of-sample series for the synthetic unit.
Y.pre	a matrix containing the pre-treatment outcome of the treated unit.
Y.post	a matrix containing the post-treatment outcome of the treated unit.
Y.donors	a matrix containing the pre-treatment outcome of the control units.
specs	a list containing some specifics of the data: <ul style="list-style-type: none"> • J, the number of control units • K, a numeric vector with the number of covariates used for adjustment for each feature • M, number of features • KM, the total number of covariates used for adjustment • KMI, the total number of covariates used for adjustment • I, number of treated units • period.pre, a numeric vector with the pre-treatment period • period.post, a numeric vector with the post-treatment period

- `T0.features`, a numeric vector with the number of periods used in estimation for each feature
- `T1.outcome`, the number of post-treatment periods
- `constant`, for internal use only
- `effect`, for internal use only
- `anticipation`, number of periods of potential anticipation effects
- `out.in.features`, for internal use only
- `treated.units`, list containing the IDs of all treated units
- `donors.list`, list containing the IDs of the donors of each treated unit

The second list is labeled 'est.results' containing all the results from `scest`.

<code>w</code>	a matrix containing the estimated weights of the donors.
<code>r</code>	a matrix containing the values of the covariates used for adjustment.
<code>b</code>	a matrix containing <code>w</code> and <code>r</code> .
<code>Y.pre.fit</code>	a matrix containing the estimated pre-treatment outcome of the SC unit.
<code>Y.post.fit</code>	a matrix containing the estimated post-treatment outcome of the SC unit.
<code>A.hat</code>	a matrix containing the predicted values of the features of the treated unit.
<code>res</code>	a matrix containing the residuals $\mathbf{A} - \hat{\mathbf{A}}$.
<code>V</code>	a matrix containing the weighting matrix used in estimation.
<code>w.constr</code>	a list containing the specifics of the constraint set used on the weights.
<code>S</code>	for internal use only.

The third list is labeled 'inference.results' and contains all the inference-related results.

<code>CI.in.sample</code>	a matrix containing the prediction intervals taking only in-sample uncertainty into account.
<code>CI.all.gaussian</code>	a matrix containing the prediction intervals estimating out-of-sample uncertainty with sub-Gaussian bounds.
<code>CI.all.ls</code>	a matrix containing the prediction intervals estimating out-of-sample uncertainty with a location-scale model.
<code>CI.all.qreg</code>	a matrix containing the prediction intervals estimating out-of-sample uncertainty with quantile regressions.
<code>bounds</code>	a list containing the estimated bounds (in-sample and out-of-sample uncertainty).
<code>Sigma</code>	a matrix containing the estimated (conditional) variance-covariance Σ .
<code>u.mean</code>	a matrix containing the estimated (conditional) mean of the pseudo-residuals \mathbf{u} .
<code>u.var</code>	a matrix containing the estimated (conditional) variance-covariance of the pseudo-residuals \mathbf{u} .
<code>e.mean</code>	a matrix containing the estimated (conditional) mean of the out-of-sample error e .
<code>e.var</code>	a matrix containing the estimated (conditional) variance of the out-of-sample error e .
<code>u.missp</code>	a logical indicating whether the model has been treated as misspecified or not.
<code>u.lags</code>	an integer containing the number of lags in <code>B</code> used in predicting moments of the pseudo-residuals \mathbf{u} .

<code>u.order</code>	an integer containing the order of the polynomial in B used in predicting moments of the pseudo-residuals \mathbf{u} .
<code>u.sigma</code>	a string indicating the estimator used for Sigma.
<code>u.user</code>	a logical indicating whether the design matrix to predict moments of \mathbf{u} was user-provided.
<code>u.T</code>	a scalar indicating the number of observations used to predict moments of \mathbf{u} .
<code>u.params</code>	a scalar indicating the number of parameters used to predict moments of \mathbf{u} .
<code>u.D</code>	the design matrix used to predict moments of \mathbf{u} ,
<code>u.alpha</code>	a scalar determining the confidence level used for in-sample uncertainty, i.e. $1-u.alpha$ is the confidence level.
<code>e.method</code>	a string indicating the specification used to predict moments of the out-of-sample error e .
<code>e.lags</code>	an integer containing the number of lags in B used in predicting moments of the out-of-sample error e .
<code>e.order</code>	an integer containing the order of the polynomial in B used in predicting moments of the out-of-sample error e .
<code>e.user</code>	a logical indicating whether the design matrix to predict moments of e was user-provided.
<code>e.T</code>	a scalar indicating the number of observations used to predict moments of \mathbf{u} .
<code>e.params</code>	a scalar indicating the number of parameters used to predict moments of \mathbf{u} .
<code>e.alpha</code>	a scalar determining the confidence level used for out-of-sample uncertainty, i.e. $1-e.alpha$ is the confidence level.
<code>e.D</code>	the design matrix used to predict moments of \mathbf{u} ,
<code>rho</code>	an integer specifying the estimated regularizing parameter that imposes sparsity on the estimated vector of weights.
<code>Q.star</code>	a list containing the regularized constraint on the norm.
<code>epskappa</code>	a vector containing the estimates for ϵ_K .
<code>sims</code>	an integer indicating the number of simulations used in quantifying in-sample uncertainty.
<code>failed.sims</code>	a matrix containing the number of failed simulations per post-treatment period to estimate lower and upper bounds.

Author(s)

Matias Cattaneo, Princeton University. <cattaneo@princeton.edu>.

Yingjie Feng, Tsinghua University. <fengyj@sem.tsinghua.edu.cn>.

Filippo Palomba, Princeton University (maintainer). <fpalomba@princeton.edu>.

Rocio Titiunik, Princeton University. <titiunik@princeton.edu>.

References

- [Abadie, A. \(2021\)](#). Using synthetic controls: Feasibility, data requirements, and methodological aspects. *Journal of Economic Literature*, 59(2), 391-425.
- [Cattaneo, M. D., Feng, Y., and Titiunik, R. \(2021\)](#). Prediction intervals for synthetic control methods. *Journal of the American Statistical Association*, 116(536), 1865-1880.
- [Cattaneo, M. D., Feng, Y., Palomba F., and Titiunik, R. \(2022\)](#). scpi: Uncertainty Quantification for Synthetic Control Estimators, *arXiv:2202.05984*.

See Also

[scdata](#), [scdataMulti](#), [scest](#), [scplot](#), [scplotMulti](#)

Examples

```
data <- scpi_germany

df <- scdata(df = data, id.var = "country", time.var = "year",
             outcome.var = "gdp", period.pre = (1960:1990),
             period.post = (1991:2003), unit.tr = "West Germany",
             unit.co = unique(data$country)[-7], constant = TRUE,
             cointegrated.data = TRUE)

result <- scpi(df, w.constr = list(name = "simplex", Q = 1), cores = 1, sims = 10)
result <- scpi(df, w.constr = list(lb = 0, dir = "==", p = "L1", Q = 1),
               cores = 1, sims = 10)
```

scpi_germany

Economic indicators for OECD countries

Description

A dataset containing some economic indicators of 17 OECD countries from 1960 to 2003.

Usage

```
scpi_germany
```

Format

A data frame with 748 rows and 11 variables:

index country index.

country name of the country.

year time index, in years.

gdp GDP per Capita (PPP, 2002 USD).

infrate annual percentage change in consumer prices (base year 1995).

trade trade openness measured as export plus imports as percentage of GDP.

schooling percentage of secondary school attained in the total population aged 25 and older.

industry industry share of value added.

Source

<https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/24714>

Description

The command plots the actual pre-treatment and post-treatment series of the treated unit and the estimated counterfactual synthetic control unit with corresponding prediction intervals. Prediction intervals can take into account either in-sample uncertainty only or in-sample and out-of-sample uncertainty using the techniques developed in Cattaneo, Feng, and Titiunik (2021). `scpi`. The input object should come from the command `scest` or from the command `scpi`.

Companion `Stata` and `Python` packages are described in Cattaneo, Feng, Palomba, and Titiunik (2022).

Companion commands are: `scdata` and `scdataMulti` for data preparation in the single and multiple treated unit(s) cases, respectively, `scest` for point estimation, `scpi` for inference procedures, and `scplotMulti` for plots with multiple treated units.

Related `Stata`, `R`, and `Python` packages useful for inference in SC designs are described in the following website:

<https://nppackages.github.io/scpi/>

For an introduction to synthetic control methods, see Abadie (2021) and references therein.

Usage

```
scplot(
  result,
  fig.path = NULL,
  fig.name = NULL,
  fig.format = "png",
  e.out = TRUE,
  joint = FALSE,
  col.treated = "black",
  col.synth = "mediumblue",
  label.xy = NULL,
  plot.range = NULL,
  x.ticks = NULL,
  event.label = NULL,
  plot.specs = NULL,
  save.data = NULL
)
```

Arguments

<code>result</code>	a class <code>'scpi_est'</code> object, obtained by calling <code>scest</code> , or a class <code>'scpi_pi'</code> object, obtained by calling <code>scpi</code>
<code>fig.path</code>	a string indicating the path where the plot(s) should be saved.
<code>fig.name</code>	a string indicating the name of the plot(s). If multiple plots will be saved the command automatically generates a numeric suffix to avoid overwriting them.
<code>fig.format</code>	a string indicating the format in which the plot(s) should be saved.

<code>e.out</code>	a logical specifying whether out-of-sample uncertainty should be included in the plot(s).
<code>joint</code>	a logical specifying whether simultaneous prediction intervals should be included in the plot(s). It requires <code>e.out = TRUE</code> .
<code>col.treated</code>	a string specifying the color for the treated unit series. Find the full list at http://sape.inf.usi.ch/quick-reference/ggplot2/colour .
<code>col.synth</code>	a string specifying the color for the synthetic unit series. Find the full list at http://sape.inf.usi.ch/quick-reference/ggplot2/colour .
<code>label.xy</code>	a character list with two elements indicating the name of the axes (eg. <code>label.xy = list(x.lab = "Year", y.lab = "GDP growth (%)")</code>).
<code>plot.range</code>	a numeric array indicating the time range of the plot(s).
<code>x.ticks</code>	a numeric list containing the location of the ticks on the x axis.
<code>event.label</code>	a list containing a character object ('lab') indicating the label of the event and a numeric object indicating the height of the label in the plot.
<code>plot.specs</code>	a list containing some specifics to be passed to <code>ggsave</code> (eg. <code>img.width</code> , <code>img.height</code> , <code>dpi</code>)
<code>save.data</code>	a character specifying the name and the path of the saved dataframe containing the processed data used to produce the plot.

Value

<code>plots</code>	a list containing standard ggplot object(s) that can be used for further customization.
--------------------	---

Author(s)

Matias Cattaneo, Princeton University. <cattaneo@princeton.edu>.

Yingjie Feng, Tsinghua University. <fengyj@sem.tsinghua.edu.cn>.

Filippo Palomba, Princeton University (maintainer). <fpalomba@princeton.edu>.

Rocio Titiunik, Princeton University. <titiunik@princeton.edu>.

References

- [Abadie, A. \(2021\)](#). Using synthetic controls: Feasibility, data requirements, and methodological aspects. *Journal of Economic Literature*, 59(2), 391-425.
- [Cattaneo, M. D., Feng, Y., and Titiunik, R. \(2021\)](#). Prediction intervals for synthetic control methods. *Journal of the American Statistical Association*, 116(536), 1865-1880.
- [Cattaneo, M. D., Feng, Y., Palomba F., and Titiunik, R. \(2022\)](#), `scpi`: Uncertainty Quantification for Synthetic Control Estimators, *arXiv:2202.05984*.

See Also

[scdata](#), [scdataMulti](#), [scest](#), [scpi](#), [scplotMulti](#)

Examples

```
data <- scpi_germany

df <- scdata(df = data, id.var = "country", time.var = "year",
            outcome.var = "gdp", period.pre = (1960:1990),
            period.post = (1991:2003), unit.tr = "West Germany",
            unit.co = unique(data$country)[-7], constant = TRUE,
            cointegrated.data = TRUE)

result <- scest(df, w.constr = list(name = "simplex", Q = 1))

scplot(result)
```

scplotMulti	<i>Plot Synthetic Control Point Estimates and Prediction Interval with Multiple Treated units and Staggered Adoption</i>
-------------	--

Description

The command produces a wide range of plots of Synthetic Control estimates and corresponding prediction intervals. The command allows for multiple treated units and staggered adoption. Prediction intervals can take into account either in-sample uncertainty only or in-sample and out-of-sample uncertainty using the techniques developed in [Cattaneo, Feng, and Titiunik \(2021\)](#). [scpi](#). The input object should come from the command [scest](#) or from the command [scpi](#).

Companion [Stata](#) and [Python](#) packages are described in [Cattaneo, Feng, Palomba, and Titiunik \(2022\)](#).

Companion commands are: [scdata](#) and [scdataMulti](#) for data preparation in the single and multiple treated unit(s) cases, respectively, [scest](#) for point estimation, [scpi](#) for inference procedures, and [scplotMulti](#) for plots with multiple treated units.

Related Stata, R, and Python packages useful for inference in SC designs are described in the following website:

<https://nppackages.github.io/scpi/>

For an introduction to synthetic control methods, see [Abadie \(2021\)](#) and references therein.

Usage

```
scplotMulti(
  result,
  type = "series",
  e.out = TRUE,
  joint = FALSE,
  col.treated = "black",
  col.synth = "mediumblue",
  scales = "fixed",
  point.size = 1.5,
  ncols = 3,
  save.data = NULL,
  verbose = TRUE
)
```

Arguments

result	a class 'scpi_est' object, obtained by calling scest , or a class 'scpi_pi' object, obtained by calling scpi . The data object given as input to this command has to be processed with scdataMulti .
type	a character that specifies the type of plot to be produced. If set to 'treatment' then treatment effects are plotted. If set to 'series' (default), the actual and synthetic time series are reported.
e.out	a logical specifying whether out-of-sample uncertainty should be included in the plot(s).
joint	a logical specifying whether simultaneous prediction intervals should be included in the plot(s). It requires e.out = TRUE.
col.treated	a string specifying the color for the treated unit series. Find the full list at http://sape.inf.usi.ch/quick-reference/ggplot2/colour .
col.synth	a string specifying the color for the synthetic unit series. Find the full list at http://sape.inf.usi.ch/quick-reference/ggplot2/colour .
scales	should axes scales be fixed ("fixed", the default), free ("free"), or free in one dimension ("free_x", "free_y")?
point.size	a scalar controlling the size of points in the scatter plot. Default is 1.5.
ncols	an integer controlling the number of columns in the plot.
save.data	a character specifying the name and the path of the saved dataframe containing the processed data used to produce the plot.
verbose	if TRUE prints additional information in the console.

Value

plots	a list containing standard ggplot object(s) that can be used for further customization.
-------	---

Author(s)

Matias Cattaneo, Princeton University. <cattaneo@princeton.edu>.

Yingjie Feng, Tsinghua University. <fengyj@sem.tsinghua.edu.cn>.

Filippo Palomba, Princeton University (maintainer). <fpalomba@princeton.edu>.

Rocio Titiunik, Princeton University. <titiunik@princeton.edu>.

References

- [Abadie, A. \(2021\)](#). Using synthetic controls: Feasibility, data requirements, and methodological aspects. *Journal of Economic Literature*, 59(2), 391-425.
- [Cattaneo, M. D., Feng, Y., and Titiunik, R. \(2021\)](#). Prediction intervals for synthetic control methods. *Journal of the American Statistical Association*, 116(536), 1865-1880.
- [Cattaneo, M. D., Feng, Y., Palomba F., and Titiunik, R. \(2022\)](#), scpi: Uncertainty Quantification for Synthetic Control Estimators, *arXiv:2202.05984*.

See Also

[scdata](#), [scdataMulti](#), [scest](#), [scpi](#), [scplotMulti](#)

Examples

```
datager <- scpi_germany

datager$tr_id <- 0
datager$tr_id[(datager$country == "West Germany" & datager$year > 1990)] <- 1
datager$tr_id[(datager$country == "Italy" & datager$year > 1992)] <- 0

outcome.var <- "gdp"
id.var <- "country"
treatment.var <- "tr_id"
time.var <- "year"
df.unit <- scdataMulti(datager, id.var = id.var, outcome.var = outcome.var,
                      treatment.var = treatment.var,
                      time.var = time.var, features = list(c("gdp", "trade")),
                      cointegrated.data = TRUE, constant = TRUE)

res.unit <- scpi(df.unit, sims = 10, cores = 1)
scplotMulti(res.unit, joint = TRUE)
```

summary.scest

Summary Method for Synthetic Control Estimation

Description

The summary method for synthetic control estimation objects.

Usage

```
## S3 method for class 'scest'
summary(object, ...)
```

Arguments

object	Class "scpi_scest" or "scpi_scest_multi" object, obtained by calling scest .
...	Additional arguments

Value

No return value, called to summarize [scest](#) results.

Author(s)

Matias Cattaneo, Princeton University. <cattaneo@princeton.edu>.
 Yingjie Feng, Tsinghua University. <fengyj@sem.tsinghua.edu.cn>.
 Filippo Palomba, Princeton University (maintainer). <fpalomba@princeton.edu>.
 Rocio Titiunik, Princeton University. <titiunik@princeton.edu>.

See Also

[scest](#)

Supported methods: [print.scest](#), [summary.scest](#).

summary.scp*i**Summary Method for Synthetic Control Inference*

Description

The summary method for synthetic control inference objects.

Usage

```
## S3 method for class 'scpi'  
summary(object, ...)
```

Arguments

object	Class "scpi" object, obtained by calling scpi .
...	Additional arguments

Value

No return value, called to summarize [scpi](#) results.

Author(s)

Matias Cattaneo, Princeton University. <cattaneo@princeton.edu>.
Yingjie Feng, Tsinghua University. <fengyj@sem.tsinghua.edu.cn>.
Filippo Palomba, Princeton University (maintainer). <fpalomba@princeton.edu>.
Rocio Titiunik, Princeton University. <titiunik@princeton.edu>.

See Also

[scpi](#)
Supported methods: [print.scp*i*](#), [summary.scp*i*](#).

Index

* datasets

scpi_germany, [23](#)

_PACKAGE (scpi-package), [2](#)

as.Date, [5](#), [8](#)

CVXR, [12](#), [18](#)

nloptr, [12](#), [18](#)

print.scest, [3](#), [3](#), [28](#)

print.scpi, [3](#), [4](#), [29](#)

scdata, [2](#), [4](#), [7](#), [9–12](#), [14–16](#), [20](#), [23–27](#)

scdataMulti, [2](#), [4](#), [7](#), [7](#), [11](#), [12](#), [15](#), [23–27](#)

scest, [2–4](#), [7](#), [10](#), [11](#), [15](#), [21](#), [23–28](#)

scpi, [2–4](#), [7](#), [10](#), [11](#), [15](#), [15](#), [24–27](#), [29](#)

scpi-package, [2](#)

scpi_germany, [23](#)

scplot, [2](#), [4](#), [7](#), [10–12](#), [15](#), [18](#), [23](#), [24](#)

scplotMulti, [2](#), [4](#), [7](#), [10](#), [11](#), [15](#), [23–26](#), [26](#), [27](#)

summary.scest, [3](#), [28](#), [28](#)

summary.scpi, [4](#), [29](#), [29](#)