Package ‘TSCI’

October 9, 2023

Title Tools for Causal Inference with Possibly Invalid Instrumental Variables

Version 3.0.4


URL https://github.com/dlcarl/TSCI

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.2.3

Depends R (>= 4.0.0)

Suggests fda, MASS, testthat (>= 3.0.0), withr

Config/testthat/edition 3

Imports xgboost, Rfast, stats, ranger, parallel, fastDummies

NeedsCompilation no

Author David Carl [aut, cre] (<https://orcid.org/0000-0003-0615-0133>), Corinne Emmenegger [aut] (<https://orcid.org/0000-0003-0353-8888>), Wei Yuan [aut], Mengchu Zheng [aut], Zijian Guo [aut] (<https://orcid.org/0000-0002-2888-7016>)

Maintainer David Carl <david.carl@phd.unibocconi.it>

Repository CRAN

Date/Publication 2023-10-09 18:50:02 UTC
R topics documented:

- coef.tsci ................................................................. 2
- confint.tsci ......................................................... 3
- create_interactions ............................................... 3
- create_monomials .................................................. 4
- print.summary.tsci .................................................. 5
- print.tsci ............................................................. 5
- summary.tsci ........................................................ 6
- tsci_boosting ......................................................... 7
- tsci_forest ........................................................... 13
- tsci_poly ............................................................. 19
- tsci_secondstage ..................................................... 24

Index 29

---

 coef.tsci  

*Extract Model Coefficients of TSCI Fits.*

**Description**

Extract Model Coefficients of TSCI Fits.

**Usage**

```r
## S3 method for class 'tsci'
coef(object, parm = NULL, ...)
```

**Arguments**

- `object`  
an object of class 'tsci'.
- `parm`  
a specification for which treatment effect estimates should be returned. Either a vector of numbers or a vector of names or 'all'. If missing, the treatment effect estimate by violation space selection is returned. If 'all', the treatment effect estimates for all violation space candidates are returned.
- `...`  
arguments to be passed to or from other methods.

**Value**

Coefficients extracted form the model object object.
Description
Confidence Intervals of Treatment Effect Estimates for TSCI Fits.

Usage
### S3 method for class 'tsci'

confint(object, parm = NULL, level = 0.95, ...)

Arguments

- **object**: an object of class 'tsci'.
- **parm**: a specification of the parameters for which confidence intervals should be calculated. Either a vector of numbers or a vector of names or 'all'. If missing, the confidence interval of treatment effect estimate by violation space selection is returned. If 'all', the confidence intervals for all violation space candidates are returned.
- **level**: the confidence level required.
- **...**: additional argument(s) for methods.

Value

a matrix containing the confidence intervals.

create_interactions

Interactions as Violation Space Candidates

Description

Interactions as Violation Space Candidates

Usage

create_interactions(Z, X = NULL)

Arguments

- **Z**: observations of the instrumental variable(s). Either a numeric vector of length n or a numeric matrix with dimension n by s.
- **X**: observations of baseline covariate(s) for which interactions with the instrumental variable(s) should be part of the violation space candidates. Either a numeric vector of length n or a numeric matrix with dimension n by p or NULL (if only interactions between the instrumental variables itself should be part of the violation space candidates).
create_monomials

Value

A list. The first element contains the observations of the instrumental variable(s) Z. The second element contains all interactions between the instrumental variable(s) and the baseline covariate(s) X.

Examples

```r
Z <- matrix(rnorm(100 * 3), nrow = 100, ncol = 3)
X <- matrix(rnorm(100 * 3), nrow = 100, ncol = 3)
vio_space <- create_interactions(Z = Z, X = X)
```

create_monomials Monomials as Violation Space Candidates

Description

Monomials as Violation Space Candidates

Usage

```r
create_monomials(Z, degree, type = c("monomials_main", "monomials_full"))
```

Arguments

- **Z**: observations of the instrumental variable(s). Either a numeric vector of length n or a numeric matrix with dimension n by s.
- **degree**: The degree up to which monomials should be created. Either a single positive integer or a vector of length s containing positive integers.
- **type**: One out of monomials_main or monomials_full.

- monomials_main creates the monomials for the polynomials of each instrumental variable up to degree degree.
- monomials_full creates the monomials for the polynomials of a combination of all instrumental variables up to degree degree. Default is monomials_full.

Details

assuming there are 3 instrumental variables Z1, Z2, and Z3 and degree = c(d1, d2, d3) with d1 < d2 < d3, monomials_main creates the monomials of the polynomials (Z1 + 1)^d1, (Z2 + 1)^d2, (Z3 + 1)^d3 without the constants and monomials_full creates the monomials (Z1 + Z2 + Z3), (Z1 + Z2 + Z3)^2, ..., (Z1 + Z2 + Z3)^d3 without the constants and excluding monomials that are products of Z1^d or Z2^d with d > d1 resp. d > d2. Thus type = monomials_main does not include interactions between the instrumental variables.

Value

A list. Each element is a matrix consisting of the monomials to be added to the next violation space candidate.
print.summary.tsci

Examples

Z <- matrix(rnorm(100 * 3), nrow = 100, ncol = 3)
vio_space <- create_monomials(Z = Z, degree = 4, type = "monomials_full")

print.summary.tsci

Print Content of summary.tsci Object.

Description

Print Content of summary.tsci Object.

Usage

## S3 method for class 'summary.tsci'
print(x, digits = max(3, getOption("digits") - 3), ...)

Arguments

x an object of class 'summary.tsci'.
digits number of significant digits to display.
... arguments to be passed to or from other methods.

print.tsci

Print Content of tsci Object.

Description

Print Content of tsci Object.

Usage

## S3 method for class 'tsci'
print(x, digits = max(3, getOption("digits") - 3), ...)

Arguments

x an object of class 'tsci'.
digits number of significant digits to display.
... arguments to be passed to or from other methods.
Summarizing Two Stage Curvature Identification Fits

## S3 method for class 'tsci'
summary(object, extended_output = FALSE, ...)

### Arguments

- **object**: an object of class 'tsci'.
- **extended_output**: logical. If TRUE are more detailed summary is returned.
- **...**: arguments to be passed to or from other methods.

### Value

An object of class 'summary.tsci' containing the following elements:

- **coefficient**: a data frame with columns for the estimated treatment coefficient, its standard error, confidence interval and (two-sided) p-value.
- **invalidity**: a vector containing the number of times the instrumental variable(s) were considered valid, invalid or too weak to perform the test.
- **viospace_selection**: a data frame with columns for the number of times each of the violation space candidate was selected by comparison, the conservative method and as the largest violation space candidate for which the instrumental variable was considered to be strong enough.
- **treatment_model**: a data frame with information about the method used to fit the treatment model.
- **sample_size_A1**: the number of observations in the subset used to fit the outcome model.
- **sample_size_A2**: the number of observations in the subset used to train the parameters for fitting the treatment model.
- **n_splits**: the number of sample splits performed.
- **mult_split_method**: the method used to calculate the standard errors and p-values if n_splits is larger than 1.
- **alpha**: the significance level used.
- **iv_strength**: a data frame with columns containing the estimated instrumental variable strength and the estimated instrumental variable strength threshold for each violation space candidate. Will only be returned if extended_output is true.
- **coefficients_all**: a data frame with columns for the estimated treatment coefficients, its standard errors, confidence intervals and (two-sided) p-values for each violation space candidate.
**tsci_boosting**  

*Two Stage Curvature Identification with Boosting*

**Description**

`tsci_boosting` implements Two Stage Curvature Identification (Guo and Buehlmann 2022) with boosting. Through a data-dependent way, it tests for the smallest sufficiently large violation space among a pre-specified sequence of nested violation space candidates. Point and uncertainty estimates of the treatment effect for all violation space candidates including the selected violation space will be returned amongst other relevant statistics.

**Usage**

```r
  tsci_boosting(
    Y, D, Z, X = NULL, W = X, vio_space, create_nested_sequence = TRUE, sel_method = c("comparison", "conservative"), split_prop = 2/3, nrounds = 50, eta = 0.3, max_depth = c(1:6), subsample = 1, colsample_bytree = 1, early_stopping = TRUE, nfolds = 5, self_predict = FALSE, sd_boot = TRUE, iv_threshold = 10, threshold_boot = TRUE, alpha = 0.05, intercept = TRUE, parallel = c("no", "multicore", "snow"), nsplits = 10, mult_split_method = c("FWER", "DML"), ncores = 1, cl = NULL, raw_output = NULL, B = 300
  )
```

Arguments

Y observations of the outcome variable. Either a numeric vector of length n or a numeric matrix with dimension n by 1. If outcome variable is binary use dummy encoding.

D observations of the treatment variable. Either a numeric vector of length n or a numeric matrix with dimension n by 1. If treatment variable is binary use dummy encoding.

Z observations of the instrumental variable(s). Either a vector of length n or a matrix with dimension n by s. If observations are not numeric dummy encoding will be applied.

X observations of baseline covariate(s). Either a vector of length n or a matrix with dimension n by p or NULL (if no covariates should be included). If observations are not numeric dummy encoding will be applied.

W (transformed) observations of baseline covariate(s) used to fit the outcome model. Either a vector of length n or a matrix with dimension n by p_w or NULL (if no covariates should be included). If observations are not numeric dummy encoding will be applied.

vio_space list with vectors of length n and/or matrices with n rows as elements to specify the violation space candidates. If observations are not numeric dummy encoding will be applied. See Details for more information.

create_nested_sequence logical. If TRUE, the violation space candidates (in form of matrices) are defined sequentially starting with an empty violation matrix and subsequently adding the next element of vio_space to the current violation matrix. If FALSE, the violation space candidates (in form of matrices) are defined as the empty space and the elements of vio_space. See Details for more information.

sel_method The selection method used to estimate the treatment effect. Either "comparison" or "conservative". See Details.

split_prop proportion of observations used to fit the outcome model. Has to be a value in (0, 1).

nrounds number of boosting iterations. Can either be a single integer value or a vector of integer values to try.

eta learning rate of the boosting algorithm. Can either be a single numeric value or a vector of numeric values to try.

max_depth maximal tree depth. Can either be a single integer value or a vector of integer values to try.

subsample subsample ratio of the training instance. Can either be a single numeric value or a vector of numeric values to try. Has to be a numeric value in (0, 1].

colsample_bytree subsample ratio of columns when constructing each tree. Can either be a single numeric value or a vector of numeric values to try. Has to be a numeric value in (0, 1].

early_stopping logical. If TRUE, early stopping will be applied to choose the optimal number of boosting iteration using the cross-validation mean squared error.
tsci_boosting

nfolds a positive integer value specifying the number of folds used for cross-validation to choose best parameter combination.

self_predict logical. If FALSE, it sets the diagonal of the hat matrix of each tree to zero to avoid self-prediction and rescales the off-diagonal elements accordingly.

sd_boot logical. If TRUE, it determines the standard error using a bootstrap approach.

iv_threshold a numeric value specifying the minimum of the threshold of IV strength test.

threshold_boot logical. If TRUE, it determines the threshold of the IV strength using a bootstrap approach. If FALSE, it does not perform a bootstrap. See Details.

alpha the significance level. Has to be a numeric value between 0 and 1.

intercept logical. If TRUE, an intercept is included in the outcome model.

parallel one out of "no", "multicore", or "snow" specifying the parallelization method used.

nsplits number of times the data will be split. Has to be an integer larger or equal 1. See Details.

mult_split_method method to calculate the standard errors, p-values and to construct the confidence intervals if multi-splitting is performed. Default is "DML" if nsplits == 1 and "FWER" otherwise. See Details.

ncores the number of cores to use. Has to be an integer value larger or equal 1.

cl either a parallel or snow cluster or NULL.

raw_output logical. If TRUE, the coefficient and standard error estimates of each split will be returned. This is only needed for the use of the function confint if mult_split_method equals "FWER". Default is TRUE if mult_split_method is TRUE and FALSE otherwise.

B number of bootstrap samples. Has to be a positive integer value. Bootstrap methods are used to calculate the IV strength threshold if threshold_boot is TRUE and for the violation space selection.

Details

The treatment and outcome models are assumed to be of the following forms:

\[ D_i = f(Z_i, X_i) + \delta_i \]
\[ Y_i = \beta \cdot D_i + h(Z_i, X_i) + \phi(X_i) + \epsilon_i \]

where \( f(Z_i, X_i) \) is estimated using L2 boosting with regression trees as base learners, \( h(Z_i, X_i) \) is approximated using the violation space candidates and \( \phi(X_i) \) is approximated by a linear combination of the columns in \( W \). The errors are allowed to be heteroscedastic. To avoid overfitting bias the data is randomly split into two subsets \( A1 \) and \( A2 \) where the proportion of observations in the two sets is specified by split_prop. \( A2 \) is used to train the random forest and \( A1 \) is used to perform violation space selection and to estimate the treatment effect.

The package xgboost is used for boosting. If any of nrounds, eta, max_depth, subsample or colsample_bytree has more than one value, the best parameter combination is chosen by minimizing the cross-validation mean squared error.
The violation space candidates should be in a nested sequence as the violation space selection is performed by comparing the treatment estimate obtained by each violation space candidate with the estimates of all violation space candidates further down the list that provide enough IV strength. Only if no significant difference was found in all of those comparisons, the violation space candidate will be selected. If \( \text{sel\_method} \) is 'comparison', the treatment effect estimate of this violation space candidate will be returned. If \( \text{sel\_method} \) is 'conservative', the treatment effect estimate of the successive violation space candidate will be returned provided that the IV strength is large enough. The specification of suitable violation space candidates is a crucial step because a poor approximation of \( g(Z_i, X_i) \) might not address the bias caused by the violation of the IV assumption sufficiently well. The function \text{create\_monomials} \) can be used to create a predefined sequence of violation space candidates consisting of monomials. The function \text{create\_interactions} \) can be used to create a predefined sequence of violation space candidates consisting of two-way interactions between the instruments themselves and between the instruments and baseline covariates.

\( W \) should be chosen to be flexible enough to approximate the functional form of \( \phi \) well as otherwise the treatment estimator might be biased.

The instrumental variable(s) are considered strong enough for violation space candidate \( V_q \) if the estimated IV strength using this violation space candidate is larger than the obtained value of the threshold of the IV strength. The formula of the threshold of the IV strength has the form 
\[
\min\{\max\{2 \cdot \text{Trace}[M(V_q)], \text{iv\_threshold}\} + S(V_q), 40\} \quad \text{if \( \text{threshold\_boot} \) is TRUE, and}
\]
\[
\min\{\max\{2 \cdot \text{Trace}[M(V_q)], \text{iv\_threshold}\}, 40\} \quad \text{if \( \text{threshold\_boot} \) is FALSE.}
\]

The matrix \( M(V_q) \) depends on the hat matrix obtained from estimating \( f(Z_i, X_i) \), the violation space candidate \( V_q \) and the variables to include in the outcome model \( W \). \( S(V_q) \) is obtained using a bootstrap and aims to adjust for the estimation error of the IV strength. Usually, the value of the threshold of the IV strength obtained using the bootstrap approach is larger. Thus, using \( \text{threshold\_boot} \) equals \text{TRUE} leads to a more conservative IV strength test. For more information see subsection 3.3 in Guo and Buehlmann (2022).

\( \text{nsplits} \) specifies the number of data splits that should be performed. For each data split the output statistics such as the point estimates of the treatment effect are calculated. Those statistics will then be aggregated over the different data splits. It is recommended to perform multiple data splits as data splitting introduces additional randomness. By aggregating the results of multiple data splits, the effects of this randomness can be decreased. If \( \text{nsplits} \) is larger than 1, point estimates are aggregated by medians. Standard errors, p-values and confidence intervals are obtained by the method specified by the parameter \( \text{mult\_split\_method} \). 'DML' uses the approach by Chernozhukov et al. (2018). 'FWER' uses the approach by Meinshausen et al. (2009) and controls for the family-wise error rate. 'FWER' does not provide standard errors. For large sample sizes, a large values for \( \text{nsplits} \) can lead to a high running time as for each split a new hat matrix must be calculated.

There are three possibilities to set the argument \( \text{parallel} \), namely "no" for serial evaluation (default), "multicore" for parallel evaluation using forking, and "snow" for parallel evaluation using a parallel socket cluster. It is recommended to select \text{RNGkind} ("L'Ecuyer-CMRG") and to set a seed to ensure that the parallel computing of the package \text{TSCI} is reproducible. This ensures that each processor receives a different substream of the pseudo random number generator stream. Thus, the results are reproducible if the arguments (including \( \text{ncores} \)) remain unchanged. There is an op-
tational argument cl to specify a custom cluster if parallel = "snow".

See also Carl et al. (2023) for more details.

Value

A list containing the following elements:

- **Coef_all**: a series of point estimates of the treatment effect obtained by the different violation space candidates.
- **sd_all**: standard errors of the estimates of the treatment effect obtained by the different violation space candidates.
- **pval_all**: p-values of the treatment effect estimates obtained by the different violation space candidates.
- **CI_all**: confidence intervals for the treatment effect obtained by the different violation space candidates.
- **Coef_sel**: the point estimator of the treatment effect obtained by the selected violation space candidate(s).
- **sd_sel**: the standard error of Coef_sel.
- **pval_sel**: p-value of the treatment effect estimate obtained by the selected violation space candidate(s).
- **CI_sel**: confidence interval for the treatment effect obtained by the selected violation space candidate(s).
- **iv_str**: IV strength using the different violation space candidates.
- **iv_thol**: the threshold for the IV strength using the different violation space candidates.
- **Qmax**: the frequency each violation space candidate was the largest violation space candidate for which the IV strength was considered large enough determined by the IV strength test over the multiple data splits. If 0, the IV Strength test failed for the first violation space candidate. Otherwise, violation space selection was performed.
- **q_comp**: the frequency each violation space candidate was selected by the comparison method over the multiple data splits.
- **q_cons**: the frequency each violation space candidate was selected by the conservative method over the multiple data splits.
- **invalidity**: the frequency the instrumental variable(s) were considered valid, invalid or too weak to test for violations. The instrumental variables are considered too weak to test for violations if the IV strength is already too weak using the first violation space candidate (besides the empty violation space). Testing for violations is always performed by using the comparison method.
- **mse**: the out-of-sample mean squared error of the fitted treatment model.
- **FirstStage_model**: the method used to fit the treatment model.
- **n_A1**: number of observations in A1.
- **n_A2**: number of observations in A2.
- **n_splits**: number of data splits performed.
- **mult_split_method**: the method used to calculate the standard errors and p-values.
- **alpha**: the significance level used.
References


See Also

tsci_forest for TSCI with random forest.

tsci_poly for TSCI with polynomial basis expansion.

tsci_secondstage for TSCI with user provided hat matrix.

Examples

```r
### a small example without baseline covariates
if (require("MASS")) {
  # sample size
  n <- 100
  # the IV strength
  a <- 1
  # the violation strength
  tau <- 1
  # true effect
  beta <- 1
  # treatment model
  f <- function(x) {1 + a * (x + x^2)}
  # outcome model
  g <- function(x) {1 + tau * x}

  # generate data
  mu_error <- rep(0, 2)
  Cov_error <- matrix(c(1, 0.5, 0.5, 1), 2, 2)
  Error <- MASS::mvrnorm(n, mu_error, Cov_error)
  # instrumental variable
  Z <- rnorm(n)
  # treatment variable
  D <- f(Z) + Error[, 1]
  # outcome variable
  Y <- beta * D + g(Z) + Error[, 2]
}
```
```
# Two Stage L2 Boosting
# create violation space candidates
vio_space <- create_monomials(Z, 2, "monomials_main")
# perform two stage curvature identification
output_BO <- tsci_boosting(Y, D, Z, vio_space = vio_space, nsplits = 1,
                          max_depth = 2, nrounds = 10, B = 100)
summary(output_BO)
```

### tsci_forest

*Two Stage Curvature Identification with Random Forests*

#### Description

`tsci_forest` implements Two Stage Curvature Identification (Guo and Buehlmann 2022) with random forests. Through a data-dependent way, it tests for the smallest sufficiently large violation space among a pre-specified sequence of nested violation space candidates. Point and uncertainty estimates of the treatment effect for all violation space candidates including the selected violation space will be returned amongst other relevant statistics.

#### Usage

```r
tsci_forest(
    Y, D, Z,
    X = NULL, W = X, vio_space, create_nested_sequence = TRUE,
    sel_method = c("comparison", "conservative"),
    split_prop = 2/3, num_trees = 200, mtry = NULL,
    max_depth = 0, min_node_size = c(5, 10, 20),
    self_predict = FALSE, sd_boot = TRUE, iv_threshold = 10,
    threshold_boot = TRUE, alpha = 0.05, nsplits = 10,
    mult_split_method = c("FWER", "DML"),
    intercept = TRUE, parallel = c("no", "multicore", "snow"),
    ncores = 1, cl = NULL,
```

```
raw_output = NULL,
B = 300
)

Arguments

Y observations of the outcome variable. Either a numeric vector of length n or a numeric matrix with dimension n by 1. If outcome variable is binary use dummy encoding.

D observations of the treatment variable. Either a numeric vector of length n or a numeric matrix with dimension n by 1. If treatment variable is binary use dummy encoding.

Z observations of the instrumental variable(s). Either a vector of length n or a matrix with dimension n by s. If observations are not numeric dummy encoding will be applied.

X observations of baseline covariate(s). Either a vector of length n or a matrix with dimension n by p or NULL (if no covariates should be included). If observations are not numeric dummy encoding will be applied.

W (transformed) observations of baseline covariate(s) used to fit the outcome model. Either a vector of length n or a matrix with dimension n by p_w or NULL (if no covariates should be included). If observations are not numeric dummy encoding will be applied.

vio_space list with vectors of length n and/or matrices with n rows as elements to specify the violation space candidates. If observations are not numeric dummy encoding will be applied. See Details for more information.

create_nested_sequence logical. If TRUE, the violation space candidates (in form of matrices) are defined sequentially starting with an empty violation matrix and subsequently adding the next element of vio_space to the current violation matrix. If FALSE, the violation space candidates (in form of matrices) are defined as the empty space and the elements of vio_space. See Details for more information.

sel_method the selection method used to estimate the treatment effect. Either "comparison" or "conservative". See Details.

split_prop proportion of observations used to fit the outcome model. Has to be a numeric value in (0, 1).

num_trees number of trees in random forests. Can either be a single integer value or a vector of integer values to try.

mtry number of covariates to possibly split at in each node of the tree of the random forest. Can either be a single integer value or a vector of integer values to try. Can also be a list of single argument function(s) returning an integer value, given the number of independent variables. The values have to be positive integers not larger than the number of independent variables in the treatment model. Default is to try all integer values between one-third of the independent variables and two-thirds of the independent variables.

max_depth maximal tree depth in random forests. Can either be a single integer value or a vector of integer values to try. 0 correspond to unlimited depth.
min_node_size

minimal size of each leaf node in the random forest. Can either be a single integer value or a vector of integer values to try.

self_predict

logical. If FALSE, it sets the diagonal of the hat matrix of each tree to zero to avoid self-prediction and rescales the off-diagonal elements accordingly.

sd_boot

logical. If TRUE, it determines the standard error using a bootstrap approach.

iv_threshold

a numeric value specifying the minimum of the threshold of IV strength test.

threshold_boot

logical. If TRUE, it determines the threshold of the IV strength using a bootstrap approach. If FALSE, it does not perform a bootstrap. See Details.

alpha

the significance level. Has to be a numeric value between 0 and 1.

nsplits

number of times the data will be split. Has to be an integer larger or equal 1. See Details.

mult_split_method

method to calculate the standard errors, p-values and to construct the confidence intervals if multi-splitting is performed. Default is "DML" if nsplits == 1 and "FWER" otherwise. See Details.

intercept

logical. If TRUE, an intercept is included in the outcome model.

parallel

one out of "no", "multicore", or "snow" specifying the parallelization method used. See Details.

ncores

the number of cores to use. Has to be an integer value larger or equal 1.

cl

either a parallel or snow cluster or NULL.

raw_output

logical. If TRUE, the coefficient and standard error estimates of each split will be returned. This is only needed for the use of the function confint if mult_split_method equals "FWER". Default is TRUE if mult_split_method is TRUE and FALSE otherwise.

B

number of bootstrap samples. Has to be a positive integer value. Bootstrap methods are used to calculate the IV strength threshold if threshold_boot is TRUE and for the violation space selection. It is also used to calculate the standard error if sd_boot is TRUE.

Details

The treatment and outcome models are assumed to be of the following forms:

\[ D_i = f(Z_i, X_i) + \delta_i \]

\[ Y_i = \beta \cdot D_i + h(Z_i, X_i) + \phi(X_i) + \epsilon_i \]

where \( f(Z_i, X_i) \) is estimated using a random forest, \( h(Z_i, X_i) \) is approximated using the violation space candidates and \( \phi(X_i) \) is approximated by a linear combination of the columns in \( W \). The errors are allowed to be heteroscedastic. To avoid overfitting bias the data is randomly split into two subsets \( A_1 \) and \( A_2 \) where the proportion of observations in the two sets is specified by split_prop. \( A_2 \) is used to train the random forest and \( A_1 \) is used to perform violation space selection and to estimate the treatment effect.

The package ranger is used to fit the random forest. If any of num_trees, max_depth or min_node_size has more than one value, the best parameter combination is chosen by minimizing the out-of-bag
mean squared error.

The violation space candidates should be in a nested sequence as the violation space selection is performed by comparing the treatment estimate obtained by each violation space candidate with the estimates of all violation space candidates further down the list. The violation space candidate will be selected. If `sel_method` is 'comparison', the treatment effect estimate of this violation space candidate will be returned. If `sel_method` is 'conservative', the treatment effect estimate of the successive violation space candidate will be returned provided that the IV strength is large enough. The specification of suitable violation space candidates is a crucial step because a poor approximation of \( g(Z_i, X_i) \) might not address the bias caused by the violation of the IV assumption sufficiently well. The function `create_monomials` can be used to create a predefined sequence of violation space candidates consisting of monomials. The function `create_interactions` can be used to create a predefined sequence of violation space candidates consisting of two-way interactions between the instruments themselves and between the instruments and baseline covariates.

The instrumental variable(s) are considered strong enough for violation space candidate \( V_q \) if the estimated IV strength using this violation space candidate is larger than the obtained value of the threshold of the IV strength. The formula of the threshold of the IV strength has the form \( \min\{\max\{2 \cdot \text{Trace}[M(V_q)], iv\_threshold\} + S(V_q), 40\} \) if `threshold_boot` is `TRUE`, and \( \min\{\max\{2 \cdot \text{Trace}[M(V_q)], iv\_threshold\}, 40\} \) if `threshold_boot` is `FALSE`. The matrix \( M(V_q) \) depends on the hat matrix obtained from estimating \( f(Z_i, X_i) \), the violation space candidate \( V_q \) and the variables to include in the outcome model \( W \). \( S(V_q) \) is obtained using a bootstrap and aims to adjust for the estimation error of the IV strength. Usually, the value of the threshold of the IV strength obtained using the bootstrap approach is larger. Thus, using `threshold_boot` equals `TRUE` leads to a more conservative IV strength test. For more information see subsection 3.3 in Guo and Buehlmann (2022).

`nsplits` specifies the number of data splits that should be performed. For each data split the output statistics such as the point estimates of the treatment effect are calculated. Those statistics will then be aggregated over the different data splits. It is recommended to perform multiple data splits as data splitting introduces additional randomness. By aggregating the results of multiple data splits, the effects of this randomness can be decreased. If `nsplits` is larger than 1, point estimates are aggregated by medians. Standard errors, p-values and confidence intervals are obtained by the method specified by the parameter `mult_split_method`. 'DML' uses the approach by Chernozhukov et al. (2018). 'FWER' uses the approach by Meinshausen et al. (2009) and controls for the family-wise error rate. 'FWER' does not provide standard errors. For large sample sizes, a large values for `nsplits` can lead to a high running time as for each split a new hat matrix must be calculated.

There are three possibilities to set the argument `parallel`, namely "no" for serial evaluation (default), "multicore" for parallel evaluation using forking, and "snow" for parallel evaluation using a parallel socket cluster. It is recommended to select `RNGkind` ("L’Ecuyer-CMRG") and to set a seed to ensure that the parallel computing of the package `TSCI` is reproducible. This ensures that each processor receives a different substream of the pseudo random number generator stream. Thus, the results are reproducible if the arguments (including `ncores`) remain unchanged. There is an optional argument `cl` to specify a custom cluster if `parallel` = "snow".
Results obtained on different operating systems might differ even when the same seed is set. The reason for this lies in the way the random forest algorithm in ranger is implemented. Currently, we are not aware of a solution to ensure reproducibility across operating systems when using tsci_forest. However, tsci_boosting, tsci_poly and tsci_secondstage do not have this issue.

See also Carl et al. (2023) for more details.

Value

A list containing the following elements:

- Coef_all a series of point estimates of the treatment effect obtained by the different violation space candidates.
- sd_all standard errors of the estimates of the treatment effect obtained by the different violation space candidates.
- pval_all p-values of the treatment effect estimates obtained by the different violation space candidates.
- CI_all confidence intervals for the treatment effect obtained by the different violation space candidates.
- Coef_sel the point estimator of the treatment effect obtained by the selected violation space candidate(s).
- sd_sel the standard error of Coef_sel.
- pval_sel p-value of the treatment effect estimate obtained by the selected violation space candidate(s).
- CI_sel confidence interval for the treatment effect obtained by the selected violation space candidate(s).
- iv_str IV strength using the different violation space candidates.
- iv_thol the threshold for the IV strength using the different violation space candidates.
- Qmax the frequency each violation space candidate was the largest violation space candidate for which the IV strength was considered large enough determined by the IV strength test over the multiple data splits. If 0, the IV Strength test failed for the first violation space candidate. Otherwise, violation space selection was performed.
- q_comp the frequency each violation space candidate was selected by the comparison method over the multiple data splits.
- q_cons the frequency each violation space candidate was selected by the conservative method over the multiple data splits.
- invalidity the frequency the instrumental variable(s) were considered valid, invalid or too weak to test for violations. The instrumental variables are considered too weak to test for violations if the IV strength is already too weak using the first violation space candidate (besides the empty violation space). Testing for violations is always performed by using the comparison method.
- mse the out-of-sample mean squared error of the fitted treatment model.
- FirstStage_model the method used to fit the treatment model.
- n_A1 number of observations in A1.
n_A2 number of observations in A2.
nsplits number of data splits performed.
mult_split_method the method used to calculate the standard errors and p-values.
alpha the significance level used.

References


See Also

tsci_boosting for TSCI with boosting.
tsci_poly for TSCI with polynomial basis expansion.
tsci_secondstage for TSCI with user provided hat matrix.

Examples

```r
### a small example without baseline covariates
if (require("MASS")) {
  # sample size
  n <- 100
  # the IV strength
  a <- 1
  # the violation strength
  tau <- 1
  # true effect
  beta <- 1
  # treatment model
  f <- function(x) {1 + a * (x + x^2)}
  # outcome model
  g <- function(x) {1 + tau * x}

  # generate data
  mu_error <- rep(0, 2)
  Cov_error <- matrix(c(1, 0.5, 0.5, 1), 2, 2)
```
```r
Error <- MASS::mvrnorm(n, mu_error, Cov_error)
# instrumental variable
Z <- rnorm(n)
# treatment variable
D <- f(Z) + Error[, 1]
# outcome variable
Y <- beta * D + g(Z) + Error[, 2]

# Two Stage Random Forest
# create violation space candidates
vio_space <- create_monomials(Z, 2, "monomials_main")
# perform two stage curvature identification
output_RF <- tsci_forest(Y, D, Z, vio_space = vio_space, nsplits = 1,
                        num_trees = 50, B = 100)
summary(output_RF)
```

---

**tsci_poly**  
*Two Stage Curvature Identification with Polynomial Basis Expansion*

**Description**

tsci_poly implements Two Stage Curvature Identification (Guo and Buehlmann 2022) with a basis expansion by monomials. Through a data-dependent way it tests for the smallest sufficiently large violation space among a pre-specified sequence of nested violation space candidates. Point and uncertainty estimates of the treatment effect for all violation space candidates including the selected violation space will be returned amongst other relevant statistics.

**Usage**

tsci_poly(  
  Y,  
  D,  
  Z,  
  X = NULL,  
  W = X,  
  vio_space = NULL,  
  create_nested_sequence = TRUE,  
  sel_method = c("comparison", "conservative"),  
  min_order = 1,  
  max_order = 10,  
  exact_order = NULL,  
  order_selection_method = c("grid search", "backfitting"),  
  max_iter = 100,  
  conv_tol = 10^-6,  
  gcv = FALSE,  
  nfolds = 5,  
  sd_boot = TRUE,
iv_threshold = 10,
threshold_boot = TRUE,
alpha = 0.05,
intercept = TRUE,
B = 300
)

Arguments

Y observations of the outcome variable. Either a numeric vector of length n or a
umeric matrix with dimension n by 1. If outcome variable is binary use dummy
encoding.

D observations of the treatment variable. Either a numeric vector of length n or
a numeric matrix with dimension n by 1. If treatment variable is binary use
dummy encoding.

Z observations of the instrumental variable(s). Either a vector of length n or a
matrix with dimension n by s. If observations are not numeric dummy encoding
will be applied.

X observations of baseline covariate(s). Either a vector of length n or a matrix with
dimension n by p or NULL (if no covariates should be included). If observations
are not numeric dummy encoding will be applied.

W (transformed) observations of baseline covariate(s) used to fit the outcome model.
Either a vector of length n or a matrix with dimension n by p_w or NULL (if no
covariates should be included). If observations are not numeric dummy encoding
will be applied.

vio_space either NULL or a list with numeric vectors of length n and/or numeric matrices
with n rows as elements to specify the violation space candidates. If observa-
tions are not numeric dummy encoding will be applied. See Details for more
information. If NULL, then the violation space candidates are chosen to be a
nested sequence of monomials with degree depending on the orders of the poly-
nomials used to fit the treatment model.

create_nested_sequence logical. If TRUE, the violation space candidates (in form of matrices) are defined
sequentially starting with an empty violation matrix and subsequently adding
the next element of vio_space to the current violation matrix. If FALSE, the
violation space candidates (in form of matrices) are defined as the empty space
and the elements of vio_space. See Details for more information.

sel_method The selection method used to estimate the treatment effect. Either "comparison"
or "conservative". See Details.

min_order either a single integer value or a vector of integer values of length s specifying
the smallest order of polynomials to use in the selection of the treatment model.
If a single integer value is provided, the polynomials of all instrumental variables
use this value.

max_order either a single integer value or a vector of integer values of length s specifying
the largest order of polynomials to use in the selection of the treatment model. If
a single integer value is provided, the polynomials of all instrumental variables
use this value.
exact_order  
either a single integer value or a vector of integer values of length s specifying 
the exact order of polynomials to use in the treatment model. If a single integer 
value is provided, the polynomials of all instrumental variables use this value.

order_selection_method  
method used to select the best fitting order of polynomials for the treatment 
model. Must be either 'grid search' or 'backfitting'. 'grid search' can be very slow 
if the number of instruments is large.

max_iter  
number of iterations used in the backfitting algorithm if order_selection_method 
is 'backfitting'. Has to be a positive integer value.

conv_tol  
tolerance of convergence in the backfitting algorithm if order_selection_method 
is 'backfitting'.

gcv  
logical. If TRUE, the generalized cross-validation mean squared error is used 
to determine the best fitting order of polynomials for the treatment model. If FALSE, k-fold cross-validation is used instead.

nfolds  
number of folds used for the k-fold cross-validation if gcv is FALSE. Has to be a 
positive integer value.

sd_boot  
logical. if TRUE, it determines the standard error using a bootstrap approach.

iv_threshold  
a numeric value specifying the minimum of the threshold of IV strength test.

threshold_boot  
logical. if TRUE, it determines the threshold of the IV strength using a bootstrap 
approach. If FALSE, it does not perform a bootstrap. See Details.

alpha  
the significance level. Has to be a numeric value between 0 and 1.

intercept  
logical. If TRUE, an intercept is included in the outcome model.

B  
number of bootstrap samples. Has to be a positive integer value. Bootstrap 
methods are used to calculate the iv strength threshold if threshold_boot is 
TRUE and for the violation space selection.

Details

The treatment and outcome models are assumed to be of the following forms:

\[ D_i = f(Z_i, X_i) + \delta_i \]
\[ Y_i = \beta \cdot D_i + h(Z_i, X_i) + \phi(X_i) + \epsilon_i \]

where \( f(Z_i, X_i) \) is estimated using a polynomial basis expansion of the instrumental variables and
a linear combination of the baseline covariates, \( h(Z_i, X_i) \) is approximated using the violation space candidates and \( \phi(X_i) \) is approximated by a linear combination of the columns in \( W \). The errors are
allowed to be heteroscedastic.

The violation space candidates should be in a nested sequence as the violation space selection
is performed by comparing the treatment estimate obtained by each violation space candidate with
the estimates of all violation space candidates further down the list \text{vio_space} that provide enough
IV strength. Only if no significant difference was found in all of those comparisons, the violation
space candidate will be selected. If \text{sel_method} is 'comparison', the treatment effect estimate of
this violation space candidate will be returned. If \text{sel_method} is 'conservative', the treatment effect
estimate of the successive violation space candidate will be returned provided that the IV strength
is large enough. If `vio_space` is NULL the violation space candidates are chosen to be a nested sequence of polynomials of the instrumental variables up to the degrees used to fit the treatment model. This guarantees that the possible spaces of the violation will be tested. If the functional form of the outcome model is not well-known it is advisable to use the default values for \( W \) and `vio_space`.

The instrumental variable(s) are considered strong enough for violation space candidate \( V_q \) if the estimated IV strength using this violation space candidate is larger than the obtained value of the threshold of the IV strength. The formula of the threshold of the IV strength has the form

\[
\min\{\max\{2 \cdot \text{Trace}[M(V_q)], \text{iv}_\text{threshold}\} + S(V_q), 40\} \quad \text{if } \text{threshold\_boot} \text{ is TRUE, and } \\
\min\{\max\{2 \cdot \text{Trace}[M(V_q)], \text{iv}_\text{threshold}\}, 40\} \quad \text{if } \text{threshold\_boot} \text{ is FALSE.}
\]

The matrix \( M(V_q) \) depends on the hat matrix obtained from estimating \( f(Z_i, X_i) \), the violation space candidate \( V_q \) and the variables to include in the outcome model \( \mathcal{W} \). \( S(V_q) \) is obtained using a bootstrap and aims to adjust for the estimation error of the IV strength. Usually, the value of the threshold of the IV strength obtained using the bootstrap approach is larger. Thus, using `threshold\_boot` equals `TRUE` leads to a more conservative IV strength test. For more information see subsection 3.3 in Guo and Buehlmann (2022).

See also Carl et al. (2023) for more details.

**Value**

A list containing the following elements:

- `Coef_all` a series of point estimates of the treatment effect obtained by the different violation space candidates.
- `sd_all` standard errors of the estimates of the treatment effect obtained by the different violation space candidates.
- `pval_all` p-values of the treatment effect estimates obtained by the different violation space candidates.
- `CI_all` confidence intervals for the treatment effect obtained by the different violation space candidates.
- `Coef_sel` the point estimator of the treatment effect obtained by the selected violation space candidate(s).
- `sd_sel` the standard error of `Coef_sel`.
- `pval_sel` p-value of the treatment effect estimate obtained by the selected violation space candidate(s).
- `CI_sel` confidence interval for the treatment effect obtained by the selected violation space candidate(s).
- `iv_str` IV strength using the different violation space candidates.
- `iv_thol` the threshold for the IV strength using the different violation space candidates.
- `Qmax` the violation space candidate that was the largest violation space candidate for which the IV strength was considered large enough determined by the IV strength test. If 0, the IV Strength test failed for the first violation space candidate. Otherwise, violation space selection was performed.
q_comp the violation space candidate that was selected by the comparison method over the multiple data splits.

q_cons the violation space candidate that was selected by the conservative method over the multiple data splits.

invalidity shows whether the instrumental variable(s) were considered valid, invalid or too weak to test for violations. The instrumental variables are considered too weak to test for violations if the IV strength is already too weak using the first violation space candidate (besides the empty violation space). Testing for violations is always performed by using the comparison method.

mse the out-of-sample mean squared error of the treatment model.

References


See Also

tsci_forest for TSCI with random forest.

tsci_boosting for TSCI with boosting.

tsci_secondstage for TSCI with user provided hat matrix.

Examples

```r
### a small example without baseline covariates
if (require("MASS")) {
  # sample size
  n <- 100
  # the IV strength
  a <- 1
  # the violation strength
  tau <- 1
  # true effect
  beta <- 1
  # treatment model
  f <- function(x) {1 + a * (x + x^2)}
  # outcome model
  g <- function(x) {1 + tau * x}

  # generate data
  mu_error <- rep(0, 2)
  Cov_error <- matrix(c(1, 0.5, 0.5, 1), 2, 2)
  Error <- MASS::mvrnorm(n, mu_error, Cov_error)
  # instrumental variable
```
```r
Z <- rnorm(n)
# treatment variable
D <- f(Z) + Error[, 1]
# outcome variable
Y <- beta * D + g(Z) + Error[, 2]

# Two Stage Polynomials
output_PO <- tsci_poly(Y, D, Z, max_order = 3, max_iter = 20, B = 100)
summary(output_PO)
```

---

**tsci_secondstage**

**Two Stage Curvature Identification with User Provided Hat Matrix**

**Description**

tsci_secondstage implements Two Stage Curvature Identification (Guo and Buehlmann 2022) for a user-provided hat matrix. Through a data-dependent way it tests for the smallest sufficiently large violation space among a pre-specified sequence of nested violation space candidates. Point and uncertainty estimates of the treatment effect for all violation space candidates including the selected violation space will be returned amongst other relevant statistics.

**Usage**

tsci_secondstage(
  Y,
  D,
  Z,
  W = NULL,
  vio_space,
  create_nested_sequence = TRUE,
  weight,
  A1_ind = NULL,
  sel_method = c("comparison", "conservative"),
  sd_boot = TRUE,
  iv_threshold = 10,
  threshold_boot = TRUE,
  alpha = 0.05,
  intercept = TRUE,
  B = 300
)

**Arguments**

- **Y** observations of the outcome variable. Either a numeric vector of length n or a numeric matrix with dimension n by 1. If outcome variable is binary use dummy encoding.
observations of the treatment variable. Either a numeric vector of length n or a numeric vector with dimension n by 1. If treatment variable is binary use dummy encoding.

\( Z \) observations of the instrumental variable(s). Either a vector of length n or a matrix with dimension n by s. If observations are not numeric dummy encoding will be applied.

\( W \) (transformed) observations of baseline covariate(s) used to fit the outcome model. Either a vector of length n or a matrix with dimension n by \( p_w \) or NULL (if no covariates should be included). If observations are not numeric dummy encoding will be applied.

\text{vio\_space} list with vectors of length n and/or matrices with n rows as elements to specify the violation space candidates. If observations are not numeric dummy encoding will be applied. See Details for more information.

\text{create\_nested\_sequence} logical. If TRUE, the violation space candidates (in form of matrices) are defined sequentially starting with an empty violation matrix and subsequently adding the next element of \text{vio\_space} to the current violation matrix. If FALSE, the violation space candidates (in form of matrices) are defined as the empty space and the elements of \text{vio\_space}. See Details for more information.

\text{weight} the hat matrix of the treatment model.

\text{A1\_ind} indices of the observations that will be used to fit the outcome model. Must be of same length as the number of rows and columns of \text{weight}. If NULL, all observations will be used.

\text{sel\_method} The selection method used to estimate the treatment effect. Either "comparison" or "conservative". See Details.

\text{sd\_boot} logical. If TRUE, it determines the standard error using a bootstrap approach.

\text{iv\_threshold} a numeric value specifying the minimum of the threshold of IV strength test.

\text{threshold\_boot} logical. If TRUE, it determines the threshold of the IV strength using a bootstrap approach. If FALSE, it does not perform a bootstrap. See Details.

\text{alpha} the significance level. Has to be a numeric value between 0 and 1.

\text{intercept} logical. If TRUE, an intercept is included in the outcome model.

\text{B} number of bootstrap samples. Has to be a positive integer value. Bootstrap methods are used to calculate the iv strength threshold if \text{threshold\_boot} is TRUE and for the violation space selection.

**Details**

The treatment and outcome models are assumed to be of the following forms:

\[
D_i = f(Z_i, X_i) + \delta_i \\
Y_i = \beta \cdot D_i + h(Z_i, X_i) + \phi(X_i) + \epsilon_i
\]

where \( f(Z_i, X_i) \) is estimated using a random forest, \( h(Z_i, X_i) \) is approximated using the hat matrix \( \text{weight} \) provided by the user and \( \phi(X_i) \) is approximated by a linear combination of the columns in \( W \). The errors are allowed to be heteroscedastic. \( A1 \) is used to perform violation space selection and
to estimate the treatment effect.

The violation space candidates should be in a nested sequence as the violation space selection is performed by comparing the treatment estimate obtained by each violation space candidate with the estimates of all violation space candidates further down the list. Only if no significant difference was found in all of those comparisons, the violation space candidate will be selected. If sel_method is 'comparison', the treatment effect estimate of this violation space candidate will be returned. If sel_method is 'conservative', the treatment effect estimate of the successive violation space candidate will be returned provided that the IV strength is large enough. The specification of suitable violation space candidates is a crucial step because a poor approximation of $g(Z_i, X_i)$ might not address the bias caused by the violation of the IV assumption sufficiently well. The function create_monomials can be used to create a predefined sequence of violation space candidates consisting of monomials. The function create_interactions can be used to create a predefined sequence of violation space candidates consisting of two-way interactions between the instruments themselves and between the instruments and the instruments and baseline covariates.

The instrumental variable(s) are considered strong enough for violation space candidate $V_q$ if the estimated IV strength using this violation space candidate is larger than the obtained value of the threshold of the IV strength. The formula of the threshold of the IV strength has the form $\min\{\max\{2 \cdot \text{Trace}[M(V_q)], \text{iv\_threshold}\} + S(V_q), 40\}$ if threshold_boot is TRUE, and $\min\{\max\{2 \cdot \text{Trace}[M(V_q)], \text{iv\_threshold}\}, 40\}$ if threshold_boot is FALSE. The matrix $M(V_q)$ depends on the hat matrix obtained from estimating $f(Z_i, X_i)$, the violation space candidate $V_q$ and the variables to include in the outcome model $W$. $S(V_q)$ is obtained using a bootstrap and aims to adjust for the estimation error of the IV strength. Usually, the value of the threshold of the IV strength obtained using the bootstrap approach is larger. Thus, using threshold_boot equals TRUE leads to a more conservative IV strength test. For more information see subsection 3.3 in Guo and Buehlmann (2022).

See also Carl et al. (2023) for more details.

**Value**

A list containing the following elements:

- Coef_all a series of point estimates of the treatment effect obtained by the different violation space candidates.
- sd_all standard errors of the estimates of the treatment effect obtained by the different violation space candidates.
- pval_all p-values of the treatment effect estimates obtained by the different violation space candidates.
- CI_all confidence intervals for the treatment effect obtained by the different violation space candidates.
- Coef_sel the point estimator of the treatment effect obtained by the selected violation space candidate(s).
- sd_sel the standard error of Coef_sel.
- pval_sel p-value of the treatment effect estimate obtained by the selected violation space candidate(s).
CI_sel confidence interval for the treatment effect obtained by the selected violation space candidate(s).

iv_str IV strength using the different violation space candidates.

iv_thol the threshold for the IV strength using the different violation space candidates.

Qmax the violation space candidate that was the largest violation space candidate for which the IV strength was considered large enough determined by the IV strength test. If 0, the IV Strength test failed for the first violation space candidate. Otherwise, violation space selection was performed.

q_comp the violation space candidate that was selected by the comparison method over the multiple data splits.

q_cons the violation space candidate that was selected by the conservative method over the multiple data splits.

invalidity shows whether the instrumental variable(s) were considered valid, invalid or too weak to test for violations. The instrumental variables are considered too weak to test for violations if the IV strength is already too weak using the first violation space candidate (besides the empty violation space). Testing for violations is always performed by using the comparison method.

References


See Also

tsci_boosting for TSCI with boosting.

tsci_forest for TSCI with random forest.

tsci_poly for TSCI with polynomial basis expansion.

Examples

```r
# a small example without baseline covariates
if (require("MASS")) {
  # sample size
```
n <- 100  # the IV strength
a <- 1     # the violation strength
tau <- 1   # true effect
beta <- 1 # treatment model
f <- function(x) {1 + a * (x + x^2)}  # outcome model
g <- function(x) {1 + tau * x}

# generate data
mu_error <- rep(0, 2)
Cov_error <- matrix(c(1, 0.5, 0.5, 1), 2, 2)
Error <- MASS::mvrnorm(n, mu_error, Cov_error)
Z <- rnorm(n)  # instrumental variable
D <- f(Z) + Error[, 1]  # treatment variable
Y <- beta * D + g(Z) + Error[, 2]  # outcome variable

# Two Stage User Defined
# get hat matrix of treatment model
A <- cbind(1, Z, Z^2, Z^3)
weight <- A %*% chol2inv(chol(t(A) %*% A)) %*% t(A)
# create violation space candidates
vio_space <- create_monomials(Z, 2, "monomials_main")
# perform two stage curvature identification
output_UD <- tsci_secondstage(Y, D, Z, vio_space = vio_space, weight = weight, B = 100)
summary(output_UD)
Index

coef.tsci, 2
confint.tsci, 3
create_interactions, 3, 10, 16, 26
create_monomials, 4, 10, 16, 26

print.summary.tsci, 5
print.tsci, 5

RNGkind, 10, 16

summary.tsci, 6

tsci_boosting, 7, 17, 18, 23, 27
tsci_forest, 12, 13, 23, 27
tsci_poly, 12, 17, 18, 19, 27
tsci_secondstage, 12, 17, 18, 23, 24