Package 'causalSLSE'

May 8, 2026

Version 0.4-1

Date 2025-08-26

Title Semiparametric Least Squares Inference for Causal Effects

Description Several causal effects are measured using least squares regressions and basis function approximations. Backward and forward selection methods based on different criteria are used to select the basis functions.

Depends R (>= 4.0.0)

Suggests knitr, rmarkdown

Collate 'cslse.R' 'slse.R' 'Fcslse.R' 'altCausal.R'

Imports stats, sandwich, texreg, methods

License GPL (>= 2)

NeedsCompilation yes

VignetteBuilder knitr

Author Pierre Chausse Developer [aut, cre],
Mihai Giurcanu Developer [aut]

Maintainer Pierre Chausse Developer <pchausse@uwaterloo.ca>

Repository CRAN

Date/Publication 2025-08-29 14:00:08 UTC

Contents

`altCausal` *Alternative Causal Effect estimation methods.*

Description

This documentation file presents a collection of popular methods used to estimate the average causal effect, the causal effect on the treated and the causal effect on the nontreated.

Usage

```
matching(form,  balm, data, type=c("ACE","ACT","ACN"), M=4,
         psForm=NULL, bcForm=NULL, vJ=4)

LLmatching(form, psForm, data, type=c("ACE","ACT","ACN"),
           kern=c("Gaussian","Epanechnikov"),tol=1e-4,
           h=NULL, from=.00001, to=5, ngrid=10, maxit=100,
           hMethod=c("Brent","Grid"))

ipw(form, psForm, data, type=c("ACE","ACT","ACN"),
    normalized=FALSE, tolPS=0, ...)
```

Arguments

`form`	A formula that links the outcome variable to the treatment indicator. For the moment, only one treatment group is allowed.
`balm`	A formula or a matrix with balancing covariates to be matched.
`data`	A data.frame or a matrix with column names.
`type`	The type of causal effect to compute. `ACE` stands for average causal effect, `ACT` for causal effect on the treated and `ACN` for causal effect on the control or non-treated.
`M`	The minimum number of matches
`psForm`	It is the `formula` argument passed to `glm` to estimate the propensity scores by a logistic regression.

bcForm	A formula that represents the right hand side in the regression used for the bias correction.
kern	The type of kernel to use in the local linear regression method.
tol	The tolerance level for the stopping rule used to compute the optimal bandwidth.
tolPS	Observations for which the propensity scores are not between tolPS and (1-tolPS) are removed. It is meant to satisfy the assumption that the propensity score is bounded away from, 0 and 1. The default is 0, in which case observations with negative or greater than 1 propensity scores are dropped.
h	A fixed bandwidth. By default, the optimal bandwidth is found by minimizing a cross-validation.
from	The lower bound for the search of the optimal bandwidth.
to	The upper bound for the search of the optimal bandwidth.
ngrid	The number of grid points if the optimal bandwidth is obtained by grid search.
maxit	The maximum number of iterations for the minimization of the cross-validation.
hMethod	The method used to find the optimal bandwidth.
normalized	Should the weights be normalized. If set to GPE, the GPE method is used.
vJ	The minimum number of matches for the standard error estimation
...	Additional arguments to pass to optim when the GPE method is used.

Value

All methods return an object of classes "causalfit". It is a list with the following elements:

estim	The causal effect estimate.
se	The standard error estimate of the causal effect.
type	The type of causal effect. It is the value of the argument type.
method	A description of the method.
form	A list of formulas used for the estimation.
details	A more detailed description of the method.
info	Convergence information if relevant.

Examples

```
data(simDat3)

balm <-  ~ X1 * X2
g <- Y ~ Z
ps <- Z~X1*X2

## A print and summary methods exist for this class

fit1 <- matching(form=g, balm=balm, data=simDat3, type="ACE", bcForm=balm)
fit1

fit2 <- LLmatching(form=g, psForm=ps, data=simDat3, type="ACE")
```

```
fit2

fit3 <- ipw(form=g, psForm=ps, data=simDat3, type="ACE", normalized=TRUE)
fit3

## A print and summary methods exist for this class

summary(fit1)
```

`as.model` *Converter into Model Objects*

Description

When the information about a model is available, it reconstructs it and returns a valid model object.

Usage

```
## S3 method for class 'slseFit'
as.model(x, ...)

## S3 method for class 'cslseFit'
as.model(x, ...)

## S3 method for class 'cslse'
as.model(x, ...)
```

Arguments

`x`	An object containing the model to extract.
`...`	Other arguments to pass to other methods. Currently not used.

Value

The method returns an object of class `slseModel` or `cslseModel`.

Examples

```
data(simDat3)
mod <- cslseModel(Y ~ Z | ~ X1 * X2, data = simDat3)
fit <- estSLSE(mod)

## Extract the model from a cslseModel object

as.model(fit)

## Extract the model from a cslse object
```

```
cs <- causalSLSE(mod)
as.model(cs)
```

---

causalSLSE *Causal Effect Based on Semiparametric Least Squares Models*

Description

This is the main method to estimate the causal effects using the semiparametric least squares method. It returns an object of class cslse and is registered for objects of class cslseModel and cslseFit.

Usage

```
## S3 method for class 'cslseModel'
causalSLSE(object,
                              selType=c("SLSE","BLSE","FLSE"),
                              selCrit = c("AIC", "BIC", "PVT"),
                           selVcov = c("HC0", "Classical", "HC1", "HC2", "HC3"),
                              causal = c("ALL","ACT","ACE","ACN"),
                              pvalT = function(p) 1/log(p),
                              vcov.=vcovHC, reSelect=FALSE, ...)
## S3 method for class 'cslseFit'
causalSLSE(object, causal = c("ALL","ACT","ACE","ACN"),
                             vcov.=vcovHC, ...)
## S3 method for class 'formula'
causalSLSE(object, data, nbasis=function(n) n^0.3,
                             knots,
                             selType=c("SLSE","BLSE","FLSE"),
                             selCrit = c("AIC", "BIC", "PVT"),
                           selVcov = c("HC0", "Classical", "HC1", "HC2", "HC3"),
                             causal = c("ALL","ACT","ACE","ACN"),
                             pvalT = function(p) 1/log(p),
                             vcov.=vcovHC, reSelect=FALSE, ...)
```

Arguments

object	An object of class cslseModel created by the cslseModel function, cslseFit created by estSLSE or formula.
data	A data.frame with all variables included in form.
nbasis	A function to determined the number of basis functions. It has to be a function of one argument, the sample size.
knots	A list of knots for the treated and nontreated groups. The list must be named using the group names. Each element of the list is also a list of length equal to the number of confounders. The choice for each confounders is NULL for no knots or numeric for specific values. If missing, the knots are automatically generated.

selType	The method for selecting the knots. By default (SLSE), all knots from the model are used.
selCrit	The criterion to select the knots.
causal	What causal effect should we compute.
pvalT	A function to determine the p-value threshold for the significance of the coefficients. It has to be a function of one parameter, which is the average number of knots in the model. This value may differ across treatment group.
selVcov	The type of least squares covariance matrix used to compute the p-values needed for the selection.
vcov.	An alternative function to compute the covariance matrix of the least squares estimators. The default is vcovHC. This function is used to compute the standard errors of the causal effect estimators and the SLSE coefficients.
reSelect	By default, the stored selections are used. If reSelect is set to TRUE, the selection is re-computed.
...	Additional arguments to pass to vcov.

Value

It returns an object of class cslse, which inherits from the class cslseFit. It is a list with the following elements:

treated, nontreated	They are objects of class slseFit obtained by estSLSE.
ACE, ACT, ACN	Estimates of the average causal effect, the causal effect on the treated and the causal effect on the nontreated. Each of them is a vector of two elements: the estimate and its estimated standard error. All three are included only if the argument causal is set to "ALL".

Also, the object contains the following additional attributes:

treatedVar	The name of the variable in the dataset that represents the treatment indicator.
groupInd	A named vector with the value of the treatment indicator corresponding do each treatment group.

See Also

estSLSE for the estimation of the model, slseKnots for the format of knots, and selSLSE and update for the knots selection and to understand how stored selections are used.

Examples

```
data(simDat3)

## A causal SLSE model with the outcome Y
## the treatment indicator Z and the confounders X1, X2 and X1:X2

mod1 <- cslseModel(Y ~ Z | ~ X1 * X2, data = simDat3)
```

```
## The causal effects are estimated using the backward method and AIC criterion
## The HC1 type is used for the least squares covariance matrix

fit1 <- causalSLSE(mod1, selType = "BLSE", type = "HC1")

## This is the same for formula objects

fit2 <- causalSLSE(Y ~ Z | ~ X1 * X2, data = simDat3, selType = "BLSE", type = "HC1")
```

`cslseModel` *Semiparametric Least Squares Estimator Model*

Description

This function creates an object of class `cslseModel` or `slseModel`. The former is a list of `slseModel` objects, one for each treatment group. The model objects contain all specifications about the model being estimated including the starting knots used to construct the basis functions.

Usage

```
cslseModel(form, data, nbasis = function(n) n^0.3,
           knots, groupInd=c(treated=1, nontreated=0))
slseModel(form, data, nbasis = function(n) n^0.3,
           knots)
```

Arguments

`form`	A formula for the regression in the case of a simple spline regression, or two formulas separated by `|`, one for the outcome versus the treatment indicator and one of the confounders. See the details and examples below.
`data`	A `data.frame` that contains all variables included in `form`.
`nbasis`	A function to determined the number of basis functions. It has to be a function of one argument, the sample size.
`knots`	An optional list of knots. For `cslseModel`, it must be a named list using the group names (treated and nontreated). Each element of the list is also a list of length equal to the number of covariates. The choice for each covariate is NULL for no knots or a numeric vector for specific knots. If missing, the knots are automatically generated.
`groupInd`	A named vector with the group names and values. By default, the treatment indicator defined in `form` is equal to 1 for the treated and 0 for the nontreated. For now, these are the only allowed names, but the values can differ. In particular, the treatment indicator may be a factor with character values. See the example below.

Details

An object of class `slseModel` is a regression model with a dependent variable and a set of covariates. We assume that the model can be written as $Y = f(X) + e$, where $f(x)$ is an unknown function of the covariates that we approximate using linear combinations of basis functions. For now, we only consider local linear splines defined by sets of knots, one for each covariate generated by the `model.matrix` of `form`. The knots are automatically determined unless specified by the user.

An object of class `cslseModel` is a list of `slseModel` objects, one for each treatment group. The assignment to a group is determined by a treatment indicator, which is the right-hand side variable in the formula to the left of `|` specified in `form`. The formula to the right of `|` is used by `model.matrix` to generate a set of confounders for each treatment group. See the example below and the vignette to more details.

Value

The function `slseModel` creates a semiparametric least squares model or a class `slseModel` object. It is a list with the following elements:

`formX`	The right-hand side of the `form` argument expressed as character. It can be converted into a `formula` using `reformulate`.
`nameY`	The name of the variable representing the outcome
`data`	The dataset passed to the function `slseModel` with the missing values, if any, omitted.
`knots`	An object of class `slseKnots`. It is the set of knots that define the local linear splines.
`nameS`	The prefix added to the covariate names when the basis functions are generated. By default, it is equal to `"U."`
`xlevels`	When relevant, a list of levels of the factors included in the model.
`na`	A vector of integer representing the observations omitted because of missing values. It is `NULL` when there are no missing values.

Note that other elements may be present if a knots selection has been applied to the model.

The function `cslseModel` creates an object of class `cslseModel`. It is a list of `slseModel` objects, one for each treatment group. It also contains the following additional attributes:

`treatedVar`	The name of the variable in the dataset that represents the treatment indicator.
`groupInd`	A named vector with the value of the treatment indicator corresponding do each treatment group.

See Also

`selSLSE` for additional elements that can be included in the `slseModel` object, `estSLSE` to see how the model is estimated and `slseKnots` for the format of the knots

Examples

```
data(simDat3)

## Y is the outcome and Z the treatment indicator
## The confounders are X1, X2 and X1:X2:

mod1 <- cslseModel(Y ~ Z | ~ X1 * X2, data = simDat3)
mod1

## A simple SLSE for a regression model

mod2 <- slseModel(Y ~ X1 * X2, data = simDat3)
mod2

## If the treatment indicator differs from 0-1

data(simDat4)
mod3 <- cslseModel(Y ~ treat | ~ X1 * X2, data = simDat4,
                   groupInd = c(treated = "treat", nontreated = "notreat" ))
mod3
```

`estSLSE` *Least Squares Estimate of* `cslseModel` *or* `slseModel` *Objects*

Description

This is the main function to estimate `cslseModel` or `slseModel` objects. It generates the basis functions based on the knots specified in the model and estimates it by least squares.

Usage

```
## S3 method for class 'cslseModel'
estSLSE(model, selKnots, ...)

## S3 method for class 'slseModel'
estSLSE(model, selKnots, ...)
```

Arguments

`model`	A model of class `cslseModel` or `slseModel`.
`selKnots`	An optional list of integers to select the knots from the list of knots specified by the model. If the model is a `cslseModel` object, it must be a named list with the names being either "treated", "nontreated" or both. By default, the knots from the model stored in the element `knots` are used.
`...`	Additional arguments to pass to other methods. Currently not used.

Details

The method for `slseModel` objects generates the matrix of basis functions implied by the set of knots included in the model and estimate the model by the least squares. Let Y be the outcome and U be the matrix of basis functions. Then, the function estimates the model using the code `lm(Y~U)`.

For `cslseModel`, we could estimate the model using `lm(Y~Z+I(Z-1)+I(U0*(1-Z))+I(U1*Z))`, where `Z` is a binary variable equal to 1 for the treated and 0 for the nontreated, and `U0` and `U1` are the matrices of basis functions for the nontreated and treated, but the model is estimated separately for each group. Therefore, the function `estSLSE.cslseModel` calls the function `estSLSE.slseModel` for each `slseModel` objects included in the `cslseModel` object.

Value

It returns an object of class `slseFit` or `cslseFit` depending on which method is called. An object of class `slseFit` is a list with the following elements:

`LSE`	This is the least squares estimate of the semiparametric model. It is an object of class `lm`.
`model`	An object of class `slseModel`. It is the model that is being estimated. The model may have been created by `slseModel` or modified by `selSLSE` or `update`.

An object of class `cslseFit` is a list of `slseFit` objects, one for each treatment group. It also contains the following additional attributes:

`treatedVar`	The name of the variable in the dataset that represents the treatment indicator.
`groupInd`	A named vector with the value of the treatment indicator corresponding do each treatment group.

Examples

```
data(simDat3)

## Estimating a causal semiparametric model

mod1 <- cslseModel(Y ~ Z | ~ X1 * X2, data = simDat3)
fit1 <- estSLSE(mod1)

## Estimating a semiparametric model

mod2 <- slseModel(Y ~ X1 * X2, data = simDat3)
fit2 <- estSLSE(mod2)
```

extract `extract` *methods for some objects.*

Description

`extract` method for `slseFit` objects created by the `estSLSE` function.

`extract` method for `cslse` objects created by the `causalSLSE` function.

`extract` method for `altCausal` objects created by the functions in the list of alternative methods: `altCausal`.

Usage

```
## S4 method for signature 'slseFit'
extract(
  model,
  include.rsquared = TRUE,
  include.adjrs = TRUE,
  include.nobs = TRUE,
  include.fstatistic = FALSE,
  include.rmse = FALSE,
  ...)

## S4 method for signature 'cslse'
extract(
    model,
    include.nobs = TRUE,
    include.nknots = TRUE,
    include.numcov = TRUE,
    include.rsquared = TRUE,
    include.adjrs=TRUE,
    separated.rsquared = FALSE,
    which = c("ALL","ACE","ACT","ACN","ACE-ACT","ACE-ACN","ACT-ACN"),
    ...)

## S4 method for signature 'altCausal'
extract(
    model,
    include.nobs = TRUE,
    ...)
```

Arguments

`model`	A model object.
`include.nobs`	Report the number of obervations?
`include.nknots`	Report the total number of knots for each group?

include.numcov Report the total number of covariates (including interactions is any) per group?

include.rsquared
Report the R-squared of the final regression estimation

include.adjrs Report the adjusted R-squared of the final regression estimtion

separated.rsquared
Should we print the R-squared separately for each group? This applies as well to the adjusted R-squared.

which Which causal effect measures should be printed?

include.fstatistic
Report the F-statistics?

include.rmse Report the RMSE?

... Custom parameters, which are handed over to subroutines. Currently not in use.

Value

It returns an object of class texreg.

---

llSplines *Local Linear Splines Generator for Model Objects*

Description

It generates a matrix of basis functions using local linear splines. The number of basis functions and the breaking points are determined by the knot specifications of the slseModel or cslseModel model.

Usage

```
## S3 method for class 'slseModel'
llSplines(object, ...)
## S3 method for class 'cslseModel'
llSplines(object, ...)
```

Arguments

object A model of class slseModel or cslseModel.

... Additional arguments to pass to other methods. Currently not used.

Value

The function returns a matrix of basis functions used to estimate the semiparametric model. It is used directly as regressor in lm as shown in the example below.

Examples

```
data(simDat3)

## We manually estimate the semiparametric model

mod1 <- slseModel(Y ~ X1 * X2, data = simDat3)
U <- llSplines(mod1)
fit1 <- lm(Y ~ U, data = simDat3)

## We use estSLSE instead (results are identical)

fit2 <- estSLSE(mod1)
```

---

nsw *Lalonde Subsample of the National Supported Work Demonstration Data (NSW)*

Description

This data was collected to evaluate the National Supported Work (NSW) Demonstration project in Lalonde (1986).

Usage

```
data(nsw)
```

Format

A data frame containing 9 variables.

treat Treatment assignment

age Age

ed Years of Education

black 1 if Black, 0 otherwise

hisp 1 if Hispanic 0 otherwise

married 1 if married 0 otherwise

nodeg 1 if no college degree 0 otherwise

re75 1975 earnings

re78 1978 earnings

Details

The dataset was obtained from the ATE package (see reference).

Source

"NSW Data Files" from Rajeev Dehejia's website. URL: http://users.nber.org/~rdehejia/data/.nswdata2.html

"National Supported Work Evaluation Study, 1975-1979: Public Use Files." from the Interuniversity Consortium for Political and Social Research. URL: http://www.icpsr.umich.edu/icpsrweb/ICPSR/studies/7865

References

Lalonde, R. (1986). "Evaluating the Econometric Evaluations of Training Programs," American Economic Review, 76(4), 604-620.

Dehejia R. and Wahba S. (1999). "Causal Effects in Non-Experimental Studies: Re-Evaluating the Evaluation of Training Programs," JASA 94 (448), 1053-1062.

Asad Haris and Gary Chan (2015). ATE: Inference for Average Treatment Effects using Covariate Balancing. R package version 0.2.0. https://CRAN.R-project.org/package=ATE

`plot` *Plot of Predicted Outcome*

Description

For objects of class `slseFit`, the method plots the predicted outcome with respect to a given covariate. It is the same for objects of class `cslseFit`, but the predicted outcome is plotted for each treatment group separately.

Usage

```
## S3 method for class 'cslseFit'
plot(x, y, which = y, interval = c("none", "confidence"),
                       level = 0.95, fixedCov = list(),
                       vcov. = vcovHC,  add = FALSE, addToLegend = NULL,
                       addPoints = FALSE, FUN = mean,
                       plot=TRUE, graphPar=list(), ...)
## S3 method for class 'slseFit'
plot(x, y, which = y, interval = c("none", "confidence"),
                      level = 0.95, fixedCov = NULL,
                      vcov. = vcovHC,  add = FALSE,
                      addPoints = FALSE, FUN = mean,
                      plot=TRUE, graphPar=list(), ...)
```

Arguments

`x`	Object of class `cslseFit` or `slseFit` created by `estSLSE`.
`y`	alias for `which` for compatibility with `plot`.
`which`	Which covariate to plot against the outcome variable. It could be an integer or a character.

interval	The type of confidence interval. The default is none.
level	The confidence interval level if included.
fixedCov	List of covariates to fix to specific values for the nontreated and treated groups. By default, covariates not selected by which are set to their group specific sample means.
vcov.	An alternative function to compute the covariance matrix of the least squares estimates. The default is the vcovHC method for lm objects.
add	Should the plot be added to an existing one?
addToLegend	A character string to add to the legend next to treated and control.
addPoints	Should we add the scatterplot of the outcome and covariate on the graph?
FUN	The function to determine the fixed value for the covariates not fixed by fixedCov. The default is mean.
plot	If set to FALSE, a data.frame is returned for each group with the covariate selected by which and the prediction.
graphPar	A list of graphical parameters. See Details.
...	Additional argument to pass to the vcov. function.

Details

The default set of parameters can be obtained by running the command causalSLSE:::.initPar(). It returns a list of four elements: treated for the parameter of the lines or points of the treated, nontreated for the parameters of the nontreated, common for the common parameters not specific to a group like the main title or the axis labels, and legend for the legend parameters. The elements treated and nontreated are lists of two elements: points and lines, which are lists of graphical parameters for the scatterplot (when addPoints is TRUE) and the lines. The argument graphPar can be use to modify existing parameters or to add new ones. It must respect the structure of the initial list. See the example below.

To fix covariates to the same values for both groups, fixedCov is just a named list with values for the covariates associated with the names. To fix the covariates to different values for the treated and nontreated, fixedCov is a named list of at least 1 element with names being either treated, nontreated or both. If only one group is specified, the covariates for the other groups are determined by FUN.

Value

It returns NULL invisibly if the argument plot is TRUE. Otherwise, it returns a data.frame with the necessary variables to manually create the plot. For slseModel objects, it is a list with the following elements:

Outcome	The outcome variable from the model. The name of this element is the actual name of the outcome variable in the model.
which	The covariate selected by the argument which. The name of this element is the name of the selected covariate.
fit	The predicted outcome from the model fit.

`lower`, `upper` The lower and upper bounds of the confidence interval. It is only available when the argument `interval` is set to `"confidence"`.

Note that all returned variables are ordered with respect to the selected covariates. See the example below.

For `cslseModel`, the above list of variables is returned separately for each treatment group.

Examples

```
data(simDat3)

##  For cslse objects

mod <- cslseModel(Y ~ Z | ~ X1 + X2, data = simDat3)
fit <- causalSLSE(mod)
plot(fit, "X1")

## Let's change the type of points for the treated and lines for the
## nontreated

gpar <- list(treated = list(points = list(pch = 24, col = 5)),
             nontreated = list(lines = list(lty = 5, col =
             "darkgreen")), common = list(xlab = "New X",  main =
             "Plot with changed parameters"))

plot(fit, "X1", addPoints = TRUE, graphPar = gpar)

## For slseModel objects:

mod2 <- slseModel(Y ~ X1 + X2, data = simDat3)
fit2 <- estSLSE(mod2)
plot(fit2, "X1", interval = "confidence", addPoints = TRUE)

##  The same graph produced manually

p2 <- plot(fit2, "X1", interval = "confidence", plot = FALSE)
plot(p2$X1, p2$Y, pch = 21, main = "Y against X1", xlab = "X1", ylab =
     "Y")
lines(p2$X1, p2$fit, lwd = 2)
lines(p2$X1, p2$lower, lty = 3, lwd = 2)
lines(p2$X1, p2$upper, lty = 3, lwd = 2)
```

`predict` *Outcome Prediction*

Description

The method computes the predicted outcome for each group with standard errors and confidence intervals.

Usage

```
## S3 method for class 'cslseFit'
predict(object, interval=c("none","confidence"),
                        se.fit=FALSE, newdata=NULL, level=0.95,
                        vcov.=vcovHC, ...)
## S3 method for class 'slseFit'
predict(object, interval=c("none","confidence"),
                        se.fit=FALSE, newdata=NULL, level=0.95,
                        vcov.=vcovHC, ...)
```

Arguments

object	Object of class cslseFit or slseFit created by estSLSE.
interval	If set to "confidence", it returns the predicted values along with the lower and upper bounds of the confidence interval.
se.fit	Should the function return the standard errors of the predicted values?
level	The confidence interval level if interval is set to "confidence".
newdata	A data.frame of new data. It must include values for all covariates, and for the treatment indicator in the case of cslseFit objects.
vcov.	An alternative function to compute the covariance matrix of the least squares estimates. The default is the vcovHC.
...	Additional argument to pass to the vcov. function.

Value

For slseFit objects, it returns the predicted outcome if se.fit is FALSE or a list of the following two elements otherwise:

fit	The predicted outcome.
se.fit	The standard errors of the predicted outcomes.

If the argument confidence is set to "interval", the predicted outcome is a matrix with the predicted outcome, and the lower and upper bounds of the confidence intervals.

For objects of class 'cslseFit', the same is returned for each treatment group in a list. The elements of the list are treated and nontreated (until the package allows for more than one treatment).

Examples

```
data(simDat3)
mod <- cslseModel(Y ~ Z | ~ X1 + X2, data = simDat3)
fit <- causalSLSE(mod)

## Predicting outcome for all observations

pr <- predict(fit, interval = "confidence")

## Predicting outcome with new data
```

```
ndat <- data.frame(X1 = c(-2, 1, 2, 3), X2 = c(-4, -2, 0, 1),
                   Z = c(1, 1, 0, 0))
predict(fit, newdata = ndat)
```

---

print *Print Methods*

Description

Print methods for different objects from the package.

Usage

```
## S3 method for class 'cslseModel'
print(x, which=c("Model", "selKnots", "Pvalues"),
                           digits = max(3L, getOption("digits") - 3L), ...)
## S3 method for class 'slseModel'
print(x, which=c("Model", "selKnots", "Pvalues"),
                           digits = max(3L, getOption("digits") - 3L), ...)
## S3 method for class 'cslse'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
## S3 method for class 'cslseFit'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
## S3 method for class 'slseFit'
print(x, digits = max(3L, getOption("digits") - 3L), ...)
## S3 method for class 'summary.cslse'
print(x, digits = max(3L, getOption("digits") - 3L),
                              signif.stars = getOption("show.signif.stars"),
                              beta=FALSE, knots = FALSE,...)
## S3 method for class 'summary.cslseFit'
print(x, groups, digits = max(3L, getOption("digits") - 3L),
                                signif.stars = getOption("show.signif.stars"),
                                ...)
## S3 method for class 'summary.slseFit'
print(x, digits = max(3L, getOption("digits") - 3L),
                                signif.stars = getOption("show.signif.stars"),
                                ...)
## S3 method for class 'slseKnots'
print(x, header=c("None", "All", "Select"),
                           digits = max(3L, getOption("digits") - 3L), ...)
```

Arguments

x	A model or a fit object.
digits	The number of digits to print.

signif.stars	Should we print the significant stars?
beta	Should we print the coefficient matrix of the basis functions?
knots	Should we print the set of knots?
which	What info we should print? It prints a summary of the model when set to "Model", the current knots when set to "selKnots" and the p-values and initial knots when set to "Pvalues". For the latter, it is only available when a knots selection method has been applied to the object.
header	What description of the object should be printed.
groups	The names of the group to be printed. By default, they are all printed.
...	Argument for other types of objects.

Value

All methods return NULL invisibly.

Examples

```
## For cslseModel objects

data(simDat3)
mod1 <- cslseModel(Y ~ Z | ~ X1 * X2, data = simDat3)
print(mod1)

## For cslse objects

causal <- causalSLSE(mod1)
print(causal)

## For summary.cslse objects

s <- summary(causal)
print(s)

## For cslseFit objects

fit <- estSLSE(mod1)
print(fit)

## For summary.cslseFit objects

summary(fit)
```

---

selSLSE *Knots Selection Method*

Description

This is the main function to select the knots in cslseModel or slseModel objects. It returns a model with an optimal set of knots.

Usage

```
## S3 method for class 'cslseModel'
selSLSE(model, selType=c("BLSE", "FLSE"),
        selCrit = c("AIC", "BIC", "PVT"),
        pvalT = function(p) 1/log(p),
        vcovType = c("HC0", "Classical", "HC1", "HC2", "HC3"),
        reSelect=FALSE, ...)
## S3 method for class 'slseModel'
selSLSE(model, selType=c("BLSE", "FLSE"),
        selCrit = c("AIC", "BIC", "PVT"),
        pvalT = function(p) 1/log(p),
        vcovType = c("HC0", "Classical", "HC1", "HC2", "HC3"),
        reSelect=FALSE, ...)
```

Arguments

`model`	A model of class `cslseModel` or `slseModel`.
`selType`	The selection method: backward (`"BLSE"`) or forward (`"FLSE"`).
`selCrit`	The criterion to select the piecewise polynomial knots.
`pvalT`	A function to determine the p-value threshold for the significance of the coefficients. It has to be a function of one parameter, which is the average number of knots in the model.
`vcovType`	The type of least squares covariance matrix used to compute the p-values needed for the selection.
`reSelect`	By default, the stored selections are used. If `reSelect` is set to `TRUE`, the selection is re-computed.
`...`	Additional arguments to pass to other methods. Currently not used.

Details

It selects the knots using one of the two methods, FLSE or BLSE, with either the AIC, BIC or a p-value threshold (see the vignette for more details). Any of these selection methods requires several least squares estimations and it is performed only if the method has not been applied yet and `reSelect` is set to `TRUE`. This is possible because any new knots selection is saved into the returned model. A model may have more than one selection saved into it. The active knots (the ones used when we estimate the model) is stored into the element `knots` of the model and the saved selections are stored into the element `selections`. See below for what is included in this element.

Note that the selections for the three criteria AIC, BIC and PVT are computed and saved automatically in the returned model when `selCrit` is set to either `"AIC"` or `"BIC"`, because it does not require many more operations to select them all once we do it for AIC or BIC. However, it is only computed for PVT when `selCrit` is set to `"PVT"`.

The knots are selected jointly for all treatment groups in `cslseModel` objects. However, the active knots and all saved selections are stored separately for each treatment group. For example, the active knots for the treated in the `cslseModel` object `mod` are stored in `mod$treated$knots`. See the Vignette for more details.

Value

The method returns an object of class slseModel or cslseModel depending on which object it is applied to. When it does not already exist, the element selections is added to the slseModel object (or to each slseModel object in cslseModel objects). The element selections is a list with one or more of the following elements:

originalKnots	The original knots as selected initially by slseModel or cslseModel. This element is always included.
FLSE, BLSE	This is where selections based on the forward (FLSE) and the backward (BLSE) methods are stored.

Finally, BLSE and FLSE are lists that may contain the following elements:

AIC, BIC	A list of integer vectors, one for each covariate in the slseModel model. The vectors of integers indicate which sets the original set of knots minimizes the AIC or the BIC.
PVT	Same as the AIC and BIC, but the selection is based on a p-value threshold.
JAIC, JBIC	This criteria is for cslseModel models. It is like the AIC and BIC, but it indicates that the AIC or BIC was computes by estimating the models of all treatment groups jointly. This selection is not the same as a selection group by group.
Threshold	The p-value threshold used for the PVT criterion.
pval	A list of p-values, one for each original knots. See vignette for a definition of the p-values.

See Also

slseModel and cslseModel for model objects description and update for ways of selecting stored selections

Examples

```
data(simDat3)
mod1 <- cslseModel(Y~Z|~X1*X2, data=simDat3)
mod1 <- selSLSE(mod1, selType="FLSE", selCrit="AIC")

## The following does not require additional computation
## because the selection is stored in mod1

mod1 <- selSLSE(mod1, selType="FLSE", selCrit="BIC")


## But the following does

mod1 <- selSLSE(mod1, selType="BLSE", selCrit="BIC")

## See one selection:

mod1$treated$selections$BLSE$JBIC
```

simDat1 *Simulated Data*

Description

This dataset is used in several documentation files to illustrate the different functionalities of the package.

Usage

```
data("simDat1")
```

Format

A data frame with 300 observations on the following 9 variables.

`X` Continuous Covariate

`Z` Treatment indicator.

`Y` Observed outcome.

`Y1` Outcome for the treated (not observed in practice).

`Y0` Outcome for the nontreated (not observed in practice).

`U01`, `U02` The true basis functions for the nontreated.

`U11`, `U12` The true basis functions for the treated.

simDat2 *Simulated Data*

Description

This dataset is used in several documentation files to illustrate the different functionalities of the package.

Usage

```
data("simDat2")
```

Format

A data frame with 300 observations on the following 11 variables.

`X` Continuous Covariate

`Z` Treatment indicator.

`Y` Observed outcome.

`Y1` Outcome for the treated (not observed in practice).

`Y0` Outcome for the nontreated (not observed in practice).

`U01`, `U02`, `U03` The true basis functions for the nontreated.

`U11`, `U12`, `U13` The true basis functions for the treated.

`simDat3` *Simulated Data*

Description

This dataset is used in several documentation files to illustrate the different functionalities of the package.

Usage

```
data("simDat3")
```

Format

A data frame with 300 observations on the following 16 variables.

`X1` Continuous Covariate

`X2` Continuous Covariate

`Z` Treatment indicator.

`Y` Observed outcome.

`Y1` Outcome for the treated (not observed in practice).

`Y0` Outcome for the nontreated (not observed in practice).

`U01`, `U02`, `U03`, `U04`, `U05` The true basis functions for the nontreated.

`U11`, `U12`, `U13`, `U14`, `U15` The true basis functions for the treated.

`simDat4` *Simulated Data.*

Description

This dataset is used in several documentation files to illustrate the different functionalities of the package.

Usage

```
data("simDat4")
```

Format

A data frame with 500 observations on the following 7 variables.

`Z` Treatment indicator.

`Y` Observed outcome.

`X1` Continuous covariates with a large proportion of zeros.

`X2` A categorical variable with 3 character levels (not expressed as factor).

`X3` A categorical variable with 3 numerical levels (not expressed as factor).

`X4` A binary variable.

`treat` A character variable.

`simDat5` *Simulated Data*

Description

This dataset is used in several documentation files to illustrate the different functionalities of the package.

Usage

```
data("simDat5")
```

Format

A data frame with 300 observations on the following 16 variables.

`X1` Continuous Covariate

`X2` Continuous Covariate

`Z` Treatment indicator.

`Y` Observed outcome.

`Y1` Outcome for the treated (not observed in practice).

`Y0` Outcome for the nontreated (not observed in practice).

slseKnots *Knots Creator for Basis Functions*

Description

The function creates an object of class slseKnots for semiparametric least squares models. It returns an object of class slseKnots.

Usage

```
slseKnots(form, data, X, nbasis = function(n) n^0.3,
          knots)
```

Arguments

form	A formula that determines the covariates to include in the regression. It is a regular formula and only the right hand side is considered.
data	A data.frame with all variables included in form. It is required when X is missing.
nbasis	A function to determine the number of basis functions. It has to be a function of one argument, the sample size.
knots	An optional list of knots. Each element of the list (one for each covariate) is either a vector of numeric knots or it is NULL. The latter implies that the number of knots is set to 0. If missing (the default), the knots are automatically generated (see the details below).
X	An optional matrix of covariates. When provided, form is not needed.

Details

The automatic selection is as follows. It is applied to each variable created by the model.matrix of form except for the intercept:

The number of knots is the ceiling of what the nbasis function returns minus 1. Let p-1 be the number of knots. Then, we compute the p+1 empirical quantiles of the variable for equally spaced probabilities going from 0 to 1 and drop the first and last ones. This is done using the function quantile with type=1. We then remove the duplicated values and the ones equal to either the min or the max of the variable. if the number of remaining knots is equal to 0, the set of knots for this variable is set to NULL.

For manual selection, see the vignette.

Value

It returns an object of class slseKnots. It is a list for which the length and names are respectively the number of columns and the column names of the model.matrix of form after the intercept has been removed. Each element is a numeric vector of knots, unless the number of knots is set to 0, in which case it is NULL.

Examples

```
data(simDat3)
k <- slseKnots(Y ~ X1 * X2, data = simDat3)
k

## We can extract the set for one variable

k$X1
```

---

summary *Summary Method for Fitted Models*

Description

The method computes summary statistics for estimated semiparametric and causal models.

Usage

```
## S3 method for class 'cslse'
summary(object, ...)
## S3 method for class 'cslseFit'
summary(object, vcov.=vcovHC, ...)
## S3 method for class 'slseFit'
summary(object, vcov.=vcovHC, ...)
```

Arguments

object	A model estimated by estSLSE or causalSLSE.
vcov.	A function to compute the strandard error of the least squares coefficients. The default is vcovHC.
...	Argument for other types of objects

Value

The function summary.slseFit returns the summary statistics of the least squares estimation of slseModel objects estimated by estSLSE. It is an object of class summary.slseFit, which is a list with the following elements:

model	The slseModel model being estimated.
lseSum	An object of class summary.lm.

The function summary.cslseFit returns an object of class summary.cslseFit, which is a list of summary.slseFit objects, one for each treatment group.

The function summary.cslse returns the summary statistics of the causal effect estimates. It is an object of class summary.cslse, which is a list with the following elements:

causal	A matrix with the causal effect estimates in the first column, their standard errors in the second, their t-ratios in the third and p-values in the fourth.
lse	A list of summary.slseFit objects, one for each treatment group.

Examples

```
## A causal semiparametric model with causal effect estimates

data(simDat3)
mod1 <- cslseModel(Y ~ Z | ~ X1 * X2, data = simDat3)
CE <- causalSLSE(mod1, selType = "BLSE")
summary(CE)

## A causal semiparametric model

fit <- estSLSE(mod1)
summary(fit)

## A semiparametric model

mod2 <- slseModel(Y ~ X1 * X2, data = simDat3)
fit2 <- estSLSE(mod2)
summary(fit2)
```

---

update *Update Methods*

Description

The method updates an object by modifying its specification. Currently, it is used to change the set of knots by either selecting them manually or by specifying the selection methods.

Usage

```
## S3 method for class 'cslseModel'
update(object, selType, selCrit="AIC",
                            selKnots, ...)

## S3 method for class 'slseModel'
update(object, selType, selCrit="AIC",
                           selKnots, ...)

## S3 method for class 'slseKnots'
update(object, selKnots, ...)
```

Arguments

object	An object to be modified.
selKnots	An optional list of integers to select the knots from the original list of knots. If missing, the current knots are kept.
selType	The selection method: "BLSE" for the backward method, "FLSE" for the forward method or "none" to recover the originnal set of knots.
selCrit	The criterion to select the optimal set of knots.
...	Argument for other types of objects. Currently not used.

Details

The method for slseKnots is explained in the vignette, but it is mostly used internally. For the model objects, the method can be used to choose a set of knots already stored in the object. It avoids having to re-compute them which can be computationally intensive for large samples. It returns an error message if the type of selection requested does not exist. In that case, you need to run the selSLSE method. If the selType argument is set to None, the method returns the original model.

Value

It returns an object of the same class, but with a different set of knots.

See Also

selSLSE for more details on how stored knots can be selected from a model, slseModel for model description and slseKnots for the format of knots

Examples

```
data(simDat3)
mod1 <- cslseModel(Y ~ Z | ~ X1 * X2, data = simDat3)
mod2 <- selSLSE(mod1, "BLSE", "AIC")

## We changed the knots to the BLSE-BIC selection
## already stored in the model object

update(mod2, "BLSE", "BIC")

## We recover the original set of knots

update(mod2, "None")
```

Index