Package ‘hdm’

March 22, 2016

Type Package

Title High-Dimensional Metrics

Version 0.1.1

Date 2016-01-31

Depends R (>= 3.0.0)

Description Implementation of selected high-dimensional statistical and econometric methods for estimation and inference. Efficient estimators and uniformly valid confidence intervals for various low-dimensional causal/structural parameters are provided which appear in high-dimensional approximately sparse models. Including functions for fitting heteroscedastic robust Lasso regressions with non-Gaussian errors and for instrumental variable (IV) and treatment effect estimation in a high-dimensional setting. Moreover, the methods enable valid post-selection inference and rely on a theoretically grounded, data-driven choice of the penalty.

License GPL-3

LazyData TRUE

Imports MASS,
glmnet,
ggplot2,
checkmate

Suggests testthat,
knitr,
xtable

VignetteBuilder knitr

R topics documented:

`hdm-package` *hdm: High-Dimensional Metrics*

Description

This package implements methods for estimation and inference in a high-dimensional setting.

Details

Package:	hdm
Type:	Package
Version:	0.1
Date:	2015-05-25
License:	GPL-3

This package provides efficient estimators and uniformly valid confidence intervals for various low-dimensional causal/structural parameters appearing in high-dimensional approximately sparse models. The package includes functions for fitting heteroskedastic robust Lasso regressions with non-Gaussian erros and for instrumental variable (IV) and treatment effect estimation in a high-dimensional setting. Moreover, the methods enable valid post-selection inference. Moreover, a theoretically grounded, data-driven choice of the penalty level is provided.

Author(s)

Victor Chernozhukov, Christian Hansen, Martin Spindler

Maintainer: Martin Spindler <spindler@mea.mpisoc.mpg.de>

References

A. Belloni, D. Chen, V. Chernozhukov and C. Hansen (2012). Sparse models and methods for optimal instruments with an application to eminent domain. *Econometrica* 80 (6), 2369-2429.

A. Belloni, V. Chernozhukov and C. Hansen (2013). Inference for high-dimensional sparse econometric models. In Advances in Economics and Econometrics: 10th World Congress, Vol. 3: Econometrics, Cambirdge University Press: Cambridge, 245-295.

A. Belloni, V. Chernozhukov, C. Hansen (2014). Inference on treatment effects after selection among high-dimensional controls. The Review of Economic Studies 81(2), 608-650.

`AJR` *AJR data set*

Description

Dataset on settler mortality.

Format

Mort Settler mortality

logMort logarithm of Mort

Latitude Latitude

Latitude2 Latitude^2

Africa Africa

Asia Asia

Namer North America

Samer South America

Neo Neo-Europes

GDP GDP

Exprop Average protection against expropriation risk

Details

Data set was analysed in Acemoglu et al. (2001). A detailed description of the data can be found at http://economics.mit.edu/faculty/acemoglu/data/ajr2001

References

D. Acemoglu, S. Johnson, J. A. Robinson (2001). Colonial origins of comparative development: an empirical investigation. American Economic Review, 91, 1369–1401.

Examples

```
data(AJR)
```

`EminentDomain` *Eminent Domain data set*

Description

Dataset on judicial eminent domain decisions.

Format

- **y** economic outcome variable
- **x** set of exogenous variables
- **d** eminent domain decisions
- **z** set of potential instruments

Details

Data set was analyzed in Belloni et al. (2012). They estimate the effect of judicial eminent domain decisions on economic outcomes with instrumental variables (IV) in a setting high a large set of potential IVs. A detailed decription of the data can be found at https://www.econometricsociety.org/publications/econometrica/2012/11/01/sparse-models-and-methods-optimal-instruments-applicat The data set contains four "sub-data sets" which differ mainly in the dependent variables: repeat-sales FHFA/OFHEO house price index for metro (FHFA) and non-metro (NM) area, the Case-Shiller home price index (CS), and state-level GDP from the Bureau of Economic Analysis - all transformed with the logarithm. The structure of each subdata set is comparable and given above.

References

D. Belloni, D. Chen, V. Chernozhukov and C. Hansen (2012). Sparse models and methods for optimal instruments with an application to eminent domain. *Econometrica* 80 (6), 2369–2429.

Examples

```
data(EminentDomain)
```

`Growth Data` *Growth data set*

Description

Data set of growth compiled by Barro Lee.

Format

Dataframe with the following variables:

dependent variable: national growth rates in GDP per capita for the periods 1965-1975 and 1975-1985

outcome covariates which might influence growth

Details

The data set contains growth data of Barro-Lee. The Barro Lee data consists of a panel of 138 countries for the period 1960 to 1985. The dependent variable is national growth rates in GDP per capita for the periods 1965-1975 and 1975-1985. The growth rate in GDP over a period from t_1 to t_2 is commonly defined as $\log(GDP_{t_1}/GDP_{t_2})$. The number of covariates is p=62. The number of complete observations is 90.

Source

The full data set and further details can be found at http://www.nber.org/pub/barro.lee, http://www.barrolee.com, and, http://www.bristol.ac.uk//Depts//Economics//Growth//barlee.htm.

References

R.J. Barro, J.W. Lee (1994). Data set for a panel of 139 countries. NBER.

R.J. Barro, X. Sala-i-Martin (1995). Economic Growth. McGrwa-Hill, New York.

Examples

```
data(GrwothData)
```

`lambdaCalculation` *Function for Calculation of the penalty parameter*

Description

This function implements different methods for calculation of the penalization parameter λ. Further details can be found under rlasso.

Usage

```
lambdaCalculation(penalty = list(homoscedastic = FALSE, X.dependent.lambda =
  FALSE, lambda.start = NULL, c = 1.1, gamma = 0.1), y = NULL, x = NULL)
```

Arguments

`penalty` list with options for the calculation of the penalty.
- `c` and `gamma` constants for the penalty with default `c=1.1` and `gamma=0.1`
- `homoscedastic` logical, if homoscedastic errors are considered (default `FALSE`). Option none is described below.
- `X.dependent.lambda` if `independent` or `dependent` design matrix `X` is assumed for calculation of the parameter λ
- `numSim` number of simulations for the X-dependent methods
- `lambda.start` initial penalization value, compulsory for method "none"

`y` residual which is used for calculation of the variance or the data-dependent loadings

`x` matrix of regressor variables

Value

The functions returns a list with the penalty `lambda` which is the product of `lambda0` and `Ups0`. `Ups0` denotes either the variance (`independent` case) or the data-dependent loadings for the regressors. `method` gives the selected method for the calculation.

`LassoShooting.fit` *Shooting Lasso*

Description

Implementation of the Shooting Lasso (Fu, 1998) with variable dependent penalization weights.

Usage

```
LassoShooting.fit(x, y, lambda, control = list(maxIter = 1000, optTol =
  10^(-5), zeroThreshold = 10^(-6)), XX = NULL, Xy = NULL,
  beta.start = NULL)
```

Arguments

`x`	matrix of regressor variables (`n` times `p` where `n` denotes the number of observations and `p` the number of regressors)
`y`	dependent variable (vector or matrix)
`lambda`	vector of length `p` of penalization parameters for each regressor
`control`	list with control parameters: `maxIter` maximal number of iterations, `optTol` tolerance for parameter precision, `zeroThreshold` threshold applied to the estimated coefficients for numerical issues.
`XX`	optional, precalculated matrix $t(X) * X$
`Xy`	optional, precalculated matrix $t(X) * y$
`beta.start`	start value for beta

Details

The function implements the Shooting Lasso (Fu, 1998) with variable dependent penalization. The arguments `XX` and `Xy` are optional and allow to use precalculated matrices which might improve performance.

Value

`coefficients`	estimated coefficients by the Shooting Lasso Algorithm
`coef.list`	matrix of coefficients from each iteration
`num.it`	number of iterations run

References

Fu, W. (1998). Penalized regressions: the bridge vs the lasso. *Journal of Computational and Graphical Software* 7, 397-416.

pension — *Pension 401(k) data set*

Description

Data set on financial wealth and 401(k) plan participation

Format

Dataframe with the following variables (amongst others):

p401 participation in 401(k)

e401 eligibility for 401(k)

a401 401(k) assets

tw total wealth (in US $)

tfa financial assets (in US $)

net_tfa net financial assets (in US $)

nifa non-401k financial assets (in US $)

net_nifa net non-401k financial assets

net_n401 net non-401(k) assets (in US $)

ira individual retirement account (IRA)

inc income (in US $)

age age

fsize family size

marr married

pira participation in IRA

db defined benefit pension

hown home owner

educ education (in years)

male male

twoearn two earners

nohs, hs, smcol, col dummies for education: no high-school, high-school, some college, college

hmort home mortage (in US $)

hequity home equity (in US $)

hval home value (in US $)

Details

The sample is drawn from the 1991 Survey of Income and Program Participation (SIPP) and consists of 9,915 observations. The observational units are household reference persons aged 25-64 and spouse if present. Households are included in the sample if at least one person is employed and no one is self-employed. The data set was analysed in Chernozhukov and Hansen (2004) and Belloni et al. (2014) where further details can be found. They examine the effects of 401(k) plans on wealth using data from the Survey of Income and Program Participation using 401(k) eligibility as an instrument for 401(k) participation.

References

V. Chernohukov, C. Hansen (2004). The impact of 401(k) participation on the wealth distribution: An instrumental quantile regression analysis. The Review of Economic and Statistics 86 (3), 735–751.

A. Belloni, V. Chernozhukov, I. Fernandez-Val, and C. Hansen (2014). Program evaluation with high-dimensional data. Working Paper.

Examples

```
data(pension)
```

---

predict.rlassologit *Methods for S3 object* rlassologit

Description

Objects of class rlassologit are constructed by rlassologit or rlassologit.fit. print.rlassologit prints and displays some information about fitted rlassologit objects. summary.rlassologit summarizes information of a fitted rlassologit object. predict.rlassologit predicts values based on a rlassologit object. model.matrix.rlassologit constructs the model matrix of a lasso object.

Usage

```
## S3 method for class 'rlassologit'
predict(object, newdata = NULL, type = "response",
  ...)

## S3 method for class 'rlassologit'
model.matrix(object, ...)

## S3 method for class 'rlassologit'
print(x, all = TRUE, digits = max(3L,
  getOption("digits") - 3L), ...)

## S3 method for class 'rlassologit'
summary(object, all = TRUE, digits = max(3L,
  getOption("digits") - 3L), ...)
```

Arguments

object	an object of class rlassologit
newdata	new data set for prediction
type	type of prediction required. The default ('response) is on the scale of the response variable; the alternative 'link' is on the scale of the linear predictors.
...	arguments passed to the print function and other methods
x	an object of class rlassologit
all	logical, indicates if coefficients of all variables (TRUE) should be displayed or only the non-zero ones (FALSE)
digits	significant digits in printout

`print.rlasso` *Methods for S3 object* `rlasso`

Description

Objects of class `rlasso` are constructed by `rlasso.formula` or `rlasso.fit`. `print.rlasso` prints and displays some information about fitted `rlasso` objects. `summary.rlasso` summarizes information of a fitted `rlasso` object. `predict.rlasso` predicts values based on a `rlasso` object. `model.matrix.rlasso` constructs the model matrix of a `rlasso` object.

Usage

```
## S3 method for class 'rlasso'
print(x, all = TRUE, digits = max(3L, getOption("digits") -
  3L), ...)

## S3 method for class 'rlasso'
summary(object, all = TRUE, digits = max(3L,
  getOption("digits") - 3L), ...)

## S3 method for class 'rlasso'
model.matrix(object, ...)

## S3 method for class 'rlasso'
predict(object, newdata = NULL, ...)
```

Arguments

`x`	an object of class `rlasso`
`all`	logical, indicates if coefficients of all variables (TRUE) should be displayed or only the non-zero ones (FALSE)
`digits`	significant digits in printout
`...`	arguments passed to the print function and other methods
`object`	an object of class `rlasso`
`newdata`	new data set for prediction. An optional data frame in which to look for variables with which to predict. If omitted, the fitted values are returned.

`print.rlassoEffects` *Methods for S3 object* `rlassoEffects`

Description

Objects of class `rlassoEffects` are constructed by `rlassoEffects`. `print.rlassoEffects` prints and displays some information about fitted `rlassoEffect` objects. summary.rlassoEffects summarizes information of a fitted `rlassoEffect` object and is described at `summary.rlassoEffects`. `confint.rlassoEffects` extracts the confidence intervals. `plot.rlassoEffects` plots the estimates with confidence intervals.

Usage

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

## S3 method for class 'rlassoEffects'
confint(object, parm, level = 0.95, joint = FALSE,
  ...)

## S3 method for class 'rlassoEffects'
plot(x, main = "", xlab = "coef", ylab = "",
  xlim = NULL, ...)
```

Arguments

`x`	an object of class `rlassoEffects`
`digits`	significant digits in printout
`...`	arguments passed to the print function and other methods.
`object`	an object of class `rlassoEffects`
`parm`	a specification of which parameters are to be given confidence intervals among the variables for which inference was done, either a vector of numbers or a vector of names. If missing, all parameters are considered.
`level`	confidence level required
`joint`	logical, if `TRUE` joint confidence intervals are calculated.
`main`	an overall title for the plot
`xlab`	a title for the x axis
`ylab`	a title for the y axis
`xlim`	vector of length two giving lower and upper bound of x axis

---

`print.rlassoIV` *Methods for S3 object* `rlassoIV`

Description

Objects of class `rlassoIV` are constructed by `rlassoIV`. `print.rlassoIV` prints and displays some information about fitted `rlassoIV` objects. `summary.rlassoIV` summarizes information of a fitted `rlassoIV` object. `confint.rlassoIV` extracts the confidence intervals.

Usage

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

## S3 method for class 'rlassoIV'
summary(object, digits = max(3L, getOption("digits") - 3L),
  ...)

## S3 method for class 'rlassoIV'
confint(object, parm, level = 0.95, ...)
```

Arguments

`x`	an object of class `rlassoIV`
`digits`	significant digits in printout
`...`	arguments passed to the print function and other methods
`object`	An object of class `rlassoIV`
`parm`	a specification of which parameters are to be given confidence intervals, either a vector of numbers or a vector of names. If missing, all parameters are considered.
`level`	confidence level required.

`print.rlassoIVselectX` *Methods for S3 object* `rlassoIVselectX`

Description

Objects of class `rlassoIVselectX` are constructed by `rlassoIVselectX`. `print.rlassoIVselectX` prints and displays some information about fitted `rlassoIVselectX` objects. `summary.rlassoIVselectX` summarizes information of a fitted `rlassoIVselectX` object. `confint.rlassoIVselectX` extracts the confidence intervals.

Usage

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

## S3 method for class 'rlassoIVselectX'
summary(object, digits = max(3L, getOption("digits")
  - 3L), ...)

## S3 method for class 'rlassoIVselectX'
confint(object, parm, level = 0.95, ...)
```

Arguments

`x`	an object of class `rlassoIVselectX`
`digits`	significant digits in printout
`...`	arguments passed to the print function and other methods
`object`	an object of class `rlassoIVselectX`
`parm`	a specification of which parameters are to be given confidence intervals, either a vector of numbers or a vector of names. If missing, all parameters are considered.
`level`	the confidence level required.

print.rlassoIVselectZ *Methods for S3 object* rlassoIVselectZ

Description

Objects of class `rlassoIVselectZ` are constructed by `rlassoIVselectZ`. `print.rlassoIVselectZ` prints and displays some information about fitted `rlassoIVselectZ` objects. `summary.rlassoIVselectZ` summarizes information of a fitted `rlassoIVselectZ` object. `confint.rlassoIVselectZ` extracts the confidence intervals.

Usage

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

## S3 method for class 'rlassoIVselectZ'
summary(object, digits = max(3L, getOption("digits")
  - 3L), ...)

## S3 method for class 'rlassoIVselectZ'
confint(object, parm, level = 0.95, ...)
```

Arguments

`x`	an object of class `rlassoIVselectZ`
`digits`	significant digits in printout
`...`	arguments passed to the print function and other methods
`object`	an object of class `rlassoIVselectZ`
`parm`	a specification of which parameters are to be given confidence intervals, either a vector of numbers or a vector of names. If missing, all parameters are considered.
`level`	confidence level required.

print.rlassologitEffects *Methods for S3 object* rlassologitEffects

Description

Objects of class `rlassologitEffects` are construced by `rlassologitEffects` or `rlassologitEffect`. `print.rlassologitEffects` prints and displays some information about fitted `rlassologitEffect` objects. `summary.rlassologitEffects` summarizes information of a fitted `rlassologitEffects` object. `confint.rlassologitEffects` extracts the confidence intervals.

Usage

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

## S3 method for class 'rlassologitEffects'
summary(object, digits = max(3L,
  getOption("digits") - 3L), ...)

## S3 method for class 'rlassologitEffects'
confint(object, parm, level = 0.95,
  joint = FALSE, ...)
```

Arguments

`x`	an object of class `rlassologitEffects`
`digits`	number of significant digits in printout
`...`	arguments passed to the print function and other methods
`object`	an object of class `rlassologitEffects`
`parm`	a specification of which parameters are to be given confidence intervals, either a vector of numbers or a vector of names. If missing, all parameters are considered.
`level`	confidence level required.
`joint`	logical, if joint confidence intervals should be clalculated

---

`print.rlassoTE` *Methods for S3 object* `rlassoTE`

Description

Objects of class `rlassoTE` are constructed by `rlassoATE`, `rlassoATET`, `rlassoLATE`, `rlassoLATET`. `print.rlassoTE` prints and displays some information about fitted `rlassoTE` objects. `summary.rlassoTE` summarizes information of a fitted `rlassoTE` object. `confint.rlassoTE` extracts the confidence intervals.

Usage

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

## S3 method for class 'rlassoTE'
summary(object, digits = max(3L, getOption("digits") - 3L),
  ...)

## S3 method for class 'rlassoTE'
confint(object, parm, level = 0.95, ...)
```

Arguments

`x`	an object of class `rlassoTE`
`digits`	number of significant digits in printout
`...`	arguments passed to the print function and other methods
`object`	an object of class `rlassoTE`
`parm`	a specification of which parameters are to be given confidence intervals, either a vector of numbers or a vector of names. If missing, all parameters are considered.
`level`	confidence level required.

`print.tsls` *Methods for S3 object* `tsls`

Description

Objects of class `tsls` are constructed by `tsls`. `print.tsls` prints and displays some information about fitted `tsls` objects. `summary.tsls` summarizes information of a fitted `tsls` object.

Usage

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

## S3 method for class 'tsls'
summary(object, digits = max(3L, getOption("digits") - 3L),
  ...)
```

Arguments

`x`	an object of class `tsls`
`digits`	significant digits in printout
`...`	arguments passed to the print function and other methods
`object`	an object of class `tsls`

`rlasso` *rlasso: Function for Lasso estimation under homoscedastic and heterosceadstic non-Gaussian disturbances*

Description

The function estimates the coefficients of a Lasso regression with data-driven penalty under homoscedasticity and heteroscedasticity with non-Gaussian noise and X-dependent or X-independent design. The method of the data-driven penalty can be chosen. The object which is returned is of the S3 class `rlasso`.

Usage

```
rlasso(formula, data, post = TRUE, intercept = TRUE, model = TRUE,
  penalty = list(homoscedastic = FALSE, X.dependent.lambda = FALSE,
  lambda.start = NULL, c = 1.1, gamma = 0.1/log(n)), control = list(numIter =
  15, tol = 10^-5, threshold = NULL), ...)

rlasso.fit(x, y, post = TRUE, intercept = TRUE, model = TRUE,
  penalty = list(homoscedastic = FALSE, X.dependent.lambda = FALSE,
  lambda.start = NULL, c = 1.1, gamma = 0.1), control = list(numIter = 15, tol
  = 10^-5, threshold = NULL), ...)
```

Arguments

`formula`	an object of class "formula" (or one that can be coerced to that class): a symbolic description of the model to be fitted in the form `y~x`
`data`	an optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which `rlasso` is called.
`post`	logical. If `TRUE`, post-Lasso estimation is conducted.
`intercept`	logical. If `TRUE`, intercept is included which is not penalized.
`model`	logical. If `TRUE` (default), model matrix is returned.
`penalty`	list with options for the calculation of the penalty. • `c` and `gamma` constants for the penalty with default `c=1.1` and `gamma=0.1` • `homoscedastic` logical, if homoscedastic errors are considered (default `FALSE`). Option none is described below. • `X.dependent.lambda` logical, `TRUE`, if the penalization parameter depends on the the design of the matrix `x`. `FALSE`, if independent of the design matrix (default). • `numSim` number of simulations for the dependent methods, default=5000 • `lambda.start` initial penalization value, compulsory for method "none"
`control`	list with control values. `numIter` number of iterations for the algorithm for the estimation of the variance and data-driven penalty, ie. loadings, `tol` tolerance for improvement of the estimated variances. `threshold` is applied to the final estimated lasso coefficients. Absolute values below the threshold are set to zero.
`...`	further arguments (only for consistent defintion of methods)
`x`	regressors (vector, matrix or object can be coerced to matrix)
`y`	dependent variable (vector, matrix or object can be coerced to matrix)

Details

The function estimates the coefficients of a Lasso regression with data-driven penalty under homoscedasticity / heteroscedasticity and non-Gaussian noise. The options `homoscedastic` is a logical with `FALSE` by default. Moreover, for the calculation of the penalty parameter it can be chosen, if the penalization parameter depends on the design matrix (`X.dependent.lambda=TRUE`) or `independent` (default, `X.dependent.lambda=FALSE`). The default value of the constant `c` is `1.1` in the post-Lasso case and `0.5` in the Lasso case. A *special* option is to set `homoscedastic` to `none` and to supply a values `lambda.start`. Then this value is used as penalty parameter with independent design and heteroscedastic errors to weight the regressors. For details of the implementation of the Algorithm for estimation of the data-driven penalty, in particular the regressor-independent loadings, we refer to Appendix A in Belloni et al. (2012). When the option "none" is chosen for `homoscedastic` (together with `lambda.start`), lambda is set to `lambda.start` and the regressor-independent loadings und heteroscedasticity are used. The options "X-dependent" and "X-independent" under homoscedasticity are described in Belloni et al. (2013).

The option `post=TRUE` conducts post-lasso estimation, i.e. a refit of the model with the selected variables.

Value

`rlasso` returns an object of class `rlasso`. An object of class "rlasso" is a list containing at least the following components:

`coefficients`	parameter estimates (named vector of coefficients without intercept)
`intercept.value`	value of the intercept
`index`	index of selected variables (logical vector)
`lambda`	data-driven penalty term for each variable, product of lambda0 (the penalization parameter) and the loadings
`lambda0`	penalty term
`loadings`	loading for each regressor
`residuals`	residuals, response minus fitted values
`sigma`	root of the variance of the residuals
`iter`	number of iterations
`call`	function call
`options`	options

References

A. Belloni, D. Chen, V. Chernozhukov and C. Hansen (2012). Sparse models and methods for optimal instruments with an application to eminent domain. *Econometrica* 80 (6), 2369-2429.

A. Belloni, V. Chernozhukov and C. Hansen (2013). Inference for high-dimensional sparse econometric models. In Advances in Economics and Econometrics: 10th World Congress, Vol. 3: Econometrics, Cambirdge University Press: Cambridge, 245-295.

`rlassoATE` *Functions for estimation of treatment effects*

Description

This class of functions estimates the average treatment effect (ATE), the ATE of the tretated (ATET), the local average treatment effects (LATE) and the LATE of the tretated (LATET). The estimation methods rely on immunized / orthogonal moment conditions which guarantee valid post-selection inference in a high-dimensional setting. Further details can be found in Belloni et al. (2014).

Usage

```
rlassoATE(x, d, y, bootstrap = "none", nRep = 500, ...)

rlassoATET(x, d, y, bootstrap = "none", nRep = 500, ...)

rlassoLATE(x, d, y, z, bootstrap = "none", nRep = 500, post = TRUE,
  intercept = TRUE)

rlassoLATET(x, d, y, z, bootstrap = "none", nRep = 500, post = TRUE,
  intercept = TRUE)
```

Arguments

`x`	exogenous variables
`d`	treatment variable (binary)
`y`	outcome variable / dependent variable
`bootstrap`	boostrap method which should be employed: 'none', 'Bayes', 'normal', 'wild'
`nRep`	number of replications for the bootstrap
`...`	arguments passed, e.g. `intercept` and `post`
`z`	instrumental variables (binary)
`post`	logical. If `TRUE`, post-lasso estimation is conducted.
`intercept`	logical. If `TRUE`, intercept is included which is not penalized.

Details

Details can be found in Belloni et al. (2014).

Value

Functions return an object of class `rlassoTE` with estimated effects, standard errors and individual effects in the form of a `list`.

References

A. Belloni, V. Chernozhukov, I. Fernandez-Val, and C. Hansen (2014). Program evaluation with high-dimensional data. Working Paper.

`rlassoEffects` *rigorous Lasso for Linear Models: Inference*

Description

Estimation and inference of (low-dimensional) target coefficients in a high-dimensional linear model.

Usage

```
rlassoEffects(x, y, index = c(1:ncol(x)), method = "partialling out",
  I3 = NULL, post = TRUE, ...)

rlassoEffect(x, y, d, method = "double selection", I3 = NULL, post = TRUE,
  ...)
```

Arguments

`x`	matrix of regressor variables serving as controls and potential treatments. For `rlassoEffect` it contains only controls, for `rlassoEffects` both controls and potential treatments. For `rlassoEffects` it must have at least two columns.
`y`	outcome variable (vector or matrix)
`index`	vector of integers, logicals or variables names indicating the position (column) of variables (integer case), logical vector of length of the variables (TRUE or FALSE) or the variable names of `x` which should be used for inference / as treatment variables.
`method`	method for inference, either 'partialling out' (default) or 'double selection'.
`I3`	For the 'double selection'-method the logical vector `I3` has same length as the number of variables in `x`; indicates if variables (TRUE) should be included in any case to the model and they are exempt from selection. These variables should not be included in the `index`; hence the intersection with `index` must be the empty set. In the case of partialling out it is ignored.
`post`	logical, if post Lasso is conducted with default `TRUE`.
`d`	variable for which inference is conducted (treatment variable)
`...`	parameters passed to the `rlasso` function.

Details

The functions estimates (low-dimensional) target coefficients in a high-dimensional linear model. An application is e.g. estimation of a treatment effect α_0 in a setting of high-dimensional controls. The user can choose between the so-called post-double-selection method and partialling-out. The idea of the double selection method is to select variables by Lasso regression of the outcome variable on the control variables and the treatment variable on the control variables. The final estimation is done by a regression of the outcome on the treatment effect and the union of the selected variables in the first two steps. In partialling-out first the effect of the regressors on the outcome and the treatment variable is taken out by Lasso and then a regression of the residuals is conducted. The resulting estimator for α_0 is normal distributed which allows inference on the treatment effect. It presents a wrap function for `rlassoEffect` which does inference for a single variable.

Value

The function returns an object of class `rlassoEffects` with the following entries:

`coefficients`	vector with estimated values of the coefficients for each selected variable
`se`	standard error (vector)
`t`	t-statistic
`pval`	p-value
`samplesize`	sample size of the data set
`index`	index of the variables for which inference is performed

References

A. Belloni, V. Chernozhukov, C. Hansen (2014). Inference on treatment effects after selection among high-dimensional controls. The Review of Economic Studies 81(2), 608-650.

Examples

```
library(hdm)
## DGP
n <- 250
p <- 100
px <- 10
X <- matrix(rnorm(n*p), ncol=p)
beta <- c(rep(2,px), rep(0,p-px))
intercept <- 1
y <- intercept + X %*% beta + rnorm(n)
## fit rlassoEffects object with inference on three variables
rlassoEffects.reg <- rlassoEffects(x=X, y=y, index=c(1,7,20))
## methods
summary(rlassoEffects.reg)
confint(rlassoEffects.reg, level=0.9)
```

`rlassoIV` *Post-Selection and Post-Regularization Inference in Linear Models with Many Controls and Instruments*

Description

The function estimates a treatment effect in a setting with very many controls and very many instruments (even larger than the sample size).

Usage

```
rlassoIV(x, d, y, z, select.Z = TRUE, select.X = TRUE, post = TRUE, ...)

rlassoIVmult(x, d, y, z, select.Z = TRUE, select.X = TRUE, ...)
```

Arguments

`x`	matrix of exogenous variables
`d`	endogenous variable
`y`	outcome / dependent variable (vector or matrix)
`z`	matrix of instrumental variables
`select.Z`	logical, indicating selection on the instruments
`select.X`	logical, indicating selection on the exogenous variables
`post`	logical, wheter post-Lasso should be conducted (default=`TRUE`)
`...`	arguments passed to the function `rlasso`

Details

The implementation for selection on `x` and `z` follows the procedure described in Chernozhukov et al. (2015) and is built on 'triple selection' to achieve an orthogonal moment function. The function returns an object of S3 class `rlassoIV`. Moreover, it is wrap function for the case that selection should be done only with the instruments Z (`rlassoIVselectZ`) or with the control variables X (`rlassoIVselectX`) or without selection (`tsls`). Exogenous variables `x` are automatically used as instruments and added to the instrument set `z`.

Value

an object of class `rlassoIV` containing at least the following components:

`coefficients`	estimated parameter value
`se`	variance-covariance matrix

References

V. Chernozhukov, C. Hansen, M. Spindler (2015). Post-selection and post-regularization inference in linear models with many controls and instruments. American Economic Review: Paper & Proceedings 105(5), 486–490.

`rlassoIVselectX` *Instrumental Variable Estimation with Selection on the exogenous Variables by Lasso*

Description

This function estimates the coefficient of an endogenous variable by employing Instrument Variables in a setting where the exogenous variables are high-dimensional and hence selection on the exogenous variables is required. The function returns an element of class `rlassoIVselectX`

Usage

```
rlassoIVselectX(x, d, y, z, post = TRUE, ...)
```

Arguments

`x`	exogenous variables in the structural equation (matrix)
`d`	endogenous variables in the structural equation (vector or matrix)
`y`	outcome or dependent variable in the structural equation (vector or matrix)
`z`	set of potential instruments for the endogenous variables.
`post`	logical. If `TRUE`, post-lasso estimation is conducted.
`...`	arguments passed to the function `rlasso`

Details

The implementation is a special case of of Chernozhukov et al. (2015). The option `post=TRUE` conducts post-lasso estimation for the Lasso estimations, i.e. a refit of the model with the selected variables. Exogenous variables `x` are automatically used as instruments and added to the instrument set `z`.

Value

An object of class `rlassoIVselectX` containing at least the following components:

`coefficients`	estimated parameter vector
`vcov`	variance-covariance matrix
`residuals`	residuals
`samplesize`	sample size

References

Chernozhukov, V., Hansen, C. and M. Spindler (2015). Post-Selection and Post-Regularization Inference in Linear Models with Many Controls and Instruments *American Economic Review, Papers and Proceedings* 105(5), 486–490.

`rlassoIVselectZ` *Instrumental Variable Estimation with Lasso*

Description

This function selects the instrumental variables in the first stage by Lasso. First stage predictions are then used in the second stage as optimal instruments to estimate the parameter vector. The function returns an element of class `rlassoIVselectZ`

Usage

```
rlassoIVselectZ(x, d, y, z, post = TRUE, ...)
```

Arguments

`x`	exogenous variables in the structural equation (matrix)
`d`	endogenous variables in the structural equation (vector or matrix)
`y`	outcome or dependent variable in the structural equation (vector or matrix)
`z`	set of potential instruments for the endogenous variables. Exogenous variables serve as their own instruments.
`post`	logical. If `TRUE`, post-lasso estimation is conducted.
`...`	arguments passed to the function `rlasso`

Details

The implementation follows the procedure described in Belloni et al. (2012). Option `post=TRUE` conducts post-lasso estimation, i.e. a refit of the model with the selected variables, to estimate the optimal instruments. The parameter vector of the structural equation is then fitted by two-stage least square (tsls) estimation.

Value

An object of class `rlassoIVselectZ` containing at least the following components:

`coefficients`	estimated parameter vector
`vcov`	variance-covariance matrix
`residuals`	residuals
`samplesize`	sample size

References

D. Belloni, D. Chen, V. Chernozhukov and C. Hansen (2012). Sparse models and methods for optimal instruments with an application to eminent domain. *Econometrica* 80 (6), 2369–2429.

`rlassologit` *rlassologit: Function for logistic Lasso estimation*

Description

The function estimates the coefficients of a logistic Lasso regression with data-driven penalty. The method of the data-driven penalty can be chosen. The object which is returned is of the S3 class `rlassologit`

Usage

```
rlassologit(formula, data, post = TRUE, intercept = TRUE,
  penalty = list(lambda = NULL, c = 1.1, gamma = 0.1/log(n)),
  control = list(threshold = NULL), ...)

rlassologit.fit(x, y, post = TRUE, intercept = TRUE, penalty = list(lambda
  = NULL, c = 1.1, gamma = 0.1/log(n)), control = list(threshold = NULL), ...)
```

Arguments

`formula`	an object of class 'formula' (or one that can be coerced to that class): a symbolic description of the model to be fitted in the form `y~x`
`data`	an optional data frame, list or environment
`post`	logical. If `TRUE`, post-lasso estimation is conducted.
`intercept`	logical. If `TRUE`, intercept is included which is not penalized.
`penalty`	list with options for the calculation of the penalty. `c` and `gamma` constants for the penalty.
`control`	list with control values. `threshold` is applied to the final estimated lasso coefficients. Absolute values below the threshold are set to zero.
`...`	further parameters passed to glmnet
`x`	regressors (matrix)
`y`	dependent variable (vector or matrix)

Details

The function estimates the coefficients of a Logistic Lasso regression with data-driven penalty. The option `post=TRUE` conducts post-lasso estimation, i.e. a refit of the model with the selected variables.

Value

`rlassologit` returns an object of class `rlassologit`. An object of class `rlassologit` is a list containing at least the following components:

`coefficients`	parameter estimates (without intercept)
`a0`	value of intercept
`index`	index of selected variables (logicals)
`lambda`	penalty term

`residuals`	residuals
`sigma`	root of the variance of the residuals
`call`	function call
`options`	options

References

Belloni, A., Chernozhukov and Y. Wei (2013). Honest confidence regions for logistic regression with a large number of controls. arXiv preprint arXiv:1304.3969.

Examples

```
## Not run:
library(hdm)
## DGP
set.seed(2)
n <- 250
p <- 100
px <- 10
X <- matrix(rnorm(n*p), ncol=p)
beta <- c(rep(2,px), rep(0,p-px))
intercept <- 1
P <- exp(intercept + X %*% beta)/(1+exp(intercept + X %*% beta))
y <- numeric(length=250)
for(i in 1:n){
  y[i] <- sample(x=c(1,0), size=1, prob=c(P[i],1-P[i]))
}
## fit rlassologit object
 rlassologit.reg <- rlassologit(y~X)
 ## methods
summary(rlassologit.reg, all=F)
print(rlassologit.reg)
predict(rlassologit.reg, type='response')
X3 <- matrix(rnorm(n*p), ncol=p)
predict(rlassologit.reg, newdata=X3)

## End(Not run)
```

`rlassologitEffects` *rigorous Lasso for Logistic Models: Inference*

Description

The function estimates (low-dimensional) target coefficients in a high-dimensional logistic model.

Usage

```
rlassologitEffects(x, y, index = c(1:ncol(x)), I3 = NULL, ...)

rlassologitEffect(x, y, d, I3 = NULL)
```

Arguments

x	matrix of regressor variables serving as controls and potential treatments. For `rlassologitEffect` it contains only controls, for `rlassologitEffects` both controls and potential treatments. For `rlassologitEffects` it must have at least two columns.
y	outcome variable
index	vector of integers, logical or names indicating the position (column) or name of variables of x which should be used as treatment variables.
I3	logical vector with same length as the number of controls; indicates if variables (TRUE) should be included in any case.
d	variable for which inference is conducted (treatment variable)
...	additional parameters

Details

The functions estimates (low-dimensional) target coefficients in a high-dimensional logistic model. An application is e.g. estimation of a treatment effect α_0 in a setting of high-dimensional controls. The function is a wrap function for `rlassologitEffect` which does inference for only one variable (d).

Value

The function returns an object of class `rlassologitEffects` with the following entries:

coefficients	estimated value of the coefficients
se	standard errors
t	t-statistics
pval	p-values
samplesize	sample size of the data set
I	index of variables of the union of the lasso regressions

References

A. Belloni, V. Chernozhukov, Y. Wei (2013). Honest confidence regions for a regression parameter in logistic regression with a loarge number of controls. cemmap working paper CWP67/13.

`summary.rlassoEffects` *Summarizing rlassoEffects fits*

Description

Summary method for class `rlassoEffects`

Usage

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

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

Arguments

`object`	an object of class `rlassoEffects`, usually a result of a call to `rlassoEffects`
`...`	further arguments passed to or from other methods.
`x`	an object of class `summary.rlassoEffects`, usually a result of a call or `summary.rlassoEffects`
`digits`	the number of significant digits to use when printing.

Details

Summary of objects of class `rlassoEffects`

`tsls` *Two-Stage Least Squares Estimation (TSLS)*

Description

The function does Two-Stage Least Squares Estimation (TSLS).

Usage

```
tsls(y, d, x, z, intercept = TRUE, homoscedastic = TRUE)
```

Arguments

`y`	outcome variable
`d`	endogenous variables
`x`	exogenous variables
`z`	instruments
`intercept`	logical, if intercept should be included
`homoscedastic`	logical, if homoscedastic (`TRUE`, default) or heteroscedastic erros (`FALSE`) should be calculated.

Details

The function computes tsls estimate (coefficients) and variance-covariance-matrix assuming homoskedasticity for outcome variable `y` where `d` are endogenous variables in structural equation, `x` are exogensous variables in structural equation and `z` are instruments. It returns an object of class `tsls` for which the methods `print` and `summary` are provided.

Value

The function returns a list with the following elements

`coefficients`	coefficients
`vcov`	variance-covariance matrix
`residuals`	outcome minus predicted values
`call`	function call
`samplesize`	sample size
`se`	standard error

Index